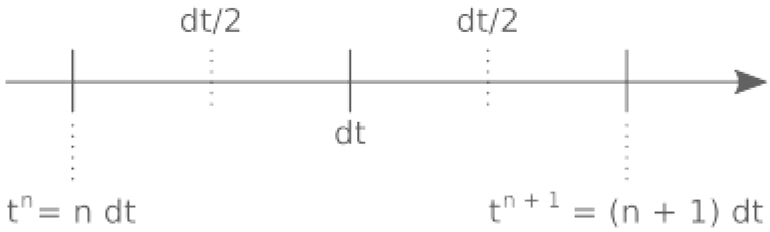
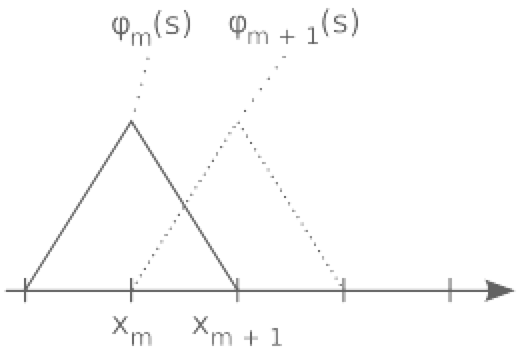
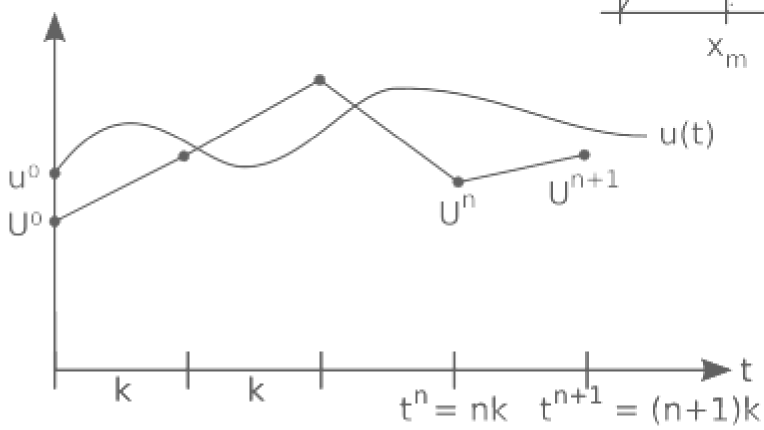
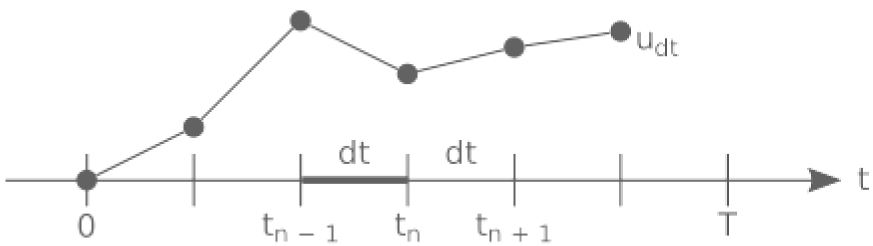
DN1240
Numeriska metoder, grundkurs 2 (F/CL)



$$u(T) = \int_0^t f(t)\,dt \approx \sum_{n=0}^{N} f(nk)k$$







$$\int_a^b u'\varphi'_j\,dx = \sum_{i=1}^{J} \int_a^b u_i \varphi'_i \varphi'_j\,dx, \quad j = 1, 2, \ldots, J$$

Version: 0
Sammanfattare: Mattias Andrée
Kontaktadress: maandree@kth.se

Alla moduler

2010−(10)okt−25: dag 1

Numeriska metoder F/CL 2010 DN1240

Lärare i kursen:

Ninne Carlsund
Katarina Gustavsson
Johan Hoffman
Johan Jansson
Matthias Sandberg
Jeannette Hiromi Spühler
Rodrigo Vilela de Abreu

Gemensam e-mail-adress för kursen (denna omgång):

numfcl-2010@csc.kth.se

Vi kommer använda Python 2.6 (status quo ante) som är installerat på NADA:s datorer (färgsalarna).

Numeriska metoder?

Computational Science/Beräkningsvetenskap

Scientific Computing/Vetenskapliga beräkningar

High Preformance Computing

Numerisk analys

Simuleringsteknik

Beräkningsmatematik

Matematiska modeller + dator

Modellering och simulering

Bygg en matematisk modell (ekvation) av verkligheten

En ekvation från verkligheten är ofta alltför komplex för att lösa med penna och papper

Simulera verkligheten (lös ekvationen):
numerisk metod + dator

Datorspel och film

Upplevelsen är ofta viktigare än realismen

Fysikmotor

Bräkningsfel är okej.

Klimatmodeller

Politiska beslut baseras på simuleringar

Vilken modell?

Noggrannhet?

Denna kurs

Matematiska modeller

Numeriska metoder

Implementering

Simulering

Uppskatta fel i simuleringen

Differentialekvationer

Beskriver kontinuum/fält/funktioner: temperatur, hastighet, tryck, densitet, &c.

Finita element metoden: approximera lösningen med enkla funktioner på en triangulering.

Eulers ekvationer (1755)

$\partial u / \partial t + u \cdot \nabla u + \nabla p = f$ Strömningsmekanik
$\nabla \cdot u = 0$

Maxwells ekvationer (1873)

$\partial B/\partial t + \nabla \times E = 0$ Elektromagnetism
$\partial D/\partial t + \nabla \times H = 0$
$\nabla \cdot D = 0$, $\nabla \cdot B = 0$

Schrödingers ekvation (1925)

$i\partial\psi/\partial t = H\psi$ Kvantmekanik

Simulering är ofta nödvändigt

- För små skalor för experiment
- För stora skalor för experiment
- Oetiska experiment
- Dyra experiment
- Simulera framtiden
- Simulera virtuell verklighet

…

5-dygnsprognos

- Samla in data från väderstationer
- Simulera vädret med Eulers ekvationer

Varför inte 10-dygnsprognos?

Diskussionsforum på BILDA.

Kolla på kurshemsidan!!!

http://www.csc.kth.se/utbildning/kth/kurser/DN1240/numfcl10/

Kursmaterial på 6 moduler.

Laborationstillfällen med bonuspoäng till tentamen
i slutet på varje modul.

Material för Python-programmering.

Schema

E-bok

Moduler:

Modul 1:	Fundamental satsen
Modul 2:	Funktionsapproximation
Modul 3:	ODE 1 (explicita/implicita metoder)
Modul 4:	ODE 2
Modul 5:	Fixpunktiteration
Modul 6:	PDE

Projekt — 3 delar

Supplementär kurs för F: DN1242

Grundexamination
Projektexamination

Vi ska få en enkel partikel, en boll, att studsa.

Vi behöver:

- Gravitation
- Partikel:
  - Koordinat $x \in [x0; xN]$
  - Massa M
  - Radie
  - Elastisitet
- Tid $t \in [t0 = 0; \infty[$

Diskreta tidssteg, dt

$$tn + 1 = tn + dt$$

tidskoordinat:	$tn = n\, dt$	
koordinat:	$xn = x(n\, dt)$	(xn är x med index n, alltså xn) x av n·dt, n:e tidssteget
hastighet:	$vn = v(n\, dt)$	v av n·dt
acceleration:	$an = a(n\, dt)$	a av n·dt
kraft:	$Fn = F(n\, dt)$	F av n·dt
massa:	M	

Newtons 2:a lag:

$$F = Ma$$

Rörelselagar:

- $v = \frac{dx}{dt}$
- $a = \frac{dv}{dt}$

Diskretiserade förändingar:

dt = tn + 1 – tn

dxn = xn + 1 – xn

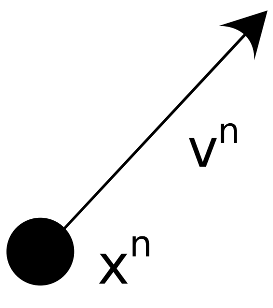
dvn = vn + 1 – vn

(d = Δ ; d()n = Δn())

Newtonsrörelselagar (för M = 1):

vn + 1 = vn + Fn dt

xn + 1 = xn + vn dt

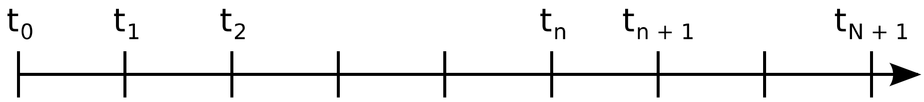


2010−(10)okt−28: dag 2

Differentialekvation

$$\begin{cases} \dot{u}(t) = f(t) \\ u(0) = u^0 \end{cases}$$

- tid t, tidssteg dt



$t_n = n\ dt$

$u^n = u(n\ dt)$

$f^n = f(n\ dt)$

- tidsstegning

$$u^{n+1} = u^n + f^n\ dt \qquad \left(\frac{u^{n+1} - u^n}{dt} = f^n\right)$$

$$u^n = u^{n-1} + f^{n-1}\ dt$$

$u^{n-1} = u^{n-2} + f^{n-2}\ dt$

$u^{n+1} = u^0 + (f^0 + f^1 + f^2 + \ldots + f^n)\ dt$

$$u^{n+1} = u^0 + \sum_{i=0}^{n} f^i\ dt$$

$$u^{n+1} - u^0 = \sum_{i=0}^{n} f^i\ dt$$

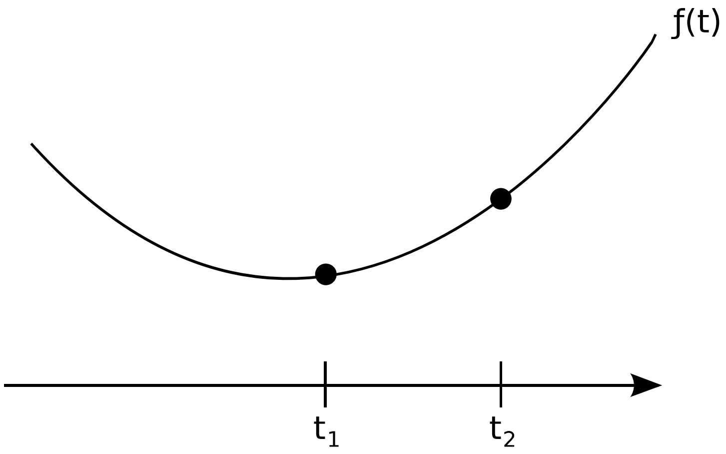
Blir integral då $dt \to 0^+$

$$\left(u(T) - u(0) = \int_0^T f(t)\ dt\right)$$

Lipschitz-kontinuietet

$f(t)$ är Lipschitz-kontinuerlig om

$$|f(t_2)-f(t_1)| \le L|t_2-t_1|$$



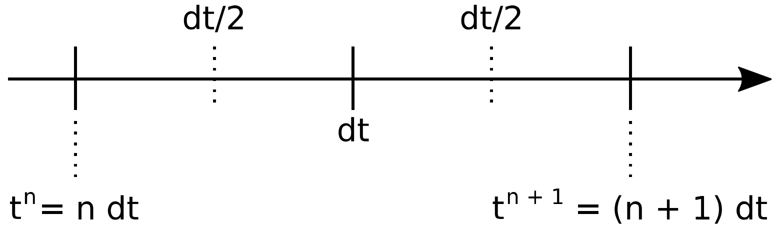
$$\left|\frac{f(t_2)-f(t_1)}{t_2-t_1}\right| \le L\,, \qquad f(t) = t^2$$

$$f(t_2)-f(t_1)=t_2^{\,2}-t_1^{\,2}=(t_2-t_1)\underbrace{(t_2+t_1)}_{L=\max}=\underbrace{f'(\xi)}_{\mathbf{L=max}}(t_2-t_1), \qquad \xi \in [t_1;\, t_2]$$

$$L=\max_{t_1 \le \xi \le t_2} f'(\xi)$$

$f(x) – f(y) = f'(\xi)(x – y)$, $\xi \in [x; y]$

Medelvärdessatsen!



Effekt av tidsstegets längd för ett steg

$$u\big((n+1)dt\big)=u(n\,dt)+f(n\,dt)\;dt$$

$$\bar{u}((n+1)dt) = u(n\,dt) + f(n\,dt)\,\frac{dt}{2} + f\left(n\,dt + \frac{dt}{2}\right)\frac{dt}{2}$$

$$u((n+1)dt) - \bar{u}((n+1)dt) =$$

$$= u(n\,dt) + f(n\,dt)\,dt - \left(u(n\,dt) + f(n\,dt)\,\frac{dt}{2} + f\left(n\,dt + \frac{dt}{2}\right)\frac{dt}{2}\right) =$$

$$= f(n\,dt)\,dt - f(n\,dt)\,\frac{dt}{2} - f\left(n\,dt + \frac{dt}{2}\right)\frac{dt}{2} =$$

$$= f(n\,dt)\,\frac{dt}{2} - f\left(n\,dt + \frac{dt}{2}\right)\frac{dt}{2} =$$

$$= \left[f(n\,dt) - f\left(n\,dt + \frac{dt}{2}\right)\right]\frac{dt}{2}$$

$$\left|f(t_2) - f(t_1)\right| \le L\left|t_2 - t_1\right|$$

där ƒ(t) är Lipschitz-kontinuerlig.

$$u((n+1)dt) - \bar{u}((n+1)dt) =$$

$$= \left(f(n\,dt) - f\left(n\,dt + \frac{dt}{2}\right)\right)\frac{dt}{2} \le$$

$$\le L\left|n\,dt - \left(n\,dt + \frac{dt}{2}\right)\right|\frac{dt}{2} =$$

$$= L\left|-\frac{dt}{2}\right|\frac{dt}{2} =$$

$$= \frac{L}{4}\,dt^2$$

Skillnad över [0; T] $T = (N + 1)\,dt$

$$\left|u(T) - \bar{u}(T)\right| \le \frac{L}{4}\,dt^2 \times (N+1) = \frac{LT}{4}\,dt$$

❑ Samma sak för $\bar{u}$ motsvarande $\frac{dt}{4}$

— Låt $u_{dt}(T)$ motsvara lösning med tidssteg dt

$$|u_{dt}(T)-u_{dt/4}(T)|=$$

$$=|u_{dt}(T)-\underbrace{u_{dt/2}(T)+u_{dt/2}(T)}_{=0,\text{ Används för uppdelning}}-u_{dt/4}(T)|\leq$$

$$\leq|u_{dt}(T)-u_{dt/2}|+|u_{dt/2}(T)-u_{dt/4}|\leq$$

$$\leq\frac{LT}{4}dt+\frac{LT}{4}\frac{dt}{2}$$

❑ Upprepa för $\frac{dt}{8}$, $\frac{dt}{16}$, ...

