Beskriv förändring med hjälp av derivator

Många frågeställningar handlar om hur snabbt ett visst förlopp ändras. T.ex. "Hur fort kör bilen?", "Hur snabbt sönderfaller ett radioaktivt ämne?", "Hur snabbt sjunker lufttrycket med stigande höjd över havet?", etc. Vi uttrycker förändringar med derivator.

Derivatan $f'(x)$ av en funktion $f(x)$ mäter känsligheten för förändring i punkten x. Om vi t.ex. låter $f(x)$ vara positionen för ett visst (rörligt) objekt i tiden x, så är derivatan $f'(x)$ ett mått på objektets hastighet i tiden x. Ett mått på hur snabbt objektets position ändras när tiden går. Objektets acceleration $f''(x)$ anger förändring av hastigheten per tidsenhet.

Exempel: I en behållare finns 200 liter av en saltlösning som från början har koncentrationen 10 g/l. Genom ett rör tillförs 8 l/h av en saltlösning med koncentraionen 2g/l. Genom ett annat rör avtappas samtidigt lika stor mängd av lösningen i behållaren. Lösningen hålls hela tiden homogen genom omrörning. Vi vill bestämma saltmängden $y(t)$ (gram) i behållaren vid tiden t timmar efter starten.

Saltmängdens förändring per tidsenhet $y'(t)$ kan uttryckas som den tillförda saltmängden $(8 \cdot 2 = 16g/h)$ minus den bortförda saltmängden $(8 \cdot y(t)/200 = 1/25 \cdot y(t)g/h)$, så

$$y'(t) = 16 - \frac{y(t)}{25}$$

Vi vet också att saltmängden när vi startar $y(0) = 200 \cdot 10 = 2000g$. Sambandet

$$\begin{aligned} y'(t) &= 16 - \frac{y(t)}{25} \\ y(0) &= 2000 \end{aligned}$$

utgör en matematisk modell som beskriver saltmängden $y(t)$ i behållaren. För att kunna bestämma saltmängden vid en viss tidpunkt behöver vi lösa differentialekvationen.

Ordinära differentialekvationer

En differentialekvation är en ekvation där den sökta funktionen $(u(t))$ är given genom sina derivator. När man bestämmer en analytisk lösning bestämmer man ett uttryck för funktionen, utan funktionens derivator.

Exempel: Låt

$$u'(t) = u(t)$$

Vi är ute efter en funktion vars derivata är lika med funktionen självt. Dvs. lösningen är $u(t) = Ce^t$ där C är en konstant.

Exempel: Differentialekvationen

$$u'(t) = -u(t) + \sin(t) + \cos(t)$$

har den allmänna lösningen

$$u(t) = \sin(t) + Ce^{-t}$$

där C är en konstant. (Vi kommer att gå igenom hur man beräknar den allmänna lösningen till exemplet lite senare i kursen.) Kontrollera lösningen genom att derivera den allmänna lösningen:

$$u'(t) = \cos(t) - Ce^{-t} = \cos(t) - Ce^{-t} + \sin(t) - \sin(t) =$$
$$-\sin(t) - Ce^{-t} + \sin(t) + \cos(t) = -u(t) + \sin(t) + \cos(t)$$

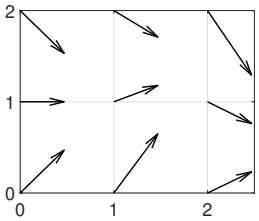
Riktningsfält

När man ritar ett riktningsfält skapar man ett antal punkter $(t_i, u(t_i))$ i t, u-planet. I varje punkt ritar man en pil (eller ett litet streck) i den riktning som lösningskurvan $(t_i, u(t_i))$ har i just den punkten.

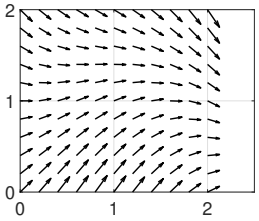
Exempel: Låt

$$u'(t) = -u(t) + \sin(t) + \cos(t)$$

t	u	$u'(t) = -u(t) + \sin(t) + \cos(t)$
0	0	$u' = 0 + \sin(0) + \cos(0) = 1$
0	1	$u' = -1 + \sin(0) + \cos(0) = 0$
0	2	$u' = -2 + \sin(0) + \cos(0) = -1$
1	0	$u' = 0 + \sin(1) + \cos(1) \approx 1.38$
$\vdots$	$\vdots$	$\vdots$



Samma riktningsfält med fler punkter:



Genom att införa ett begynnelsevärde fixerar man en av lösningarna.