❑ Låt $\bar{u}$ vara lösning med godtyckligt litet dt

$$|u_{dt}(T)-\bar{u}(T)|\leq$$

$$\leq\frac{LT}{4}dt+\frac{LT}{4}\frac{dt}{2}+\frac{LT}{4}\frac{dt}{4}+\frac{LT}{4}\frac{dt}{8}+\ldots=$$

$$\leq\frac{LT}{2}\left(\frac{1}{2}+\frac{1}{4}+\frac{1}{8}+\frac{1}{16}\right)<$$

$$<\left\{\sum_{i=1}^{N}\frac{1}{2^n}\cong 1\right\}<\frac{LT}{2}$$

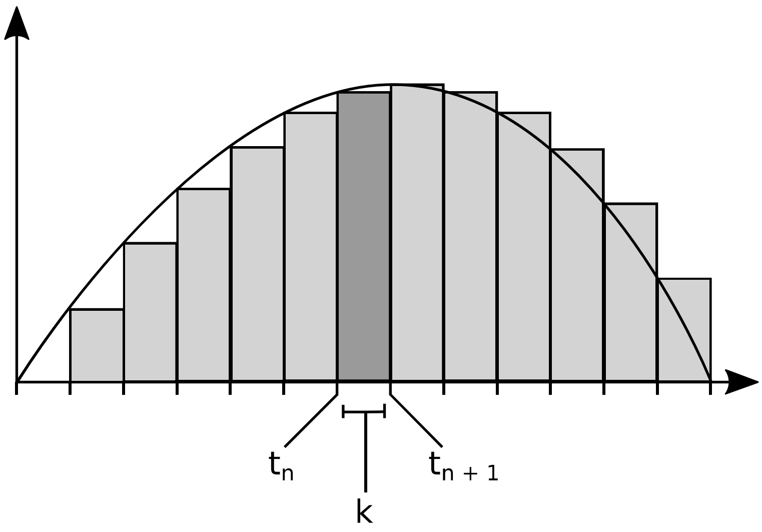
Fundamentalsatsen

Om $f:[0;T]\rightarrow\mathbb{R}$ är Lipschitz-kontinuerlig så är funktionen $u(t)=\int_0^t f(s)\,ds$

(som definieras genom tidstegning med försvinnande kort tidssteg,

motsvarande $\bar{u}$) lösning till ekvationen

$$\begin{cases}\dot{u}(t)=f(t)\\ u(0)=u^0\end{cases}\quad\text{för } t\in[0;T]$$



Riemann-summa

$$u(T)=\int_0^t f(t)\,dt \approx \sum_{n=0}^{N} f(nk)k$$

2010−(10)okt−29: dag 3

Övning 1

I slutet av kursen ska vi göra beräkningar av viktiga problem.

Vi ska titta på en metod med tidsteg.

Kommer ta fram en approximativ lösning.

Konceptet: tidstegning

$$\frac{du}{dt} = f(t;\, u) \qquad \text{given funktion} = f(t;\, u)$$

Enklare $f = f(t)$ enbart.

Exempel: Newtons andra lag

$$\left.\begin{array}{ll} \dfrac{dv}{dt} = F & f(t) = F \\ \dfrac{dx}{dt} = v & f(t) = v \end{array}\right\} \text{Samma form som } \frac{du}{dt} = f(t)$$

Idén: $dv = F \cdot dt$

$dv = v^{n+1} - v^{n}$

$dt = t_{n+1} + f^{n}$

Differenskvot:

$$v' = \lim_{dt \to 0} \frac{dv}{dt}$$

v^n är hastigheten vid ett tillfälle.
dt skrivs ofta k, dt = k är ett tidssteg.

$v^{n+1} - v^n = F \cdot dt$

$v^{n+1} = v^n + dt \cdot F(t^n)$, $n \in \mathbb{N}$ Kallas Euler framåt

För att inte få oändligt med lösningar måste vi fixera u vid en punkt.

$t = 0$ $v = 0$

$n = 0 \Rightarrow v^1 = v^0 + dt\, F(t_0) \Rightarrow v' = 0 + dt\, F(t_0)$

$t_1 = t_0 + dt$

$t = 1$

$v^2 - v^2 \cdot dt\, t_1 = dt\, F(t_0) + dt\, F(t_1)$

$t_2 - t_1 + dt$

För den generella formen $\frac{du}{dt} = f(t)$

Euler framåt

$u^{n+1} = u^n + dt\, f(t_n)$, $n \in \mathbb{N}$

Trapetsmetoden

$$u^{n+1} = u^n + \frac{dt}{2}\left(f(t_n) + f(t_{n+1})\right), \quad n \in \mathbb{N}$$

Om man använder sig av Euler kommer man få ett fel som väger som Δt.

I andra fallet [Trapetsmetoden] så växer felet kvadratisk,
vilket kallas för ordning två.

Uppgift 1

En given kraft $f(t) = \cos(t)$ verkar på en partikel med massan, m = 1.
Rörelsen hos partikeln kan beskrivas med Newtons andra lag enligt

$$m \frac{dv}{dt} = \cos(t)$$

$$\frac{dx}{dt} = v(t)$$

Vid tiden t = 0 har partikeln hastigheten v = 0 och befinner sig vid x = 0.
Vilken hastighet har partikeln och var befinnr den sig vid t = 0,2?

Om kraften istället ges av $f(t)$ sin(t) / t,
vilken hastighet och position kommer partikeln ha vid t = 1,2?

I det här fallet vet vi att den vid t = 1 har hastigheten v = 0 och positionen x = 0.

$v^{n+1} = v^n + dt \cos(t_0)$

$x^{n+1} = x + dt\, v^n$

Man börjar med att välja ett tidssteg: dt = 0,2

t = 0, x = 0, v = 0

n = 0 $v^{0+1} = v^0 + 0{,}2 \cos(t^0) = 0 + 0{,}2 \cos 0 = 0{,}2$

$x^1 = x^0 + dt\, v^0 = 0$

$$0 = 0 + \underbrace{0{,}2 \cdot 0}_{\text{skumt}}$$

Fel! För stort steg!

Analytisk lösning

$$m\frac{dv}{dt}=\cos(t),\quad m=1,\ t=0,\ v=0,\ x=0$$

$$\frac{dx}{dt}=v$$

$$\frac{dv}{dt}=\cos(t) \qquad \begin{array}{l} v(t)=\sin(t)+C_1 \\ v(0)=\sin(0)+C_1=0 \Rightarrow C_1=0 \end{array}$$

$$\frac{dx}{dt}=\sin(t) \qquad \begin{array}{l} x(t)=-\cos(t)+C_2 \\ x(0)=-\cos(0)+C_2=0 \Rightarrow C_2=1 \end{array}$$

$x(t) = 1 - \cos(t)$

$v(0,2) = \sin(0,2) \approx 0,19867$

$x(0,2) = 1 - \cos(0,2) \approx 0,019933$

Igen med ett mindre steg, $dt = 0,1$ (2 steg)

$$v_{0,1}^2=0,1995$$

$$x_{0,1}^2=0,01$$

$$\left.\begin{array}{l} |v_{ex}(0,2)-v^1|\approx 0,00133 \\ |x_{ex}(0,2)-x^1|\approx 0,0199 \end{array}\right\} dt=0,1$$

$$|v_{ex}(0,2)-v^2|\approx 0,00083$$

$$|x_{ex}(0,2)-x^2|\approx 0,00933$$

```
t = 0
x = 0
v = 0
dt = 0,5
tslut = 0,2

varray = array(v)
xarray = array(x)
tarray = array(t)

N = (tslut - t) / dt

print N

for i in range(0, N):
     t0 = t
     x0 = x
     v0 = v

     t = t0 + dt
     v = timestep(f, t0, v0, dt, "Euler")
     x = x0 + dt * v0

     varray = append(varray, v)
     xarray = append(varray, x)
     tarray = append(varray, t)

err_v = v_ex(t) - varray[-1]
err_x = x_ex(t) - xarray[-1]

print
print
```

0,5 ger felen

$|v_{ex}(0,2) - v^4| \approx 0,00046$
$|x_{ex}(0,2) - x^4| \approx 0,0056$ dt = 0,05

Med midpoint

dt = 0,2

$err_r = 6{,}63 \cdot 10^{-4}$

dt = 0,1

$err_r = 1{,}66 \cdot 10^{-4}$

Halvering av dt gör att felet blir en fjärdedel av vad det var innan.

Fundamentalsatsen

Relationen mellan derivata och integral

$\frac{du}{dt} = f(t), \ u(0) = 0$ u(t) är en primitiv funktion till $f(s)$

$u(t) = \int_0^t f(s)\,ds$ är lösningen till

$$u(T) - \bar{u}(T) \leq \frac{LT}{4}\,dt$$

Approximativ lösning med dt, u(T)

Approximativ lösning med $\frac{dt}{2}, \ \bar{u}(T)$

$$L = \text{“Lipschitz-konstanten”} = \max_{t_0 \leq t \leq T} \left| \frac{df}{dt} \right|$$

2010–(11)nov–01: dag 4

$f(t)$ är Lipschitz-kontinuerlig på intervallet $[t_1; t_2]$ om

$$|f(s)-f(t)| \leq L|s-t| \quad \forall \quad s, t \in [t_1; t_2]$$

Fundamentalsatsen:

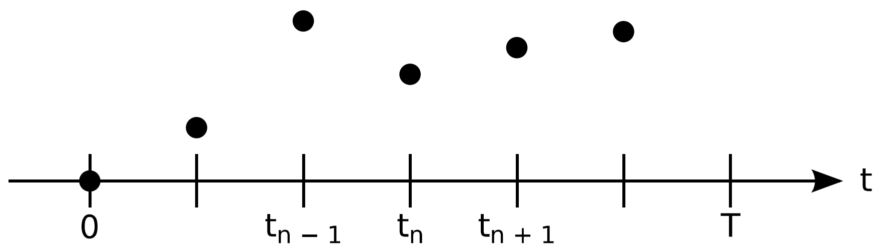
Om f är Lipschitz-kontinuerlig $[0; t]$ så är

$$u(t) = \int_0^t f(s)\, ds$$ en lösning till differentialekvationen

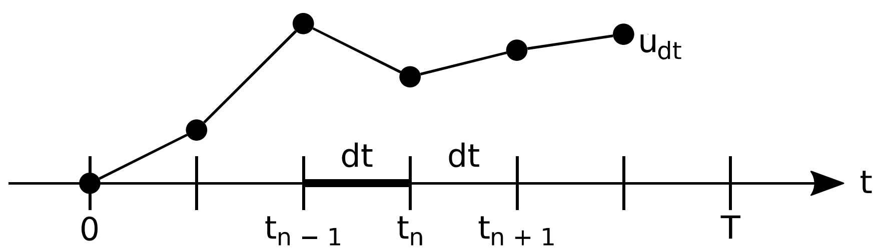
$$\begin{cases} \dot{u}(t) = f(t) \\ u(0) = 0 \end{cases} \quad t \in [0; T]$$

$$u(t) = \int_0^t f(s)\, ds$$ definieras genom tidsstegning

$u^{n+1} = u^n + f(t_n)\, dt$ (1) med försvinnande litet tidssteg, dt



Styckvis linjär interpolation



För $[t_n; t_{n+1}]$:

$$u_{dt}(t)=u^n\frac{t_{n+1}-t}{dt}+u^{n+1}\frac{t-t_n}{dt}$$ u_{dt} är en approximation av u med tidssteget dt

Residual:

$\dot{u}(t)-f(t)=0$ för exakt lösning

$\dot{u}_{dt}(t)-f(t)\quad 0$ alltså inte strikt lika med noll (detta brukar skrivas ≠ 0)

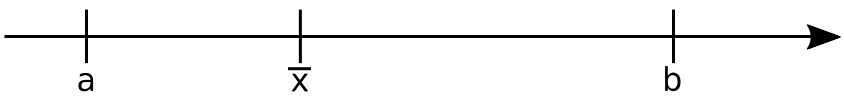
För $[t_n; t_{n+1}]$:

$$\left|\dot{u}_{dt}(t)-f(t)\right| = \left|\frac{u^{n+1}-u^n}{dt}-f(t)\right| = \{(1)\} = \left|f(t_n)-f(t)\right| \le L\,dt$$

Fourier: $u(t)\approx\sum_{m=-N}^{N}u_m e^{imx}$ (Se modul 3 för SF1637)

Taylorserie: $u(x)\approx u(\bar{u})+u'(\bar{x})(x-\bar{x})+\frac{1}{2}u''(\xi)(x-\bar{x})^2$

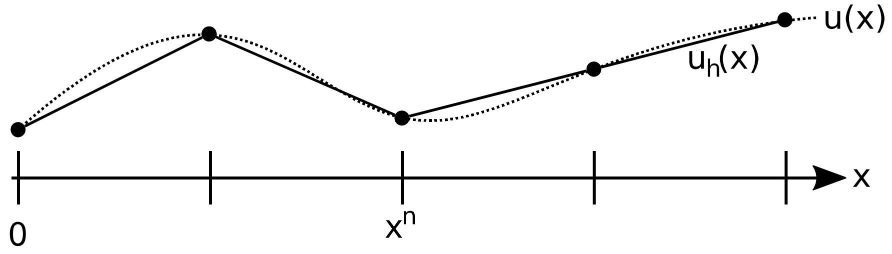
$\xi\in[a; b]$



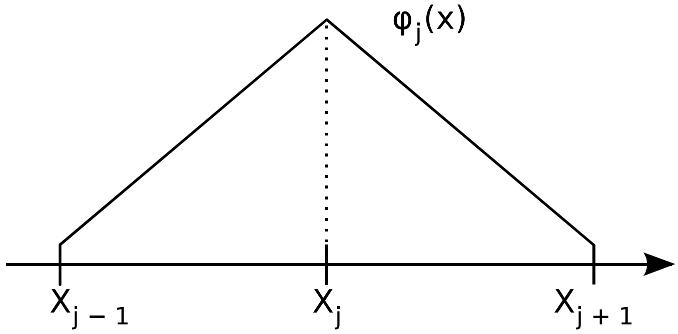
(Vi har kvar ½ trots att vi har ett “ordo”-värde,
eftersom vi vill beräkna ett felvärde.)

$\pi_1\, u(x_n) = u_h(x_n) = u(x_n)$

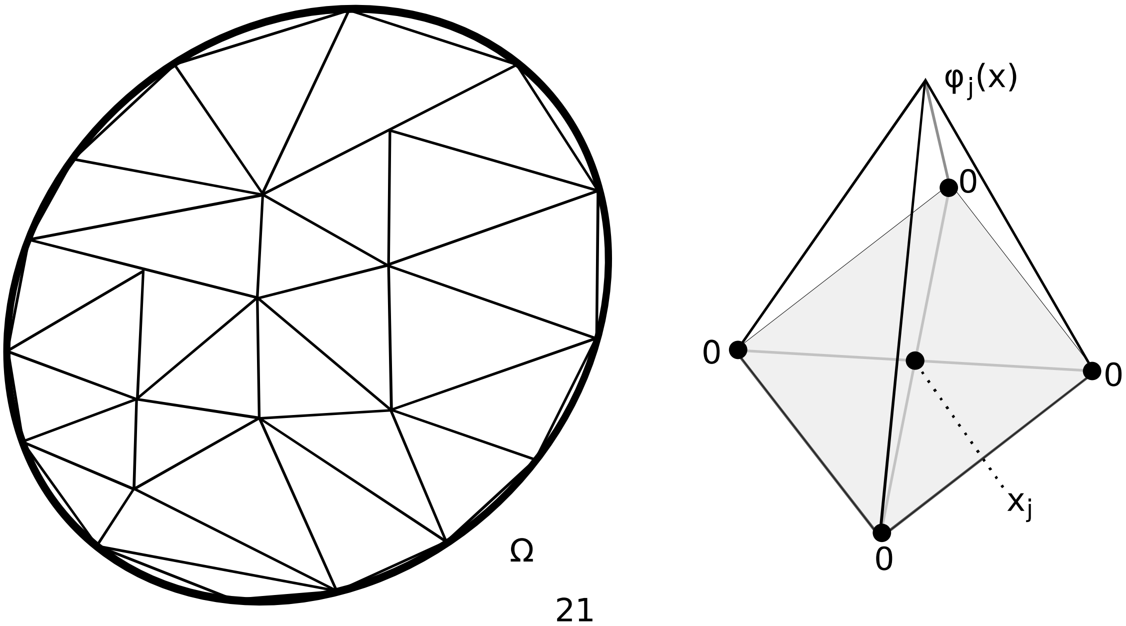
π_1 betyder linjär interpolans.

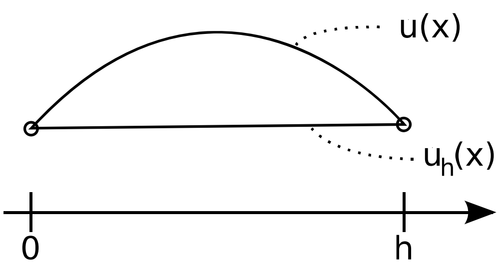


$$u_h(x) = \sum_{j=0}^{N} u(x_j)\varphi_j(x), \qquad \varphi_j(x_k) = \begin{cases} 1, & j = k \\ 0, & j \neq k \end{cases}$$



3-dimensionell interpolation (över 2D) med hjälp av triangulering med polygonpyramider (läraren sa tetraedrar (en. tetrahedrons) trots att han hade en rektangelpyramid):





$$\begin{cases} u_h(0)=u(0) \\ u_h(h)=u(h) \end{cases}$$

Antag att u(x) är två gånger deriverbar.

Uppskatta $e_n = u(x) - u_h(x)$, $x \in [0; n]$.

1. Subtrahera linjär funktion från u(x) så att $u(0) = u(h) = 0 \Rightarrow u_h \equiv 0$

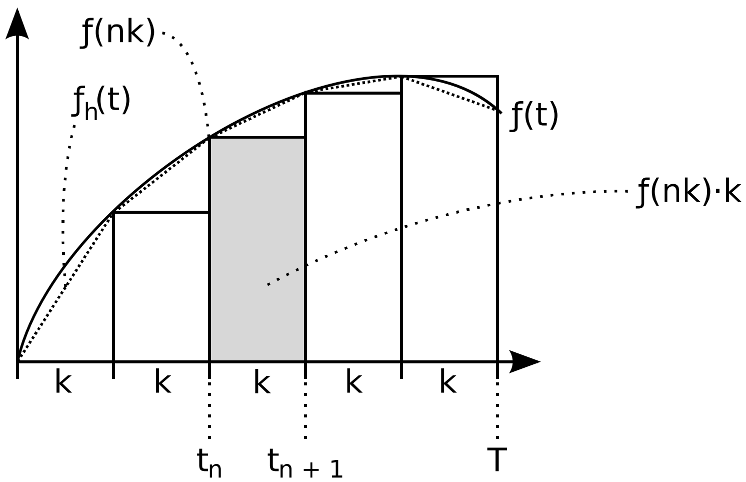
2. Antag att $u(x) \neq u_h(x)$ för något $x \in]0; h[$.

   u(x) är maximal då $x = \xi \in]0; h[\Rightarrow u'(\xi) = 0$

$$|u(x)-u_h(x)| \leqslant \max_{x\in]0;h[} |u(x)-u(\xi)| = \max_{x\in]0;h[} |u(x)-u(\xi)-\underbrace{u'(\xi)(x-\xi)}_{0}| =$$

$$= \{\text{Taylor}\} = \{\eta \in]0;h[\} = \max_{x\in]0;h[} \left|\frac{1}{2}u''(\eta)(x-\xi)^2\right| \leqslant \underbrace{\frac{1}{2}\max_{x\in]0;h[}|u''(\eta)|}_{C} h^2$$

Kvadratur : Riemannsumma



f_h är en styckvis linjärt interpolant; $f_h(t_i) = f(t_i)$, $i \in \mathbb{Z}_{0..N}$

Kvadraturfel

$$\left| \int_0^T f(t)\,dt - \sum_{n=0}^{N} f(nk)\cdot k \right| \leq \frac{LT}{2} k$$

k utan exponent betyder att konvergensordningen är 1.

$$\int_0^T f(t)\,dt \simeq \sum_{n=0}^{N} \frac{1}{2}\big(f(nk)+f((n+1)k)\big)\cdot k = \int_0^T f_h(t)\,dt$$

Kvadraturregler kan bygga på exakt integraton av en interpolant.

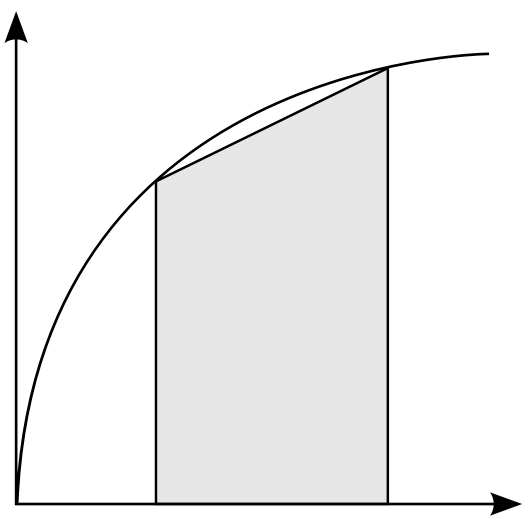
Kvadraturfel:

$$\left| \int_0^T f(t)\,dt - \int_0^T f_h(t)\,dt \right| = \left| \int_0^T \big(f(t)-f_h(t)\big)\,dt \right| \leq \int_0^T \left|f(t)-f_h(t)\right| dt \leq$$

$$\leq \max_{t\in[0;T]} \left|f(t)-f_h(t)\right| \cdot \int_0^T dt \leq CT k^2$$

Trapetsregeln (en. Trapezoidal/Trapezoid/Trapezium rule):

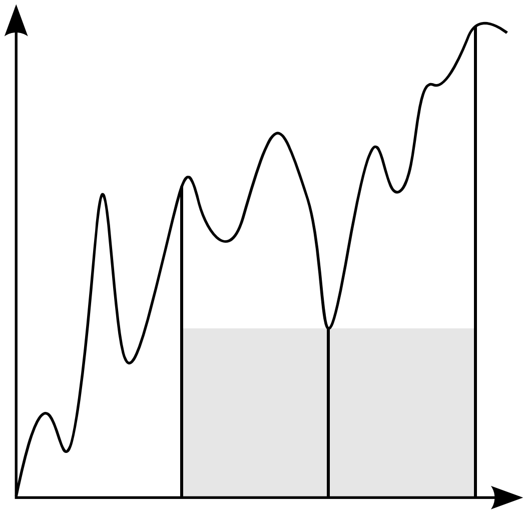
(universialt namn) (US) (UK)



Summering av trapetser.

$$\int_0^T f(t)\,dt \cong \sum_{n=0}^{N} \frac{f(t_n)+f(t_n+k)}{2}\cdot k$$

Mittpunktsregeln (en. Midpoint rule):



$$\int_0^T f(t)\,dt \cong \sum_{n=0}^{N} f\left(t_n + \frac{k}{2}\right) \cdot k$$

Samma konvergensordning
som trapetsregeln

L_2-projektion

$$u_h(x) = \sum_{j=1}^{N} u_j \varphi_j(x)$$

Definition:

L_2-projektionen, $\mathbb{P}u$ (eller Pu), av u på intervallet [0; h] som (i detta exempel) styckvis konstant så att

$$\int_0^h (u(y) - \mathbb{P}u)\, v\, dy = 0$$

för alla konstanta funktioner, v, på intervallet [0; h].

$$v\left(\int_0^h u(y)\,dy - \int_0^h \mathbb{P}u\,dy\right) = 0$$

$$\Updownarrow$$

$$v\left(\int_0^h u(y)\,dy - \mathbb{P}u \cdot h\right) = 0$$

$$\Updownarrow$$

$$\Updownarrow$$

$$\int_0^h u(y)\,dy - \mathbb{P}u \cdot h = 0$$

$$\Updownarrow$$

$$\mathbb{P}u = \frac{1}{h}\int_0^h u(y)\,dy$$

Medelvärdet av u på [0; h]

Linjär algebra

$$\vec{v} = (v_1;\ v_2)$$

$$\|\vec{v}\| = \sqrt{{v_1}^2 + {v_2}^2}$$

L_2-norm:

$$\|f\| = \left(\int_0^h f(y)^2\,dy\right)^{\frac{1}{2}}$$

$$\|u - \mathbb{P}u\|^2 = \int_0^h \left(u(y) - \mathbb{P}u\right)^2 dy = \{\text{konstant funktion, } v\} =$$

$$= \int_0^h \left(u(y) - \mathbb{P}u\right)\left(u(y) - \mathbb{P}u\right) dy + \underbrace{\int_0^h \left(u(y) - \mathbb{P}u\right)\underbrace{\left(\mathbb{P}u - v\right)}_{\text{konstant}} dy}_{0} =$$

$$= \int_0^h \left(u(y) - \mathbb{P}u\right)\left(u(y) - v\right) dy \le \left\{\begin{matrix}\text{\{Linjär algebra:} \\ \text{Cauchys olikhet:} \\ v \cdot w \le |v| \cdot |w|\end{matrix}\right\} \le$$

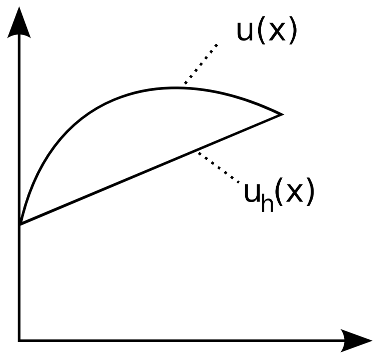
$$\le \left(\int_0^h \left(u(y) - \mathbb{P}u\right)^2 dy\right)^{\frac{1}{2}} \cdot \left(\int_0^h \left(u(y) - v\right)^2 dy\right)^{\frac{1}{2}} =$$

$$= \|u - \mathbb{P}u\| \cdot \|u - v\| \Rightarrow$$

$$\Rightarrow \|u - \mathbb{P}u\| \le \|u - v\| \quad \forall\, v$$

2010–(11)nov–03: dag 5

Styckvis linjär interpolation



$$|u(x)-u_h(x)| \leq C \cdot C_0 \cdot h^2,$$

$$C = \max_{x \in [0;h]} |u''(x)|, \quad C_0 = \frac{1}{8}$$

(För första steget, vid nästa steg måste gränserna för x ändras)

Felkontroll:

Garantera att $|u(x)-u_h(x)| < TOL =$ "tolerans"

N stycken intervall

För varje intervall

max felbidrag: $\frac{TOL}{N}$

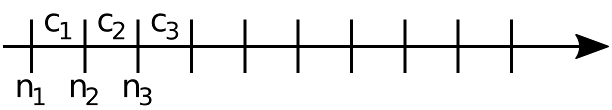
Detta är uppfyllt om $C \cdot C_0 \cdot h^2 < \frac{TOL}{N}$ för varje intervall.

1) Lägg till flera intervall (och noder) — h-adaptivitet

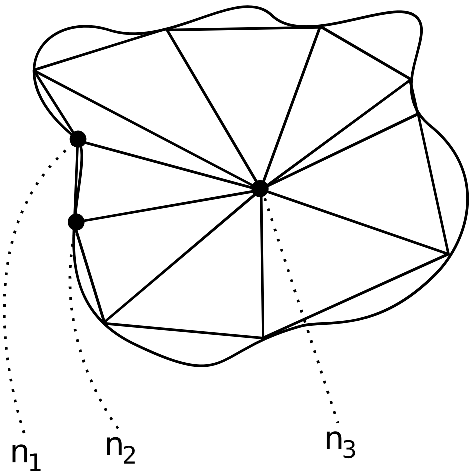
2) Öka approximationsordning — p-adaptivitet

3) Flytta noder — r-adaptivitet

Mesh



ID: $\quad n = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \end{bmatrix} \qquad c = \begin{bmatrix} n_1 & n_2 \\ n_2 & n_3 \\ \vdots & \vdots \end{bmatrix}$



ID: $\quad n = \begin{bmatrix} x_1 & x_2 \\ x_2 & x_3 \\ \vdots & \vdots \end{bmatrix} \qquad c = \begin{bmatrix} n_1 & n_2 & n_3 \\ \vdots & \vdots & \vdots \end{bmatrix}$

$$\int_\Omega t(x)\,dx \approx \sum_{C\in\tau}\sum_{n\in C}\frac{1}{3}f(x_n)|C| \quad ,\ \text{där}\ |C|\ \text{är arean av cellen.}$$

$$\begin{cases} \dot{u}(t) = f(t) \\ u(0) = u^0 \end{cases} \qquad t \in\ ]0;\tau]$$

Euler framåt $\qquad u((n+1)k) = u(nk) + f(nk)\cdot k$

Felet $\leq \dfrac{LT}{2}k$

Ordinär differentialekvation

$$\begin{cases} \dot{u}(t) = f(u(t)) \\ u(0) = u^0 \end{cases}$$

Framåt Euler

$u((n + 1)k) = u(nk) + f(u(nk))\cdot k$
$u((n + 1)k) = u(nk) + \dot{u}(nk)\cdot k$
$u^{n+1} = u^n + \dot{u}^n\cdot k$

(explicit Euler)

Bakåt Euler

$u((n + 1)k) = u(nk) + f(u[(n + 1)k])\cdot k$
$u((n + 1)k) = u(nk) + \dot{u}((n + 1)k)\cdot k$
$u^{n+1} = u^n + \dot{u}^{n+1}\cdot k$

(implicit Euler)

Trapetsregeln:

$$u((n+1)k) = u(nk) + \frac{k}{2}\left(f(u[nk]) + f(u[(n+1)k])\right)$$

$u((n + 1)k) = u(nk) + \tfrac{1}{2}[\dot{u}(nk) + \dot{u}((n + 1)k)]\cdot k$

$u^{n+1} = u^n + \tfrac{1}{2}(\dot{u}^n + \dot{u}^{n+1})\cdot k$

Exempel

$$\begin{cases} \dot{u}(t) = u(t) \\ u(0) = 1 \end{cases}, \quad t \in [0; 2], \quad u(t) = e^t$$

ODE

$$u(t)=\overrightarrow{\left(u_1(t);\,u_2(t)\right)}$$

$$\begin{cases} \dot{u}(t)=f(u(t)) \\ u(0)=u^0 \end{cases}$$

$$\dot{u}_1(t)=f_1(u(t))$$

$$\dot{u}_2(t)=f_2(u(t))$$

$$u_1(t)=u_1^0$$

$$u_2(t)=u_2^0$$

Exempel:

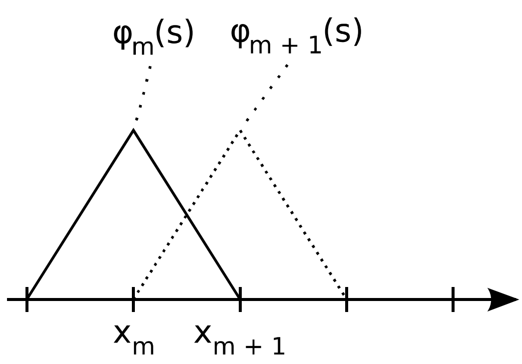
Volterra-Lotka ("predetor-prey system")

$$\dot{u}_1(t)=u_1(t)\left(\alpha-\beta u_2(t)\right)$$

$$\dot{u}_2(t)=u_2(t)\left(\gamma-\delta u_1(t)\right)$$

$$u_n(x)=\sum_{i=1}^{N} u_i\,\varphi_i(x)$$

$$\varphi_i(x_j)=\begin{cases} 1, & i=j \\ 0 \end{cases}$$



Exempel: Kemisk reaktion:

ämnen (konc.) A, B, C, D

1. $A + B \rightarrow 2B$
2. $B + C \rightarrow 2C$
3. $C \rightarrow D$

Differentialekvationsbidrag:

1. $\dot{A}(t) = -k_1 A(t) B(t)$

$\dot{B}(t) = k_1 A(t) B(t)$

2. $\dot{B}(t) = -k_2 B(t) C(t)$

$\dot{C}(t) = k_2 B(t) C(t)$

3. $\dot{C}(t) = -k_3 C(t)$

$\dot{D}(t) = k_3 C(t)$

Sätt $u(t) = \begin{bmatrix} A(t) \\ B(t) \\ C(t) \\ D(t) \end{bmatrix}$

$$\dot{u}_1(t) = -k_1 u_1(t) u_2(t)$$

$$\dot{u}_2(t) = k_1 u_1(t) u_2(t) - k_2 u_2(t) u_3(t)$$

$$\dot{u}_3(t) = k_2 u_2(t) u_3(t) - k_3 u_3(t)$$

$$\dot{u}_4(t) = k_3 u_3(t)$$

Exempel: Mass-fjäder system

M=1

x(t) — position
v(t) — hastighet

(1) $\dot{x}(t)=\dot{v}(t)$

(2) $\dot{v}(t)=F(t)=-x$

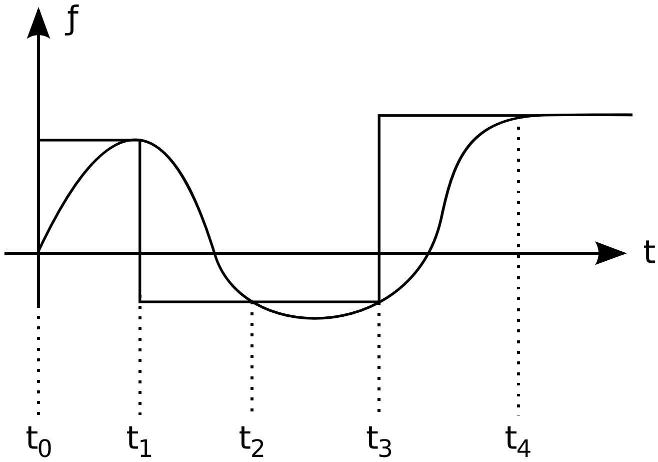
Tidsderivata av (1):

$$\ddot{x}(t)=\dot{v}(t)\underset{(2)}{=}-x$$

2010–(11)nov–04: dag 6

Idag: Styckvis konstant och styckvis linjär interpolation.
Numerisk integrering i 1D och 2D (kvadratur).
Tidsstegning (förra gången)

Idén:



Styckvis konstant interpolaton.
(enkelt att använda!)