Exempel: Differentialekvationen

$$\begin{cases} u'(t) = -u(t) + \sin(t) + \cos(t) \\ u(0) = 1 \end{cases}$$

har allmänna lösningen $u(t) = \sin(t) + Ce^{-t}$ där C är en konstant. Begynnelsevärdet, $u(0) = 1$, bestämmer värdet på konstanten:

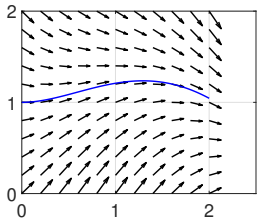
$$u(0) = 1 \Rightarrow \sin(0) + Ce^0 = 1, \text{ dvs. } C = 1$$

Så $u(t) = \sin(t) + e^{-t}$

I figuren till höger har vi ritat lösningen

$$u(t) = \sin(t) + e^{-t}$$

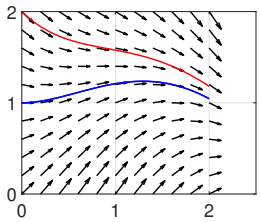
Lösningen följer pilarna i riktningsfältet.



Exempel: Begynnelsevärdesproblemet

$$\begin{cases} u'(t) = -u(t) + \sin(t) + \cos(t) \\ u(0) = 2 \end{cases}$$

har lösning $u(t) = \sin(t) + 2e^{-t}$. Lösningen har markerats i riktningsfältet (den röda kurvan). Lösningen följer pilarna i riktningsfältet.



Numerisk lösning av differentialekvationer

Numerisk lösning av differentialekvationer bygger på att följa lösningskurvor i riktningsfält så nogrannt som möjligt.

Ett begynnelsevärdesproblem formuleras allmänt så här:

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

- $a \leq t \leq b$ är intervallet man är intresserad av.
- u_a är begynnelsevärdet, funktionsvärdet i a.
- $f(t, u)$, derviatan $(u'(t))$ är en funktion som beror av både t och $u(t)$.

Eulers metod för numerisk lösning av begynnelsevärdesproblem:

A: Dela in intervallet $[a, b]$ i N stycken delar med steglängden $h = \dfrac{b-a}{N}$.
$t_0 = a, t_1 = t_0 + h, t_2 = t_0 + 2h, \ldots, t_N = b$

B: Beräkna funktionsvärden $u(t_0), u(t_1), u(t_2) \ldots u(b)$ genom iterationen

$$\begin{cases} u(t_0) = 0 \\ u(t_{i+1}) = u(t_i) + h \cdot f(t_i, u(t_i)) \end{cases} \quad i = 0, 1, 2, ...$$

Exempel: Låt

$$\begin{cases} u'(t) = -u(t) + \sin(t) + \cos(t) \\ u(0) = 1 \end{cases} \quad 0 \leq t \leq 2$$

Beräkna en lösning med Eulers metod, låt steglängden $h = 0.5$.

Vi har $t_0 = 0, t_1 = 0.5, t_2 = 1, t_3 = 1.5, t_4 = 2$.

$$u(t_0) = u(0) = 1$$

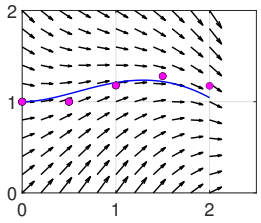
$$u(t_1) = u(t_0) + h \cdot f(t_0, u(t_0)) = 1 + 0.5 \cdot (-1 + \sin(0) + \cos(0)) = 1$$

$$u(t_2) = u(t_1) + h \cdot f(t_1, u(t_1)) = 1 + 0.5 \cdot (-1 + \sin(0.5) + \cos(0.5)) \approx 1.2$$

$$u(t_3) = u(t_2) + h \cdot f(t_2, u(t_2)) = 1.2 + 0.5 \cdot (-1.2 + \sin(1) + \cos(1)) \approx 1.3$$

$$u(t_4) = u(t_3) + h \cdot f(t_3, u(t_3)) = 1.3 + 0.5 \cdot (-1.3 + \sin(1.5) + \cos(1.5)) \approx 1.2$$

I figuren till höger har vi markerat lösningen vi får med Eulers metod med •. Våra värden avviker från den blå lösningskurvan lite. Om vi valt kortare steglängd hade vår lösning följt lösningskurvan bättre.