$f \approx C_0 \cdot 1 \qquad C_0 = f(t_1)$
$t_0 \le t \le t_1$

1 · "basfunktion"

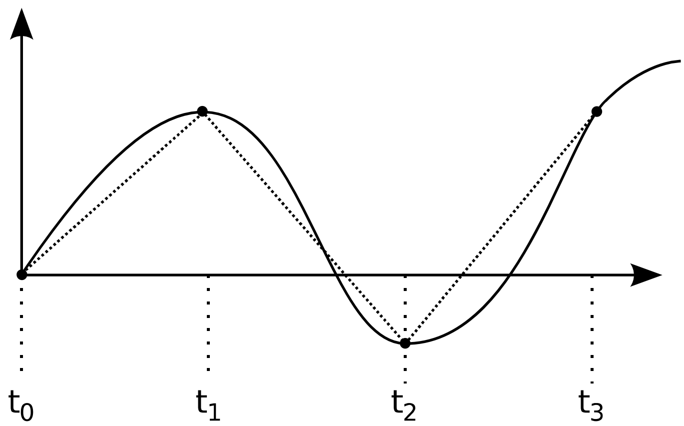
Styckvis linjär interpolation.

$f(t) \approx C_0 \cdot 1 + C_1(t - t_0), \quad t_0 \le t \le t_1$

2 basfunktioner: 1 och $(t - t_0)$

Bestäm C_0 och C_1.

$$f(t_0) = C_0 + C_1(t_0 - t_0) \Leftrightarrow C_0 = f(t_0)$$



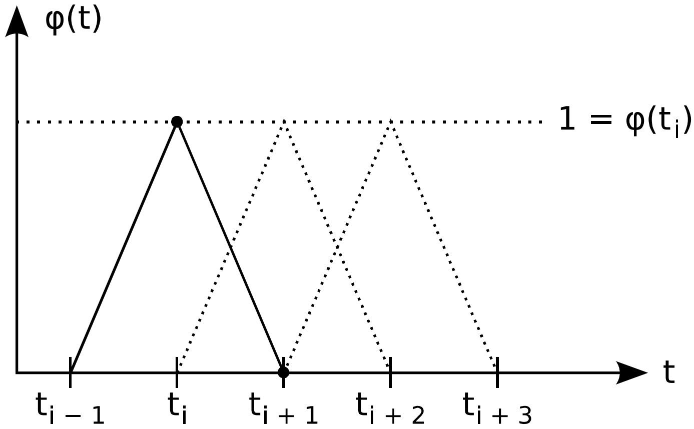
$$f(t_1)-C_0+C_1(t_1-t_0) \;\Leftrightarrow\; C_1=\frac{f(t_1)-f(t_0)}{t_1-t_0}$$

C_0 är värdet vid t_0
$C_0 + C_1(t_1 - t_0)$ är värdet vid t_1

$$f(t)=f(t_0)+\frac{f(t_1)-f(t_0)}{t_1-t_0}(t-t_0), \qquad t_0 \le t \le t_1$$

Andra typer av basfunktioner.

Hatt-funktioner



$\varphi_j(t_i) = 1$ omm $i = j$

$\varphi_j(t_i) = 0$ omm $|i - j| \ge 1$

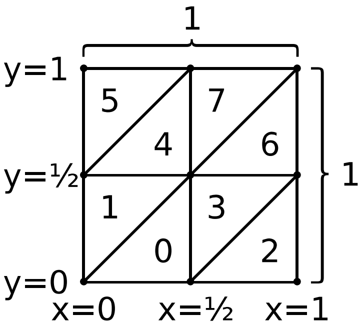
$\varphi_j(t_i) = 1 - |i - j|$ omm $|i - j| \le 1$

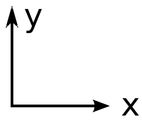
$$f(t)=\sum_{i=0}^{N} f(t_i)\cdot\varphi(t)$$

Styckvis linjär interpolation.

Styckvis linjär interpolation i 2D:

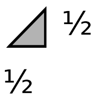
Beräkningsnät (en. mesh)





• noder (en. vertex)

 cell (triangel)



“area” $= A = \frac{1}{8} = 0{,}125$

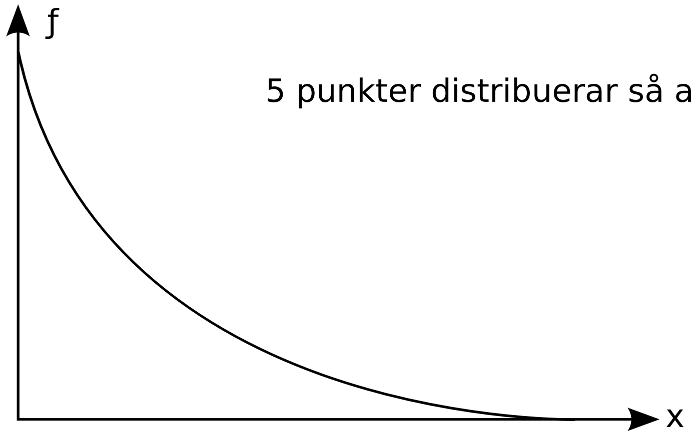
(1) $$\left|u(x) - u_h(x)\right| \leq \frac{1}{8} C \cdot h^2$$

↑ verklig funktion

↑ linjär funktion

$$C = \max_{0 \leq x \leq h} \left|u''(x)\right|$$

(2) $f(x) = e^{-10x}$, $0 \le x \le 1$



5 punkter distribuerar så att felet är mindre än 0,15.

$$C = \max_{0 \le x \le h} |f''(x)|$$

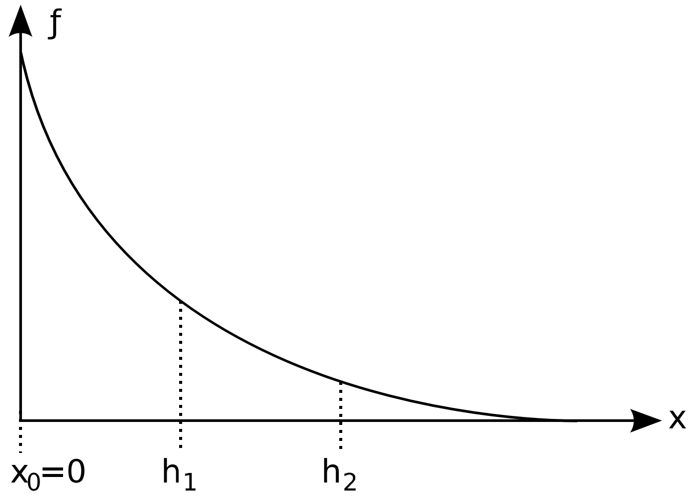
$$f''(x) = 100e^{-10x}$$

$$C = 100$$

$$\frac{1}{8} 100 \cdot h^2 < 0{,}15$$

$$\Downarrow$$

$$h < \sqrt{\frac{0{,}15 \cdot 8}{100}} = 0{,}1095$$



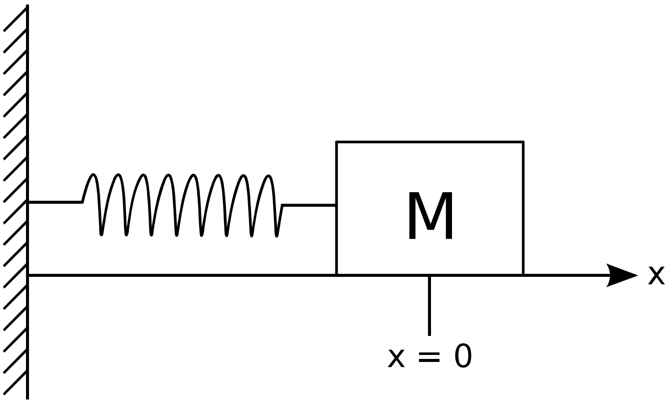
$h_1 = 0{,}1$ OK!

$$\sum_{k=1}^{8} \sum_{j=1}^{3} \frac{1}{3} f(x_i; y_i)\, v_k$$

$\uparrow$ över varje cell $\uparrow$ area

2010–(11)nov–10: dag 7

Modul 3: ODE



x(t) — position
v(t) — hastighet

$$\begin{cases} \dot{x}(t) = v(t) \\ \dot{v}(t) = -x \end{cases}$$

M = 1

$$\begin{matrix}(1)\\(2)\end{matrix}\begin{cases} x^{n+1} = x^n + kv^n \\ v^{n+1} = v^n - kx^n \end{cases} \qquad \begin{cases} x^{n+1} = x^n + kv^{n+1} \\ v^{n+1} = v^n - kx^{n+1} \end{cases}$$

(†) (‡)

(†) är en approximation med “Framåt Euler”
(‡) är en approximation med “Bakåt Euler”

Trapetsregel:

$$\begin{cases} x^{n+1} = x^n + \frac{k}{2}(v^n + v^{n+1}) \\ v^{n+1} = v^n - \frac{k}{2}(x^n + x^{n+1}) \end{cases}$$

Trapetsregeln (Trapetsmetoden) är ett medelvärde mellan “Framåt Euler” och “Bakåt Euler”.

Total Energy, E för M = 1

$$E(t) \;=\; \underset{\substack{\text{Potentiell}\\\text{energi}}}{\frac{1}{2}x^2(t)} \;+\; \underset{\substack{\text{Kinetis}\\\text{energi}}}{\frac{1}{2}v^2(t)}$$

Energin i ett system kan beskrivas med E(t) som här angiven. Energin i ett slutet system är bevarad vilket innebär att derivatan av E, eller ΔE = 0 för en bra approximation.

$$\frac{dE}{dt}=\dot{x}\,x+\dot{v}\,v=\underset{(1)}{vx}-\underset{(2)}{xv}=0$$

Alternativt:

Multiplicera (1) med x och (2) med v.
Adderar sedan resultaten.

$$\begin{cases}\dot{x}\,x=vx\\ \dot{v}\,v=-xv\end{cases} \;\Rightarrow\; \dot{x}\,x+\dot{v}\,v=vx-xv=0$$

$$\frac{d}{dt}\left(\frac{1}{2}x^2\right)+\frac{d}{dt}\left(\frac{1}{2}v^2\right)=0 \;\Rightarrow\; \frac{d}{dt}E=0$$

$$\begin{cases}x^{n+1}-x^n \;=\; \frac{k}{2}(v^n+v^{n+1})\\ v^{n+1}-v^n \;=\; -\frac{k}{2}(x^n+x^{n+1})\end{cases}$$

Multiplicera med

$$\begin{cases}x^{n+1}+x^n\\ v^{n+1}+v^n\end{cases}$$

$$\Downarrow$$

$$\begin{cases}(x^{n+1}-x^n)(x^{n+1}+x^n) \;=\; \frac{k}{2}\overbrace{(v^n+v^{n+1})(x^n+x^{n+1})}^{\alpha}\\ (v^{n+1}-v^n)(v^{n+1}+v^n) \;=\; -\frac{k}{2}\underbrace{(x^n+x^{n+1})(v^n+v^{n+1})}_{\alpha}\end{cases}$$

$$\Updownarrow$$

$$\Updownarrow$$

$$\begin{cases} (x^{n+1})^2 - (x^n)^2 = \dfrac{k}{2}\alpha \\ (v^{n+1})^2 - (v^n)^2 = -\dfrac{k}{2}\alpha \end{cases}$$

$$\Downarrow$$

$$(x^{n+1})^2 - (x^n)^2 + (v^{n+1})^2 - (v^n)^2 = \frac{k}{2}\alpha - \frac{k}{2}\alpha$$

$$\Updownarrow$$

$$(x^{n+1})^2 - (x^n)^2 + (v^{n+1})^2 - (v^n)^2 = 0$$

$$\Updownarrow$$

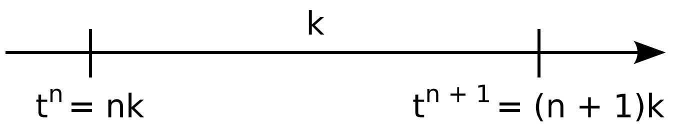
$$(x^{n+1})^2 + (v^{n+1})^2 = (x^n)^2 + (v^n)^2$$

$$E^{n+1} = \frac{1}{2}(x^{n+1})^2 + \frac{1}{2}(v^{n+1})^2 = \frac{1}{2}(x^n)^2 + \frac{1}{2}(v^n)^2 = E^n$$

$$\Updownarrow$$

$$E^{n+1} = E^n$$

$$\begin{cases} \dot{u}(t) = f(u(t)) \\ u(0) = u^0 \end{cases}$$



Tidsstegning med Euler framåt:

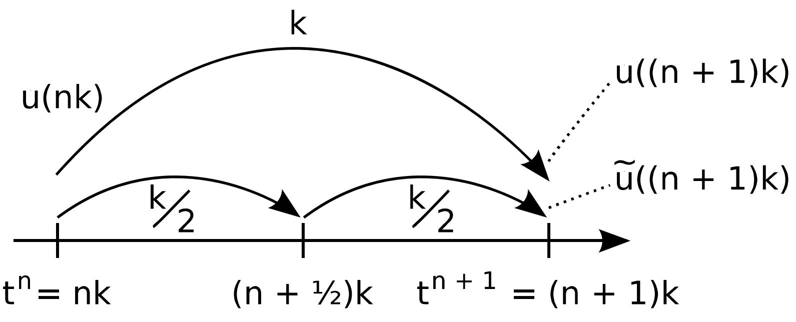
Ibland skrivs t_m som t^m

$$u((n+1)k) = u(nk) + f(u(nk))\cdot k$$

$$un + 1 = un + f(un)\cdot k = un + \dot{u}(nk)\cdot k = un + \dot{u}n\cdot k$$

f, Lipschitz-kontinuerlig

Skillnad mellan att ta ett tidssteg med längden k,
jämfört med tidssteg av längden $\frac{k}{2}$?



1. Antag att u och ũ startar från samma u(nk):

$$\left| u\big((n+1)k\big) - \tilde{u}\big((n+1)k\big) \right| = \{ S \triangleq u(nk) \} =$$

$$= \left| S + kf(S) - \left[S + \frac{k}{2} f(S) + \frac{k}{2} f\left(S + \frac{k}{2} f(S) \right) \right] \right| =$$

$$= \left| S + kf(S) - S - \frac{k}{2} f(S) - \frac{k}{2} f\left(S + \frac{k}{2} f(S) \right) \right| =$$

$$= \left| \frac{k}{2} f(S) - \frac{k}{2} f\left(S + \frac{k}{2} f(S) \right) \right| =$$

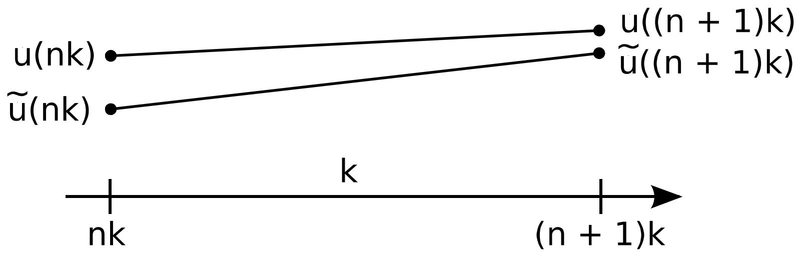
$$= \frac{k}{2} \left| f(S) - f\left(S + \frac{k}{2} f(S) \right) \right| \le$$

$$\le \frac{k}{2} L \left| S - \left(S + \frac{k}{2} f(S) \right) \right| =$$

$$= \frac{k^2}{4} L \left| f\big(u(nk)\big) \right|$$

2. Uppskatta skillnaden efter ett tidssteg med olika startvärden

$$u(nk) - \tilde{u}(nk)$$



$$\left| u\big((n+1)k\big) - \tilde{u}\big((n+1)k\big) \right| =$$

$$= \left| u(nk) + kf\big(u(nk)\big) - \tilde{u}(nk) + kf\big(\tilde{u}(nk)\big) \right| \le$$

$$= \left| u(nk) - \tilde{u}(nk) \right| + k \left| f\big(u(nk)\big) - f\big(\tilde{u}(nk)\big) \right| \le$$

$$= (1 + kL) \left| u(nk) - \tilde{u}(nk) \right|$$

1. + 2.

$\Downarrow$

Total felet $\left| u(T) - \tilde{u}(T) \right| \leq k\left(\exp(LT)\right)$

L = 10
T = 30

$\exp(LT) \gg 10^{100}$

Analys av felet och stabilitet

ODE: Hitta u(t) så att

$$\begin{cases} \dot{u}(t) + Au(t) = F(t), \quad t > 0 \\ u(0) = u^0 \end{cases}$$

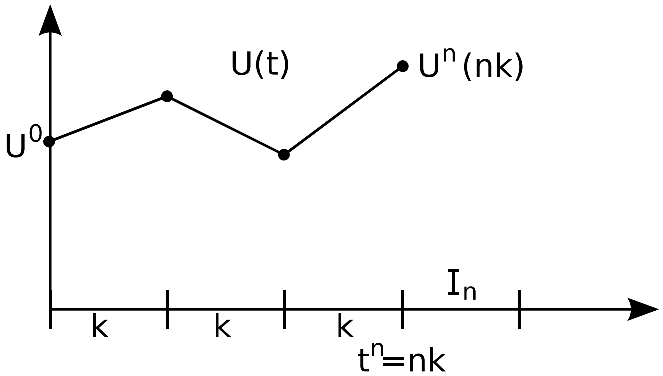
A — konstant

F(t) — given funktion

$$\left(\dot{u}(t) = f\left(t; u(t)\right) = F(t) - Au(t) \right)$$

Beräkna approximativt U(t) med trapetsmetoden.
Hitta $U^n = U(nk)$ så att $U^{n+1} + \frac{k}{2}\left(AU^{n+1} + AU^n\right) = U^n + \int_{I_n} F(t)\,dt, \quad n = 0, 1, 2, \ldots$

$U^0 \approx u^0$



$$\int_{I_n} F(t)\,dt \quad \text{kan beräknas exakt.}$$

$$\int_{I_n} \dot{U}(t)\,dt = U^{n+1} - U^n, \quad \int_{I_n} AU(t)\,dt = \frac{k}{2}\left(AU^{n+1} + AU^n\right)$$

Trapetsmetoden uppfyller:

$$\int_{I_n} \left(\dot{U} + AU - F\right) dt = 0, \quad n = 0, 1, 2, \ldots$$

Medelvärdet av residualen är noll:

$$\boxed{R(u) \equiv \dot{U} + AU - F}$$

över varje I_n

Beroende på A har ekvationen $\dot{u} + Au = F$ olika stabilitetsegenskaper:

1. konstant icke-negativ: $A \geq 0$ (stabil)
2. konstant imaginär: $A = i$
3. konstant negativ: $A < 0$ (instabil)
4. oscillerande positiv–negativ: $A(t) = \sin(t)$

Uppskatta felet till tiden $T = Nk$

Uppskatta $e(t) \triangleq u(t) - U(t)$

Inför dualproblem för att mäta känslighet

$$-\dot{\varphi}(t) + A\varphi(t) = 0, \quad T > t \geq 0 \quad \text{(baklänges i tiden)}$$

$$\varphi(T) = \pm 1 = \operatorname{sgn}\left(e(t)\right)$$

(Vi räknar med signum(0) = 0)

Felrepresentation:

$$0=\int_0^T e(t)\underbrace{\left(-\dot{\varphi}(t)+A\varphi(t)\right)}_{0}\,dt=\{\text{Partialintegration}\}=$$

$$=\int_0^T (\dot{e}+Ae)\varphi(t)\,dt-e(T)\varphi(T)+e(0)\varphi(0)$$

$$\Updownarrow$$

$$|e(T)|=-\int_0^T R(U)\varphi(t)\,dt+(u^0-U^0)\varphi(0)$$

$$|u(T)-U(T)| \le S_c(T)\max_n kR_n(U) + S_d\,|u(0)-U(0)|$$

Stabilitetsfaktorer:

$$\begin{cases} S_c(T)=\int_0^T |\dot{\varphi}(s)|\,ds \\ S_d(T)=|\varphi(0)| \end{cases}$$

2010−(11)nov−11: dag 8

Ordinära differentalekvationer (ODE)

— kort teori
— modul 3 – game
— program (~solve) (kod på tavlan). Jobba själv ☺
— hur testar vi om vi inte förstår?

ODE: (*) $\frac{du}{dt}=f(t;u), \quad u(t_0)=u_0$

Lösning till (*)

Exempel:

$$\frac{du}{dt}=-u, \quad u(0)=1$$

$$u(t)=e^{-t}$$

Lös med hjälp av tidsstegning

Eulers metod:

$$u^{n+1}=u^n+dt\, f(t^n;u^n), \quad n=0, 1, 2, \ldots$$

Om vi vill lösa $\frac{du}{dt}=f(t;u), \quad u(t_0)=u_{0,} \quad t_0 \le t \le T$

n = 0, 1, 2, …, N

N? Hur många steg måst man tag? Antal steg = N + 1, N är sista steget eftersom man börjar på 0.

Tidssteg dt ger N:

$$N=\frac{T-t_0}{dt}$$

System av ODE:er: (Jämför övning 1)

$$(T)\begin{cases}\dfrac{dv}{dt}=-\cos(t) & v(0)=0\\[2mm] \dfrac{dx}{dt}=v(t) & x(0)=0\end{cases}$$

$$\vec{u}=\begin{bmatrix}u_1(t)\\u_2(t)\end{bmatrix},\quad \vec{f}=\begin{bmatrix}f_1(t;u_1;u_2)\\f_2(t;u_1;u_2)\end{bmatrix}\qquad \left(\vec{u}=\coprod_{\forall u}u(t),\quad \vec{f}=\coprod_{\forall u}\left(t;\overbrace{\coprod_{\forall u}u}^{\vec{u}}\right)\right)$$

$$\left.\begin{array}{l}\vec{u}=\begin{bmatrix}v(t)\\x(t)\end{bmatrix},\quad \vec{f}=\begin{bmatrix}-\cos(t)\\v(t)\end{bmatrix}\\[4mm] \dfrac{d\vec{u}}{dt}=\vec{f},\quad \vec{u}(0)=\begin{bmatrix}0\\0\end{bmatrix}\end{array}\right\}\triangleq (**)$$

($\triangleq$ användes på tavlan)

$$\frac{d\vec{u}}{dt}=\vec{f}(t;\vec{u}),\quad \vec{u}(t_0)=\vec{u}^0$$

Bra (!) nummeriska metoder (tidsstegning) fungerar på detta sätt!

Eulers metod: $\vec{u}^{n+1}=\vec{u}^n+dt\,\vec{f}\left(t^n;\vec{u}^n\right)$

Egenskaper hos olika metoder: $\dfrac{du}{dt}=f(t;u),\quad t_0\le t\le T$

Euler framåt: $u^{n+1}=u^n+dt\,f\left(t_n;u^n\right)$

Noggrannhetsordning: 1 $\quad |\underbrace{u(T)}_{dt}-\underbrace{\tilde{u}(T)}_{dt/2}|\approx k\cdot dt^1$

k — konstant
felet avtar linjärt med dt

Explicit metod (givet u^n kan vi beräkna u^{n+1} direkt!)

Ej energibevarande: (varför är utanför kursen)
energin stiger.

Euler bakåt: $u^{n+1} = u^n + dt\, f\left(t_{n+1};\, u^{n+1}\right)$

Noggrannhetsordning: 1

Implicit metod (givet u^n kan ej beräkna u^{n+1} direkt, extra åtgärder krävs)

Ej energibevarande:
energin avtar.

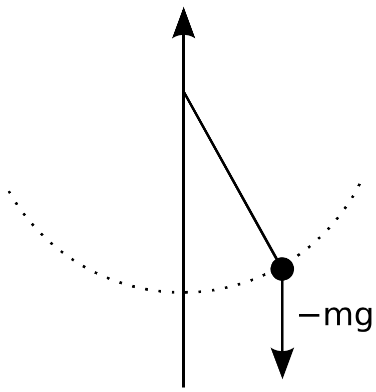
Trapetsmetoden: $u^{n+1} = u^n + \frac{dt}{2}\left[f\left(t_n;\, u^n\right) + f\left(t_{n+1};\, u^{n+1}\right)\right]$

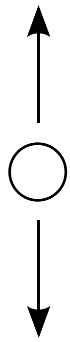
Implicit metod

Energibevarande

Noggrannhetsordning: 2 $\left|u(T) - \tilde{u}(T)\right| \approx k \cdot dt^2$

$$(T)\ \begin{cases} \dfrac{dx}{dt} = v(t) \\ \dfrac{dv}{dt} = -x(t) \end{cases} \quad \begin{array}{|l} x(0)=0 \\ v(0)=1 \end{array}$$





Harmonisk oscillation:

Svänger för evig,
energin är bevarad.

Totala energin i systemet

$$E(t) = \frac{v(t)^2}{2} + \frac{x(t)^2}{2}$$

kvadraterna och halveringarna spelar inge roll
när man analyserar om enegin bevaras eller ej.

ode_solve.py Kommer läggas upp på Katarinas nada-sida
------------------ http://www.nada.kth.se/~katarina/

```
from pylab import *
import ode

def f(t, u):

     #Tidsintervall
     I = [0, 2]
     t0 = I[0]
     T = I[1]
     dt = 1 #Steglängd

     #Antal stag
     N = (T - d0) / dt + 1

     #Vektor med tidspunkter, (t⁰, t¹, …, tᴺ)
     tarray = linespace(t0, T, N) #[0 1 2]

     #Begynnelsedata
     u0 = 1 #Ses som array av storleken 1

     #Definiera en array för lösningen
     uarray = zeros([size(u0), size(tarray)])    #[0 0 0]

     uarray[:, 0] = u0   #kolonn 0 för alla rader i uarray = u0

     un = u0
     i = 1
     for tn in tarray[0, N - 1]:
          u = ode.timestep(f, tn, un, dt, "Euler")
          uarray[:, i] = u    #[1 u¹ 0]
          i += 1
          un = u    #Uppdatera un

     plot(tarray, uarray[0])
```

2010−(11)nov−15: dag 9

M4: Skalär ODE:

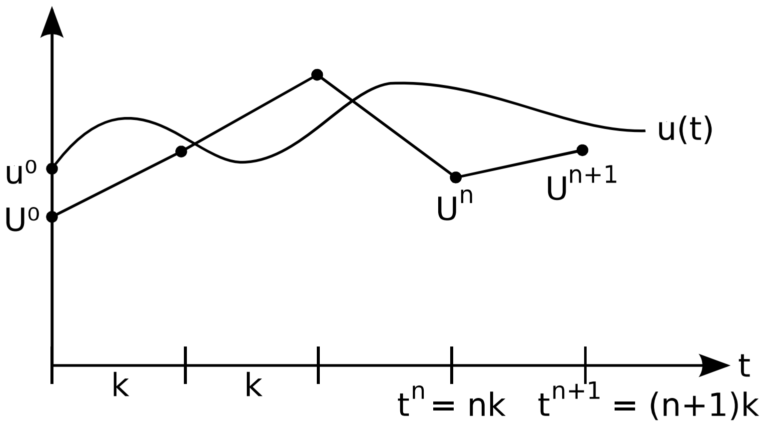
$$\begin{cases} \dot{u}(t)+Au(t)=F(t), \quad t>0 \\ u(0)=u^0 \end{cases}$$

A konstant, F(t) given funktion

Trapetsmetod:

Hitta en styckvis linjär U(t)

$$U^{n+1}+\frac{k}{2}\left(AU^{n+1}+AU^{n}\right)=U^{n}+\int_{I_n} F(t)\,dt, \quad n=1,\,2,\,\ldots$$



$U^0 \approx u^0$

Inte en interpolation, men ett försöl att approximera den.

Residual:

$$R(U)=\dot{U}+AU-F \qquad\qquad (R(u) = 0)$$

Dualproblem:

$$\begin{cases} -\dot{\varphi}+A\varphi=0, \quad T>t\geq 0 \\ \varphi(T)=\mathrm{sgn}(e(t)) \end{cases}$$

Fel: $e = u - U$

Kommentar: (Detta kan hoppas över.)