Man kan utgå från definitionen av derivatan när man härleder Eulers metod.

$$u'(t_i) = \lim_{h \to 0} \frac{u(t_{i+1}) - u(t_i)}{h}, \quad h = t_{i+1} - t_i$$

Om h tillräckligt litet så gäller

$$u'(t_i) \approx \frac{u(t_{i+1}) - u(t_i)}{h}$$

Lös ut $u(t_{i+1})$

$$u'(t_i) = \frac{u(t_{i+1}) - u(t_i)}{h} \Leftrightarrow u(t_{i+1}) = u(t_i) + h \cdot u'(t_i) = u(t_i) + h \cdot f(t_i, u(t_i))$$

Matlabdelen

```
r = 3;
A = pi*r^2;
```

Variabler används för att lagra värden i

`r:` 3 `A:` 28.2743

Tilldelning: Placera värdet som står till höger i variabeln som står till vänster.

Först beräknas värdet till höger, sedan placeras svaret i `s`. Värdet som redan fanns i `s` skrivs över.

```
s = 0;          % 0
s = s + 1^2;    % 0 + 1
s = s + 2^2;    % 1 + 4
s = s + 3^2;    % 5 + 9
```

`s` får värdena 0, 1, 5 och 14 i sekvensen ovan.

Raderna ovan kan skrivas med en **`for`-loop**:

```
s = 0;
for i = 1:3
   s = s + i^2;
end
```

Raden mellan `for` och `end` upprepas tre gånger.
Första gången är `i` = 1, andra gången 2 och tredje gången 3.
`s` får successivt värdena 1, 5 och 14 i loopen.

Se mönstret: $s = \sum_{i=1}^{3} i^2$

```
s = 0;
for i = 1:3
   s = s + i^2;
end
```

I lab 2 ska man räkna summan $s = \sum_{i=0}^{999} \frac{(-1)^i}{2i+1}$.

Med **`if`-sats** kan man villkora vilka rader som ska köras.

```
r = input('Radien:  ');
if r > 0
   A = pi*r^2;
else
   A = 0;
end
```

Användaren skriver ett värde, `input` placerar värdet i `r`.
Räkna ut arean `A` bara om $r > 0$
Bara en av raderna (`A = ..`) kommer att köras, aldrig bägge.

Villkor är uttryck som har värdet sant/falskt. I Matlab bygger man upp dem med hjälp av operatorerna `>, <, >=, <=, ==, ~=` (de matematiska motsvarigheterna är $>, <, \geq, \leq, =, \neq$)

Upprepa så länge som ett villkor är sant med **`while`-sats**

```
s = -1;
while s < 0
   s = input('Skriv ett positivt tal:  ');
end
```

Så länge som `s`< 0 upprepas inläsningen. När användaren skriver ett positivt tal bryts loopen.

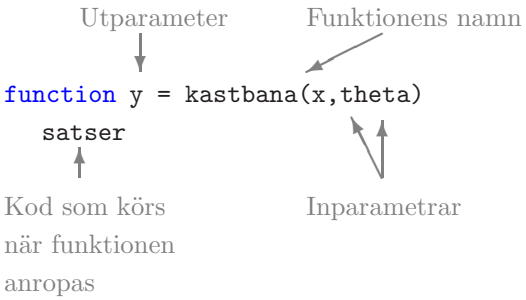
Låt $s = \sum_{i=1}^{n} \frac{1}{i}$. Hur många termer behövs för att summan ska bli ≥ 5?

```
s = 0; n = 0;
while s < 5
   n = n + 1; s = s + 1/n;
end
svar = n
```

Summera term för term till `s`. Sluta när `s`≥ 5.

I Matlab kan man definera egna funktioner på två sätt.

Externa funktioner:

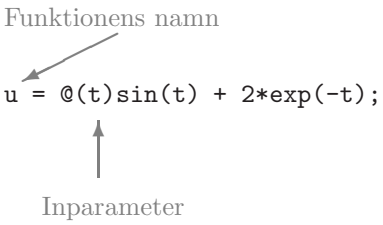


En extern funktion placeras i en egen fil, eller sist i ett skript.

Ni har sett exempel på externa funktioner i förra läsperioden.

I Lab 2 ska man skriva en extern funktion som ska heta `polylen` och som ska beräkna längd på polygontåg.

Anonyma funktioner:



Består bara av en rad.

Placeras i skriptet där de används.