$\langle Av;\, w\rangle=\ldots=\langle v;\, A^*\, w\rangle$ där * är en dualoperator

$A=\dot{v}-\Delta u$

$\langle \dot{v}+Av;\, w\rangle=\int v(-\dot{w}+Aw)\, dx\, dt=\langle v\,;-\dot{w}+Aw\rangle$

Felrepresentation:

$$|e(T)|=-\int_0^T R(U)\varphi\, dt+e(0)\varphi(0) \qquad (1)$$

Låt $\bar{\varphi}$, medelvärdet över I_n: $\bar{\varphi}(t)=\frac{1}{k}\int_{I_n} \varphi(s)\, ds$

($\bar{\varphi}$ konstant över I_n)

Vi har att $\int_{I_n} R(U)\, dt=0$

$\Downarrow$

$$\int_{I_n} R(U)\bar{\varphi}\, dt=0 \;\Rightarrow\; \int_0^T R(U)\bar{\varphi}\, dt=0 \qquad (2)$$

(1) − (2):

$$|e(T)| = -\int_0^T R(U)(\varphi-\bar{\varphi})\,dt + e(0)\varphi(0)$$

$$\Downarrow$$

$$|e(T)| \le \sum_{n=0}^{N-1} R_n \int_{I_n} |\varphi-\bar{\varphi}|\,dt + |e(0)|\cdot|\varphi(0)|$$

$$R_n(U) \triangleq \max_{t\in I_n} |R(U(t))|$$

A posterion-feluppskattning:

$$|u(T)-U(T)| \le S_c(t)\max_n kR_n(U) + S_d(t)|u(0)-U(0)|$$

$$\begin{cases} S_d(T)=|\varphi(0)| \\ S_c(T)=\int_0^T |\dot{\varphi}|\,dt \end{cases}$$

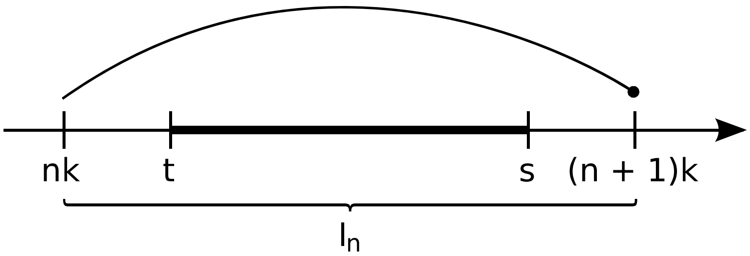
Bevis:

$$\int_{I_n} |\varphi - \bar{\varphi}| \, dt \; \leq \; k \int_{I_n} |\dot{\varphi}| \, dt \qquad (3)$$

$$\int_{I_n} |\varphi - \bar{\varphi}| \, dt \;\; ?$$

$$\varphi(t) - \bar{\varphi}\, t \; = \; \frac{1}{k} \int_{I_n} \left(\varphi(t) - \varphi(s)\right) ds \qquad (*)$$

$$\left|\varphi(t) - \varphi(s)\right| \; = \; \left| \int_s^t \dot{\varphi}(\sigma)\, d\sigma \right| \; \geq \; \int_s^t \left|\dot{\varphi}(\sigma)\, d\sigma\right| \qquad (**)$$



$$nk \leq t < s \leq (n+1)k$$

$$\int_{I_n} |\varphi - \bar{\varphi}| \, dt \; \underset{(*)}{=} \; \int_{I_n} \left| \frac{1}{k} \int_{I_n} \left(\varphi(t) - \varphi(s)\right) ds \right| dt \; \leq$$

$$\leq \; \int_{I_n} \frac{1}{k} \int_{I_n} \left|\varphi(t) - \varphi(s)\right| ds \, dt \; \underset{(**)}{\leq} \; \int_{I_n} \frac{1}{k} \int_{I_n} \int_{I_n} |\dot{\varphi}(\sigma)| \, d\sigma \, ds \, dt \; =$$

$$= \; k \int_{I_n} |\dot{\varphi}(\sigma)| \, d\sigma \qquad \blacksquare$$

Dual:

$$\begin{cases} -\dot{\varphi} + A\varphi = 0 \\ \varphi(T) = \operatorname{sgn}\left(e(t)\right) = \pm 1 \end{cases}$$

Exakt lösning:

$$\varphi(t) = \pm \exp\left(-A(T - t)\right)$$

(Primal):

$$(P)\begin{cases}\dot{u}+Au=F(t)\\ u(0)+u^0\end{cases}$$

Stabilitet hos (P) beror på A.

1) $\quad A < 0$: $\quad S_d(T) \leq \exp\left(|A|\,T\right)$,

$\qquad\qquad S_c(T) \leq \exp\left(|A|\,T\right)$

2) $\quad A \geq 0$: $\quad S_d(T) \leq 1,\; S_c(T) \leq 1$

S_c — beräkningsdel
S_d — datafel

Generalisering

1.

$$\begin{cases}\dot{\vec{u}}(t)+\mathbf{A}\vec{u}(t)=\vec{F}(t), \quad t>0\\ \vec{u}(0)=\vec{u}^0\end{cases}, \qquad \vec{u}(t)=\begin{pmatrix}u_1(t)\\ u_2(t)\\ \vdots\\ u_m(t)\end{pmatrix}, \qquad \vec{F}(t)=\begin{pmatrix}F_1(t)\\ F_2(t)\\ \vdots\\ F_m(t)\end{pmatrix}$$

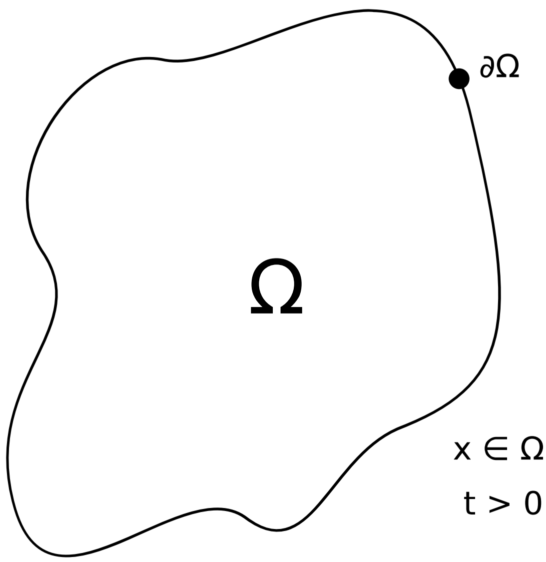
A — m×m-matris

(Modul 4 och 5)

2. Partiella differentialekationer (PDE):

Hitta $\vec{u}(x;t)$:

$$\begin{cases}\dot{\vec{u}}(x;t)+\mathbf{A}\vec{u}(x;t)=\vec{F}(x;t)\\ \vec{u}(x;0)=\vec{u}^0(x)\\ \text{randvillkor}\end{cases}$$



(Modul 6)

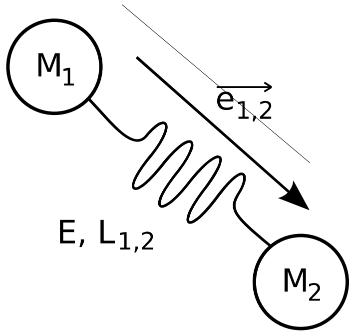
$$\mathbf{A}\vec{u}=\Delta\vec{u}$$

$$\mathbf{A}\vec{u}=\beta\cdot\nabla\vec{u}-\varepsilon\Delta\vec{u}$$

Dual problem:

$$\begin{cases} \text{ODE-system:} & -\dot{\varphi} + A^T \cdot \varphi = 0 \\ \text{PDE:} & -\dot{\varphi} + A^* \cdot \varphi = 0 \end{cases}$$

Exempel: Mass-fjäder-system:



Koordinater: x^1 (M_1)
x^2 (M_2)

Hastigheter: v^1 (M_1)
v^2 (M_2)

Kraft på M_1 från M_2:

$$F_{1,2} = \left\{ \begin{array}{l} r_{1,2} = |x^1 - x^2| \\ e_{1,2} = \dfrac{x^2 - x^1}{r_{1,2}} \end{array} \right\} = E\left(r_{1,2} - L_{1,2}\right)\overrightarrow{e_{1,2}}$$

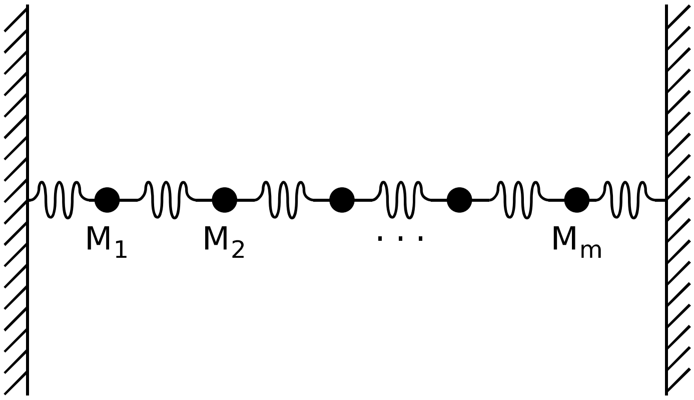
E — Fjäderkonstant
$\vec{e}$ — Normerad riktningsvektor (längden 1)
L — Fjäderns vilolängd
r — Avstånd

M_2 från M_1:

$F_{2,1} = -F_{1,2}$ (Newtons 3:e lag)

Rörelseekvationer:

$$\left\{ \begin{array}{l|l} \dot{x}^1 = v^1 & \dot{x}^2 = v^2 \\ \dot{v}^1 = \dfrac{F_{1,2}}{M_1} & \dot{v}^2 = \dfrac{F_{2,1}}{M_2} \end{array} \right.$$



På varje massa M_i, $i \neq 1, m$ agerar 2 fjäderkrafter:

$$\begin{cases} F_{i,i-1} = E\left(r_{i,i-1} - L_{i,i-1}\right) e_{i,i-1} \\ F_{i,i+1} = E\left(r_{i,i+1} - L_{i,i+1}\right) e_{i,i+1} \end{cases}$$

Rörelseekvation för M_i:

$$\dot{x}^i = v^i$$

$$\dot{v}^i = \frac{\overbrace{E\left(r_{i,i-1} - L_{i,i-1}\right) e_{i,i-1}}^{F_{i,i-1}}}{M_i} + \frac{\overbrace{E\left(r_{i,i+1} - L_{i,i+1}\right) e_{i,i+1}}^{F_{i,i+1}}}{M_i}$$

Antag $E = 1$, $M_i = 1$, $L_{i,j} = 0$

$\Downarrow$

$$\dot{v}^i = \frac{1 \cdot r_{i,i-1}\, e_{i,i-1}}{1} + \frac{1 \cdot r_{i,i+1}\, e_{i,i+1}}{1}$$

$$e_{i,j} = \frac{x_j - x_i}{r_{i,j}} \quad \Rightarrow \quad \dot{v}^i = \left(x^{i-1} - x^i\right) + \left(x^{i+1} - x^i\right) = x^{x-1} - 2x^i + x^{x+1}$$

$$\dot{\vec{u}} = \begin{bmatrix} x^1 \\ x^2 \\ \vdots \\ x^m \\ \hline v^1 \\ v^2 \\ \vdots \\ v^m \end{bmatrix} = \begin{bmatrix} v^1 \\ v^2 \\ \vdots \\ v^m \\ \hline -x^1 + x^2 \\ x^1 - 2x^2 + x^3 \\ \vdots \\ x^{m-1} - x^n \end{bmatrix} = \left[\begin{array}{cccc|cccc} & & & & 1 & & 0 & \\ & 0 & & & & 1 & & \\ & & & & & & \ddots & \\ & & & & 0 & & & 1 \\ \hline -1 & 1 & & 0 & & & & \\ & -12 & -1 & & & & & \\ & & -12 & -1 & & 0 & & \\ 0 & & & \ddots & & & & \end{array}\right] \begin{bmatrix} x^1 \\ x^2 \\ \vdots \\ x^m \\ \hline v^1 \\ v^2 \\ \vdots \\ v^m \end{bmatrix}$$

(Detta är skumt, men vi kommer gå i på det senare.)

2010–(11)nov–22: dag 10

Partikelsystem:

N partiklar (punktmassor) där partikel i (index i) har massan m^i,

position $\vec{x}^i = \begin{pmatrix} \vec{x}^i{}_x \\ \vec{x}^i{}_y \end{pmatrix}$ och hastigheten $\vec{v}^i = \begin{pmatrix} \vec{v}^i{}_x \\ \vec{v}^i{}_y \end{pmatrix}$ (i 2D).

Newtons andra lag:

$$\dot{\vec{x}}^i = \vec{v}^i$$

$$\dot{\vec{v}}^i = \frac{\vec{F}^i}{m^i}$$

Parvisa krafter:

$$F^i = \sum_{j=0}^{N} F^{ij}$$

Generell ODE-form (vektorform):

$$\vec{u} = \begin{pmatrix} \vec{x}_x \\ \vec{x}_y \\ \vec{v}_x \\ \vec{v}_y \end{pmatrix} \qquad \vec{f} = \begin{pmatrix} \vec{v}_x \\ \vec{v}_y \\ \vec{F}_x \\ \vec{F}_y \end{pmatrix} \qquad \dot{\vec{u}} = \vec{f}(\vec{u})$$

Lös med generell ODE-lösare (`solve()` från modul 3, tidsstegning)

Exempelvis:

Trapetsmetoden: $u^{n+1} = u^n + \frac{k}{2}\left(f(u^n) + f(u^{n+1})\right)$

Bakåt Euler: $u^{n+1} = u^n + kf(u^{n+1})$

```
# Initial values
x[0, 0] = 0.0  # Particle 0 starts in x=0
[...]

# Pack values into u
u[0 * M : 1 * M] = x[:, 0]
u[1 * M : 2 * M] = x[:, 1]
u[2 * M : 3 * M] = v[:, 0]
u[3 * M : 4 * M] = v[:, 1]

def f3body(t, u):
    # Unpack values into x and v
    x[:, 0] = u[0 * M : 1 * M]
    x[:, 1] = u[1 * M : 2 * M]
    v[:, 0] = u[2 * M : 3 * M]
    v[:, 1] = u[3 * M : 4 * M]

    a = zeros((M, 2))

    m = 1.0     # Mass
    E = 100.0   # Stiffness coefficient
    B = 4.0     # Damping coefficient
    L = 2.0     # Spring rest length

    for i0 in range(0, M):
        for i1 in range(0, M):
            # Don't compute force with self
            if (i0 == i1):
                continue

            r = norm(x[i1, :] - x[i0, :])
            e = (x[i1, :] - x[i0, :]) / r
            vr = v[i1, :] - v[i0, :]

            F = E * (r - L) * e      # Elastic spring force
            D = B * dot(vr, e) * e   # Damping spring force

            # Gravity force
            G = 0 # Add this yourself

            a[i0, :] += (F + D + G) / m

    # Pack values into fval
    fval = zeros(4 * M)
    fval[0 * M : 1 * M] = v[:, 0] # (velocities)
    fval[1 * M : 2 * M] = v[:, 1]
    fval[2 * M : 3 * M] = a[:, 0] # (forces / mass)
    fval[3 * M : 4 * M] = a[:, 1]

    return fval
```

Fixpunktsform:

Skriv ekvationen R(q) = 0 i fixpunktsform: q = g(q)
(Det finns hur många former som helst.)

Fixpunktsiteration: $q_{m+1} = g(q_m)$

I vårt exempel hade vi redan en fixpunktsform:

$q_{m+1} = g(q) = u^n + k f(q_m)$

(Notera olika index for tidssteg och iteration.)

Konvergerar iterationen?

```
def fixedpoint(g, q0):
    # Choose initial guess, parameters
    q = q0
    TOL = 1.0e-8
    res = 1.0

    # Iterate until convergence
    while (res > TOL):
        s = g(q)
        res = norm(s - q)
        q = s

    return x
```

Iterationen kan bete sig på olika sätt:

Snabbt konvergerandes

Divergerandes

Långsamt konvergerandes

Långsamt konvergerandes och alternaterande

Contraction mapping:

$f(0)=0$ Skriv som fixpunktsiteration

$u^{n+1}=g(u^n)$

$e^{n+1}=u^{n+1}-u^n=g(u^n)-g(u^{n-1})$

$|e^{n+1}|=|g(u^n)-g(u^{n+1})|$ Om g är lipschitz-kontinuerlig

$|e^{n+1}|=|g(u^n)-g(u^{n+1})|\leq L|u^n-u^{n-1}|$

Läs 76-77 kap. $|e^{n+1}|\leq L|e^n|$

L måste vara mindre än 1 i fixpunktsmetoden för att få konvergens. [SF1613]

$|g'(u)|<1$

Om L = 0 får vi Newtons metod.

Newtons metod konvergerar (nästan) alltid.

Generell algebraisk ekvationslösning :

R(q) = 0

Exempel: R(q) = $x^2 - 2 = 0$ (roten ur 2)

Även system (se linjära system senare till exempel)

Newtons metod:

Vi kan skriva en generell fixpunktsform för ekvationen $R(q) = 0$:

$$q = g(q) = q - \alpha R(q)$$

Konvergens:

$$g'(q) = 1 - \alpha' R(q) - \alpha R'(q)$$

Använd $R(q) = 0$:

$$g'(q) = 1 - \alpha R'(q)$$

Optimal metod:

$$g'(q) = 0$$

$$\Downarrow$$

$$1 - \alpha R'(q) = 0$$

$$\Downarrow$$

$$\alpha = \frac{1}{R(q)}$$

Alltså: Newtons metod:

$$q = q - \frac{R(q)}{R'(q)}$$

Samma sak, men $R'(q)$ är en matris:

$$q_{m+1} = q_m - \frac{R(q_m)}{R'(q_m)}$$

$$J = R'(q_m) \quad \Rightarrow \quad J \cdot q_{m+1} = J \cdot q_m - R(q_m)$$

(J är en jacobimatris)

```
def newton(f, x0):
    class LocalData:
        def __init__(self):
            self.f = f

    def g(x, iter, data):
        f = data.f

        # Compute Jacobian
        J = jacobian(f, x)
        # Compute right hand side
        r = dot(J, x) - f(x)
        # Solve linear system
        y = solve(J, r)

        return y

    data = LocalData()

    # Iterate the Newton g(x)
    return fixedpoint(g, x0, data)
```

```
def newton_fixedpoint_adapter(g):
    def R(x):
        return x - g(x)

    return R

def g(u):
    t = t0 + k
    return u0 + 0.5 * k * f(t0, u0) + 0.5 * k * f(t, u)

fnewton = newton_fixedpoint_adapter(g)
return newton(fnewton, u0)
```

Jacobi-iteration:

$$\mathbf{A}\vec{x}=\vec{b} \quad \Rightarrow \quad \{\mathbf{A}=\mathbf{D}+\mathbf{M}\} \quad \Rightarrow \quad x=\mathbf{D}^{-1}(-\mathbf{M}\vec{x}+\vec{b})=\vec{g}(\vec{x})$$

$$\|\vec{g}'\|=\|\mathbf{D}^{-1}\mathbf{M}\|<1$$

Fungerar endast för icke-linjära system.

Steepest Descent:

$$\vec{x}=\vec{x}-\alpha(\mathbf{A}\vec{x}-\vec{b})=\vec{g}(\vec{x}) \qquad (\vec{r}=\mathbf{A}\vec{x}-\vec{b})$$

$$\|\vec{g}'\|=\|\mathbf{I}-\alpha\mathbf{A}\|=0 \quad \Rightarrow \quad \alpha=\frac{\langle\vec{r};\vec{r}\rangle}{\langle\vec{r};\mathbf{A}\vec{r}\rangle}$$

Conjugate Gradient:

Samma som Steepest Decent, men man räknar $\vec{r}$ som en ortogonalsering mot Krylov-vektorer/rum.

2010–(11)nov–29: dag 11

Idag: Titta på kopplingen kontinuum och rumsderivata — diskret/partikelmodell (vågekvation och mass–fjäder)

Visa att mass–fjärdenmodellen representerar en rumsderivata
Diskret ⇒ Kontinuum

Detta gör att vi kan beskriva ekvationen mer kompakt som en partiell differentialekvation (vågekvation)

Först:

Vi har tittat på diskretisering av differentialekvationer i tiden (ODE/begynnelsevärdesproblem) — vad är diskretisering i rummet (PDE/randvärdesproblem)?

Vad händer när upplösningen (antalet pariklar) ökar?

Introducera finita elementmetoden (FEM)
Kontinuum ⇒ Diskret

Knyta ihop flera moment i kursen till ett sammanhang:
styckvisa linjära funktioner definierade av basfunktioner på ett nät/mesh (M2) ,
kvadratur/integrering (M2) och tidsstegning (M1/M3) knyts ihop för att definiera finita elementmetoden .
Vi kommer att jämföra med vågekvation/mass–fjäder (M4).

Referenser för dagen lektion:

Elastisk sträng — ekvivalens mellan mass–fjädermodell och vågekvation (45 kap.)

Partiella differentialekvationer, finita elementmetoden (149-150 kap.)

Modell: mass–fjäder, representerar vågutbredning i 1D.
u är skalär för enkelhets skull.

ODE:

$$\begin{cases} \dot{u}^i = v^i \\ Mh\dot{v}^i = F^i \end{cases}$$

Fjäderkrafter på en partikel, i (index i):

$$F_{i,i+1} = \frac{E}{h}\left(u^{i+1} - u^i\right) \quad \text{och} \quad F_{i,i-1} = -\frac{E}{h}\left(u^i - u^{i-1}\right) \quad \text{alltså} \quad F_{i,i-1} = \frac{E}{h}\left(u^{i-1} - u^i\right)$$

$F_{i,j}$ är kraften på i från j. $F_{j,i} = -F_{i,j}$. Allmänt:

$$F_{i,j} = \frac{E}{h}\left(u^i - u^j\right) \quad \text{för sammakopplade partiklar, i och j.}$$

Totalkraft på partikeln, i:

$$F_i = F_{i,i+1} + F_{i,i-1} = \frac{E}{h}\left(u^{i+1} - 2u^i + u^{i-1}\right)$$

$$Mh\dot{v}^i = F^i \stackrel{\text{def}}{=} F_i = \frac{E}{h}\left(u^{i+1} - 2u^i + u^{i-1}\right)$$

$$\frac{E}{M} = 1 \ \Rightarrow \ \dot{v}^i(t) = \ddot{u}^i(t) \triangleq \frac{u^{i+1} - 2u^i + u^{i-1}}{h^2}, \quad i = 1, \dots, J$$

Definitioner: $\Delta x = h, \quad \Delta u^i = u^{i+1} - u^i, \quad E = 1$ (E = 1 är ad hoc)

Fjäderkraft:

$$F_{i,i+1} = \frac{\Delta u^i}{\Delta x} = \frac{u^{i+1} - u^i}{h} \quad \cong_{h \to 0^+} \frac{du}{dx} = u'$$

Derivata (för infinitesimala h; små approximerar):

$$u'(x) = \frac{du}{dx} = \frac{u(x+h) - u(x)}{h}$$

u' är rumsderivata; $\dot{u}$ är tidsderivata

$$u''(x) = \frac{u'(x+h) - u'(x)}{h} = \frac{\frac{u(x+h) - u(x)}{h} - \frac{u(x) - u(x-h)}{h}}{h} =$$

$$= \frac{u(x+h) - 2u(x) + u(x-h)}{h^2}$$

Vi kan alltså skriva mass–fjädermodellen som vågekvationen:

$\ddot{u}(x;t) = u''(x;t)$ för $x \in]0; 1[, \ t > 0$

$u(0;t) = u(1;t) = 0$ för $t > 0$

$u(x;0) = u^0(x), \quad \dot{u}(x;0) = \dot{u}^0(x)$ för $x \in]0; 1[$

där $u^0(x)$ och $\dot{u}^0(x)$ är givna funktioner.
u(0; t) och u(1; t) är randvillkor som vi har modellerat med stora massor i ändarna.

$\dot{u} = v$

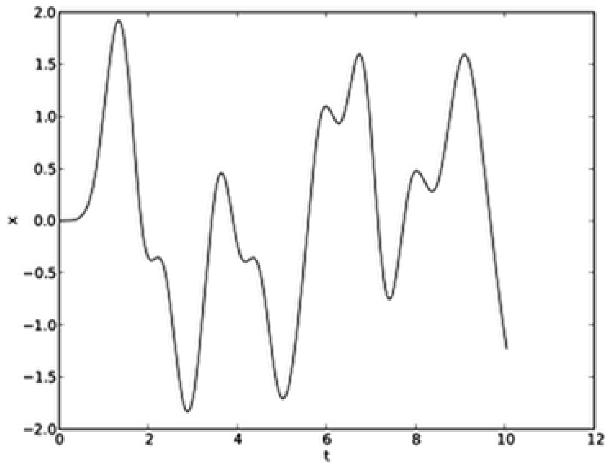
$\dot{v} = F'$

$F = u'$

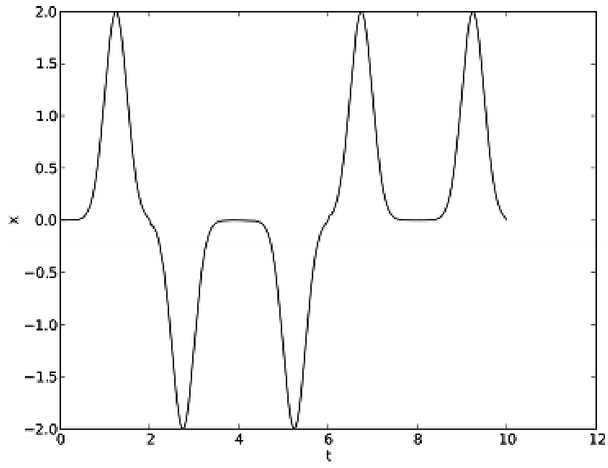
där u är förskjutning, v är hastighet och F är fjäderspänning samt att F' är fjäderkraft som agerar på partiklar.

När antalet partiklar ökar från 20 (M = 20) till 1280 förfinas simulationen:

M = 20



M = 1280



Vi vill börja direkt från kontinuummodellen (rums och tidsderivata) och automatiskt konstruera en diskret modell. Vi formulerar finita elementmetoden som ett sätt att göra det.

Hur ska vi göra i 2D/3D?
Hur ska vi sätta fjäderparametrar systematiskt &c?
Hur ska vi modellera andra fenomen?

Hur ser lösningen ut mellan punkterna?

Finns det ett systematiskt sätt att gå från kontinuum till diskret för generella partiella differentialekvationer?

Vi vill börja med att lösa differentialekvationen

$$R(u) = Au - g = 0$$

där A är någon operator på u (till exempel differentialoperator)

Exempel: $Au = \ddot{u} - u''$ och $g = 0 \Rightarrow \ddot{u} - u'' = 0$ (Vågekvation)

Exempel: $Au = Iu \Rightarrow u = g$

A är identitet (I).
Blir en L2-projektion.
Enkelt exempel för att förstå metoden.

Som lösning söker vi en funktion u(x; t) som en linjärkombination:

$$u(x;t)=\sum_{j=1}^{J} u_j(t)\varphi_j(x)$$ (Läraren använde dock ϕ istället för φ)

Repetition:

Sida 1 till 3
2010-(11)nov-01
(4:e dagen)

av J givna basfunktioner, $\varphi_1(x), \ldots, \varphi_J(x)$, som beror av x, med okända koefficienter, $u_1(t), \ldots, u_J(t)$, som beror av t.
Basfunktionerna är styckvis linjära "hattfunktioner".

$\varphi_i(jh) = 1$ om $j = i$, $\varphi_i(jh) = 0$ om $i \neq j$, $i, j = 1, \ldots, J$. Linjär interpolas!

Finita elementmetoden: L2-projektion

Exempel: Lös u = g med FEM.

1. Välj ett beräkningsnät (bestämmer basfunktionerna)

2. Definiera lösningen som en linjärr komination

$$u(x;t)=\sum_{j=1}^{J} u_j(t)\varphi_j(x)$$

3. Definiera villkor/ekvationer för koefficienter:

$$\int_a^b R(u)\varphi_i\,dx=0, \quad i=0, 1, \ldots, J$$

$$\int_a^b u\varphi_i\,dx=\int_a^b g\varphi_i\,dx, \quad i=0, 1, \ldots, J$$

(Vi har J koefficienter (från u) och j villkor/ekvationer.)

4. Lös systemet för koefficienterna.
I det här fallet blir systemet ett linjärt ekvationssystem: $\mathbf{A}u_j=\vec{b}$

Finita elementmetoden: Vågekvation

Exempel: Lös ü − u'' = 0 med FEM.

Samma första och andra steg.

3. Definiera villkor/ekvationer för koefficienter:

$$\int_a^b R(u)\varphi_i \, dx = 0, \quad i = 0, 1, \ldots, J$$

$$\int_a^b \ddot{u}\varphi_i \, dx + \int_a^b u'\varphi'_i = 0 \, dx, \quad i = 0, 1, \ldots, J$$

Vi partialintergrerar rumsferivatorna så att en hamnar på basfunktionen φ_i, ty u har inte två derivator.

(Vi har J koefficienter (från u) och j villkor/ekvationer.)

4. Lös systemet för koefficienterna.
I det här fallet blir systemet en ODE: $\ddot{u}^i + \mathbf{A} u^i = \vec{0}$

2010–(12)dec–01: dag 12

Enkelt DOLFIN värmeledning demo

```
from dolfin import *

# Create mesh and define basis
# functions (function space)
mesh = UnitInterval(8)
V = FunctionSpace(mesh, "CG", 1)

# Define FEM formulation
u = TrialFunction(V)
v = TestFunction(V)
f = Expression("0.0")
k = 0.001
u0 = Function(V)
u1 = Function(V)
# Heat equation with backward
# Euler timestepping
a = (u * v + k * inner(grad(u), grad(v))) * dx
L = (u0 * v + k * f * v) * dx

problem = VariationalProblem(a, L)


# Set initial condition
for v in vertices(mesh):
     p = v.point()[0]
     i = v.index()
     if (p > 0.5):
          u1.vector()[i] = 1.0
     else:
          u1.vector()[i] = 0.0

file = File("heat.pvd")

# Timestep
T = 2.0
t = 0.0
while (t < T):
     u0.assign(u1)
     u1 = problem.solve()

     t += k

     # Save solution in VTK format
     file << u1
```

Projektion (L2)

```
# Define basis functions
V = FunctionSpace(mesh, "CG", 1)
Pf = TrialFunction(V)
v = TestFunction(V)

# Function to project
f = dolfin.Expression("exp(-10*x[0])", degree = 5)

# FEM formulation
a = Pf * v * dx
L = f * v * dx

# Assemble (build) matrix and vector,
# solve linear system and definefunction
# as linear combination
problem = dolfin.VariationalProblem(a, L)
Pf = problem.solve()
```

Projektion (L2) expanded

```
# Define basis functions
V = FunctionSpace(mesh, "CG", 1)
Pf = TrialFunction(V)
v = TestFunction(V)

# Function to project
f = dolfin.Expression("exp(-10*x[0])", degree = 5)

# FEM formulation
a = Pf * v * dx
L = f * v * dx

# Define solution functionas
# linear combination of basis
# functions
Pf = Function(V)
x = Pf.vector()

# Assemble (build) matrix and vector
A = assemble(a)
b = assemble(L)

# Solve for x^i
solver = LinearSolver()
solver.solve(A, xi, b)
```

Andraderivata — matris

```
from dolfin import *

# Create mesh and define function space
mesh = UnitInterval(2)
V = FunctionSpace(mesh, "CG", 1)

# Define FEM formulation
u = TrialFunction(V)
v = TestFunction(V)
a = (inner(grad(u), grad(v))) * dx

A = assemble(a)
print A.array()
```

2010–(12)dec–06: dag 13

Randvärdesproblem:

Ekvation: $-u''(x) = f(x), \ x \in [a; b]$

Randvillkor typ 1: $u(a) = 0, \ u(b) = 0$
Randvillkor typ 2: $u'(a) = 0, \ u'(b) = 0$

Vi kan skriva ett generellt randvillkor som :

$u'(a) = \kappa u(a), \qquad u'(b) = \kappa u(b)$

$\kappa = 0$ blir av typ 2, $\kappa \to \infty$ blir av typ 1.

Partialintegration (kom ihåg):

$$\int_a^b w''v\,dx = -\int_a^b w'v'\,dx + w'(b)v(b) - w'(a)v(a)$$

$$R(u) = -u'' - f = 0$$

$\Downarrow$ Finita element metoden

$$\int_a^b R(u)\varphi_j\,dx = 0$$

$\Updownarrow$

$$\int_a^b (-u'' - f)\varphi_j\,dx = \int_a^b \left(-u''\varphi_j - f\varphi_j\right)dx = \{\text{partialintegration}\} =$$

$$= \int_a^b \left(u'\varphi'_j - f\varphi_j\right)dx + u'(b)\varphi_j(b) - u'(a)\varphi_j(a) = 0$$

Använd generellt randvillkor $u'(a) = \kappa u(a)$:

$$\int_a^b u'\varphi'_j\,dx + \kappa u(b)\varphi_j(b) - \kappa u(a)\varphi_j(a) = \int_a^b f\varphi_j\,dx$$

Randvillkor i Python med DOLFIN:

```
from dolfin import *

# Coefficient for boundary condition
class Kappa(Expression):
     def eval(self, values, x):
          if (x[0] < 0.5):
               values[0] = 0.0
          else:
               values[0] = 1.0e6

# Mesh and basis functions (function space)
mesh = UnitInterval(8)
V = FunctionSpace(mesh, "CG", 1)

# FEM formulation
u = TrialFunction(V)
v = TestFunction(V)
f = Expression("1.0")
kappa = Kappa(V)
# Equation: -u'' = f
a = (inner(grad(u), grad(v))) * dx + kappa * u * v * ds
L = (f * v) * dx

# Build linear system and solve for
# coefficients uh in linear combination
problem = VariationalProblem(a, L)
uh = problem.solve()
plot(uh)
```

Andraderivata – matris:

Använd definition av u som linjärkombination av basfunktioner:

$$u=\sum_{i=1}^{J} u_i \varphi_i(x)$$

$$\int_a^b u'\varphi'_j\,dx=\sum_{i=1}^{J}\int_a^b u_i\varphi'_i\varphi'_j\,dx,\quad j=1,2,\ldots,J$$

Alltså, för matriselement $\mathbf{A}_{ij}$:

$$\mathbf{A}_{ij}=\int_a^b \varphi'_j\varphi'_i\,dx$$

```
from dolfin import *

# Create mesh and define function space
mesh = UnitInterval(2)
V = FunctionSpace(mesh, "CG", 1)

# Define FEM formulation
u = TrialFunction(V)
v = TestFunction(V)
a = (inner(grad(u), grad(v))) * dx

A = assemble(a)
print A.array()
```

```
...
```

```
Matrix of size 3 x 3 has 7 nonzero entries .
[[ 2.  -2.   0.]
 [-2.   4.  -2.]
 [ 0.  -2.   2.]]
```

Konvektion-diffusion-reaktion en av en koncentration, u_1, av en art, som äts upp av en annan art med koncentrationen u_2 kan beskrivas med foljade ODE- och PDE-modeller:

ODE:

$$\begin{cases} \dot{u}_1 = -\alpha_1 u_1 u_2 \\ \dot{u}_2 = \alpha_2 u_1 u_2 - \alpha_3 u_2 \end{cases}$$

PDE:

$$\dot{u} + \beta \cdot \nabla u - \epsilon \Delta u = f(u)$$
$$[\dot{u} + \beta u' - \epsilon u'' = f(u)]$$

Elasticitet kan modelleras med en mass–fjädermodell i en ODE eller genom att använda en PDE-vågekvation:

ODE:

$$\begin{cases} \dot{x}^i = v^i \\ \dot{v}^i = \dfrac{F^i}{m^i} \\ F^i = \sum_{j=0}^{N} F^{ij} \\ F^{ij} = E\left(r^{ij} - L^{ij}\right)\vec{e}^{ij} \end{cases}$$

PDE:

$$\begin{cases} \dot{u}_1 = u_2 \\ \dot{u}_2 - \nabla \cdot \sigma = 0 \\ \dot{\sigma} = \dfrac{1}{2} E\left(\nabla u_2 + \nabla u_2^T\right) \end{cases}$$

Vågfenomen kan modelleras med en mass–fjädermodell i en ODE eller genom att använda en PDE-vågekvation:

Samma ODE som ovan.

PDE:

$$\begin{cases} \dot{u}_1 = u_2 \\ \dot{u}_2 = c^2 \Delta u_1 = 0 \end{cases}$$
$$\left[\dot{u}_2 = c^2 u''_1 = 0\right]$$

2010–(12)dec–08: dag 14

Om u är beräknat med tidssteget k, ū är beräknat med tidssteget k/2. Felluppskattning beräknas med:

$$|u - \bar{u}| \leq \frac{LT}{2} k$$

Generellt kan vi säga:

$$|u - \bar{u}| \leq Ck^p$$

Vi tittar alltså på skillnaderna och försöker bestämma p:

$d_1 \approx 2{,}87 - 2{,}31 = 0{,}56$
$d_2 \approx 2{,}31 - 2{,}25 = 0{,}07$
$d_3 \approx 2{,}24 - 2{,}23 = 0{,}01$

Varje halvering av k ger c:a en faktor $\frac{1}{8} = \frac{1}{2^3}$.

Allstså kan vi uppskatta att metoden är av ordning 3.

$\vec{f}(\vec{x})=,$

där $\vec{\cdot}$ och $\vec{x}$ är vektorvärda (arrayer).

Newtons metod:

$J \cdot x_{i+1} = J \cdot x_i - f(x_i)$

$J = f'(x)$ (Jacobianen av f; Jabobimatris)

```
def newton(f, x0):
    def g(x):
        # Compute the Jacobian: f'(x)
        J = jacobian(f, x)
        # Compute right hand side of Newton
        # iteration and solve the linear system.
        r = dot(J, x) - f(x)
        return linear_solve(J, r)

    return fixedpoint(g, x0)

def fixedpoint(g, x0):
    TOL = 1.0e-10
    while (diff > TOL):
        y = g(x)
        diff = max_norm(y - x)
        x = y

def f(x):
    [...]

x0 = zeros(3)
x = newton(f, x0)
```

$$\mathbf{A}\vec{x}=\vec{b}$$

där

$$\mathbf{A}=\begin{pmatrix}10 & 1 & 2\\ 0 & 10 & 1\\ 1 & 2 & 20\end{pmatrix},\quad \vec{b}=\begin{pmatrix}10\\12\\24\end{pmatrix}$$

Vi definierar **D** som diagonalen av **A** och **M** = **A** − **D**.
I detta fall:

$$\mathbf{D}=\begin{pmatrix}10 & 0 & 0\\ 0 & 10 & 0\\ 0 & 0 & 20\end{pmatrix},\quad \mathbf{M}=\begin{pmatrix}0 & 1 & 2\\ 0 & 0 & 1\\ 1 & 2 & 0\end{pmatrix}$$

Jacobis metod kan då formuleras som:

$$x_1=\mathbf{D}^{-1}\left(-\mathbf{M}\vec{x}_0+\vec{b}\right)$$

där 1 och 0 är iterationsnummer.

Vi väljer $\vec{x}_0=\begin{pmatrix}1 & 2 & 3\end{pmatrix}^T=\begin{pmatrix}1\\2\\3\end{pmatrix}$,

vi har då att $\vec{x}_1=\mathbf{D}^{-1}\left(-\mathbf{M}\vec{x}_0+\vec{b}\right)=\begin{pmatrix}\frac{4}{10} & \frac{9}{10} & \frac{19}{20}\end{pmatrix}^T=\begin{pmatrix}4/10\\9/10\\19/20\end{pmatrix}$.

Vi vill lösa begynnelsevärdesproblemet: $\dot{u} = f(t; w)$

$$w = \begin{pmatrix} w_0 \\ w_1 \end{pmatrix} = \begin{pmatrix} u \\ v \end{pmatrix}$$

$$\dot{w} = f(t;\ w) = \begin{pmatrix} f_0(t;\ w) \\ f_1(t;\ w) \end{pmatrix} = \begin{pmatrix} -w_1^{\ 2} w_0 + \sin(t) \\ -w_0^{\ 2} w_1 + \sin(2t) \end{pmatrix}$$

Formulera `timestep()` och `solve()` från modul 3 och anropa `solve()` med $f(t; w)$.

För att beräkna integralen $\int_0^b \left(u(t)^2 + v(t)^2\right) dt$ kan vi skriva:

$f_{energy}(t;\ z) = \left(u(t)^2 + v(t)^2\right)$ där vi stoppar in de beräknade lösningarna u(t) och v(t), genom exempelvis en funktion som `piecewise_linear_adapater()`. Sedan anropar vi `solve()` med $f_{energy}(t;\ z)$ för att beräkna integralen.

Nomenklaturlista

&c	et cetera	
ℂ	Alla komplexa tal	blackboard bold C
ℍ	Alla quaternions/hamiltoner ($i^2 = j^2 = k^2 = ijk = -1$)	blackboard bold H
ℕ	Alla naturliga tal (inklusive eller exklusive 0)	blackboard bold N
ℙ	Projektions rymd, sannolikhet, alla primtal, &c	blackboard bold P
ℚ	Alla rationalla tal	blackboard bold Q
ℝ	Alla reella tal	blackboard bold R
ℤ	Alla heltal tal	blackboard bold Z
ℑ	Imagionära delen $\Im(\alpha + i\beta) = \beta \neq i\beta$	script I
h	Planks konstant	
ℏ	Planks reducerade konstant (h / 2π)	
ℓ	liter	script l
℘	Weierstrass	script p
ℜ	Reella delen $\Re(\alpha + i\beta) = \alpha$	script R
K	Kelvin	
Å	Ångström	
ℱ	Fourierserieutveckling, eller Fouriertransformering	script F
ℒ	Linjär operation (ℒ(D), skrivs ofta L(D))	script L
ℵ	alef	
ℶ	bet	
ℷ	gimel	
ℸ	dalet	

∀	För alla
∁	Komplement
∂	Partial differential
∃	Det existerar
∄	Det existerar inte
∅	Tomma mängden
∆	Inkrement (delta)
∇	Nabla (gradient)
∈	Element av
∉	Inte element av
∋	Ägare till
∌	Inte ägare till
∎	(Matematisk) gravsten (Q.E.D.; slut av bevis)
∏	Produkt
∐	Coprodukt (se längre ner)
∑	Summa
±	Plus-minus: plus eller minus (se längre ner)
∓	Minus-plus: $\alpha \mp \beta = \alpha \pm (-\beta)$ (se längre ner)
\	Differens, till exempel $\mathbb{C} \setminus \mathbb{R}$ Alla icke-reella komplexa tal
∘	Ring operator, till exempel $(f \circ g)(\ ) = f(g(\ ))$
()	Variabel, precis som x, men saknar bokstav
(·)	Variant av () eller ( )
∝	Propotionellt med tillexempel om $y(x) = kx$ så är $y \propto x$

∴	Alltså		
∵	För att		
≀	Värde saknas		
∼	Är likartad med		
≁	Är inte likartad med		
≃	Asymptotiskt lika med (x^{-1} ≃ 0, x → ∞)		
	— eller —		
	Är likartad med eller lika med (ƒ(x) ≃ ℱ(x))		
≄	Inte asymptotiskt lika med		
≈	Ungefär lika med		
≉	Inte ungefär lika med		
≅	Approximativt lika med. Nästa samma sak som ≈.		
≆	Approximativt lika med, men inte faktiskt lika med		
≇	Varken approximativt lika med eller faktiskt lika med		
≊	Ungefär lika med, eller lika med		
≌	Alla lika med		
≍	Ekvivalent med		
≏	Skillnad mellan		
≐	Närmar sig gränsen		
≘	Korresponderar med		
≙	Uppskattar		
≚	likvinkligt med		
≜	Är lika med enligt ny definition för denna beräkning. Till exempel y ≜ uy_1.		
≝	Är lika med enligt definition. Till exempel i ≝ √(-1).		
≞	Mätt med		
≟	Ifrågasatt lika med		
≠	Inte lika med		
≡	Identiskt med, till exempel 5 ≡ 1 (Mod 2)		
≢	Inte identiskt med		
≣	Strikt ekvivalent med		
≤	Mindre eller lika med		
≥	Mer eller lika med		
≨	Mindre, men inte lika med		
≩	Mer, men inte lika med		
≪	Mycket mindre än		
≫	Mycker mer än		
≭	Inte ekvivalent med		
≺	Är innan		
≻	Är efter		
«	$\alpha \ll \beta = \alpha \cdot 2^{\beta}$	$\alpha \ll_{\xi} \beta = \alpha \cdot \xi^{\beta}$	
»	$\alpha \gg \beta = \alpha / 2^{\beta}$	$\alpha \gg_{\xi} \beta = \alpha / \xi^{\beta}$	
∪	Union	{a} ∪ {b} = {a; b}	{a; c} ∪ {b; c} = {a; b; c}
∩	Snitt	{a; c} ∩ {b; c} = {c}	
∨	Logiskt eller,	a ∨ b är sant omm a eller b, eller båda är sant	
∧	Logiskt och (men),	a ∧ b är sant omm både a och b är sanna	
{…}	En mängd med elementen …		
omm	Om och endast om		

$\operatorname{cis} x = \cos x + i \sin x = e^{ix}$

Pil ovanför = vektor
Tjock text = matris

$$a \pm b \pm c = a \pm_1 b \pm_1 c = \begin{cases} a+b+c \\ a-b-c \end{cases}$$

$$a \pm b \mp c = a \pm_1 b \mp_1 c = \begin{cases} a+b-c \\ a-b+c \end{cases}$$

$$a \pm_1 b \pm_2 c \mp_2 d = \begin{cases} a+b+c-d \\ a+b-c+d \\ a-b+c-d \\ a-b-c+d \end{cases}$$

$$a = \coprod_{n=s}^{N} a_n = (a_s, a_{s+1}, a_{s+2}, \ldots, a_{N-2}, a_{N-1}, a_N)$$

$$\left(\coprod_{i=0}^{n} {}^{\downarrow} \coprod_{j=0}^{m} {}^{\rightarrow} \ f(i;j) \right) = \begin{pmatrix} f(0;0) & f(0;1) & f(0;2) & \cdots & f(0;m) \\ f(1;0) & f(1;1) & f(1;2) & \cdots & f(1;m) \\ f(2;0) & f(2;1) & f(2;2) & \cdots & f(2;m) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f(n;0) & f(n;1) & f(n;2) & \cdots & f(n;m) \end{pmatrix}$$

Funktionskombinationer:

Antag att ƒ, g, h och k är funktioner.

(ƒ + g)(x) = ƒ(x) + g(x)
(ƒ – g)(x) = ƒ(x) – g(x)
(ƒ · g)(x) = ƒ(x) · g(x) = (ƒg)(x)
(ƒ / g)(x) = ƒ(x) / g(x)
(ƒ ∘ g)(x) = ƒ(g(x))
(ƒg + hk)(x) = ƒ(x) · g(x) + h(x) · k(x)

signum ±α = sign ±α = sgn ±α = ±1, α > 0
signum 0 = 0
signum ($re^{i\theta}$) = cis θ, r > 0

$\mathbb{Z}_{+}$ Alla positiva heltal (ℤ = heltal)
$\mathbb{Z}_{-}$ Alla negativa heltal
$\mathbb{Z}_{0+}$ Alla positiva heltal samt 0 (icke-negativa heltal)
$\mathbb{Z}_{0-}$ Alla negativa heltal samt 0 (icke-positiva heltal)
$\mathbb{Z}_{m..n}$ Alla heltal mellan m och n

ℕ är i mina dokument inklusiva 0, alltså samma mängd som $\mathbb{Z}_{0+}$