Per Jönsson

# Symbolisk matematik med MATLAB

Malmö 2010

Per Jönsson
Centrum för Teknikstudier
Malmö högskola
205 06 Malmö
email: per.jonsson@mah.se
hemsida: http://homeweb.mah.se/~tspejo/index.htm

© Per Jönsson och Studentlitteratur, 2010
Andra upplagan

Utdrag av denna text får inte kopieras eller mångfaldigas utan angivande av källa.

# Förord

Datoralgebraprogram (Computer Algebra Systems) kan användas för att förenkla uttryck, derivera, integrera, lösa differentialekvationer och så vidare. Om man förstår hur ett matematiskt problem kan lösas så kan man ta datorn till hjälp för att göra en del av de algebraiska manipulationerna. Saknar man däremot den grundläggande matematiska förståelsen hjälper inga datorkommandon, problemet förblir olösbart.

Föreliggande kompendium behandlar MATLABs tilläggsprogrammet i symbolisk matematik (Symbolic Math Toolbox) och är skrivet för matematiker, naturvetare, teknologer, lärare och andra intresserade som vill lära sig grunderna i datoralgebra. Områdena som behandlas är de man normalt stöter på under sina grundläggande matematikstudier. Dessa områden utgör bara en liten del av allt det som ingår i tilläggsprogrammet, och den intresserade läsaren som vill lära sig mer uppmanas därför att systematiskt gå igenom den dokumentation som finns tillgänglig via programmets hjälpfunktion. För att få ut så mycket som möjligt av studierna i datoralgebra förordas en aktiv inlärningsstil, där kompendiets exempel körs på dator och där man själv är nyfiken och undersöker vad som händer om man ändrar på saker och ting och hur kommandona beter sig på närliggande problem.

Kompendiet är ett komplement till boken *MATLAB-beräkningar inom teknik och naturvetenskap*, Studentlitteratur, 2010. I kompendiet refererar jag till denna bok som huvudboken (HB).

För de som inte har tillgång till MATLAB och tilläggsprogrammet i symbolisk matematik rekommenderar jag programpaketet Maxima. Maxima går under GNU Public License och kan laddas ner från

`http://maxima.sourceforge.net`

Maxima beskrivs ingående i min bok *Matmatik med datoralgebrasystem*, Studentlitteratur, 2008.

Materialet har under flera år utprovats i olika kurser på Malmö högskola. I samband med detta tackar jag alla studenter för värdefulla synpunkter och

förslag till förbättringar och förtydligande. Jag vill också rikta ett stort tack till Dr Stefan Gustafsson, min vän och rumskamrat på Malmö högskola, för hans engagemang i detta projekt.

Malmö i maj 2010

Per Jönsson

# Innehåll


**1 Symboliska uttryck** **7**
1.1 Om programmet för symbolisk matematik . . . . . 7
1.2 Hjälpkommandon och dokumentation . . . . . 7
1.3 Inledande exempel . . . . . 9
1.4 Symboliska objekt och variabler . . . . . 12
1.5 Räkning med symboliska objekt . . . . . 14
1.6 Substitution, omvandling till numerisk form . . . . . 16
1.7 Ordning mellan symboliska variabler . . . . . 18
1.8 Olika format av symboliska uttryck . . . . . 20
1.9 Vidare läsning . . . . . 21
1.10 Instuderingsfrågor . . . . . 21
1.11 Övningar . . . . . 22

**2 Algebra** **27**
2.1 Algebraiska manipulationer . . . . . 27
2.2 Förenkling av uttryck . . . . . 29
2.3 Ekvationer . . . . . 32
2.4 Vidare läsning . . . . . 35
2.5 Instuderingsfrågor . . . . . 35
2.6 Övningar . . . . . 35

**3 Linjär algebra** **39**
3.1 Linjära ekvationssystem . . . . . 39
3.2 Vektorer . . . . . 41
3.3 Skalär- och vektorprodukt . . . . . 42
3.4 Matriser . . . . . 48
3.5 Transponering . . . . . 49
3.6 Invers . . . . . 50
3.7 Determinant . . . . . 52
3.8 Egenvärden och egenvektorer . . . . . 53
3.9 Tillämpningsexempel; ortogonala polynom . . . . . 55
3.10 Vidare läsning . . . . . 56

3.11 Instuderingsfrågor . . . . . 56
3.12 Övningar . . . . . 57

**4 Gränsvärden, derivator och summor 61**
4.1 Funktioner . . . . . 61
4.2 Speciella matematiska funktioner . . . . . 63
4.3 Gränsvärden . . . . . 66
4.4 Derivator . . . . . 68
4.5 Taylorutvecklingar . . . . . 72
4.6 Summor . . . . . 74
4.7 Tillämpningsexempel; optimering . . . . . 76
4.8 Vidare läsning . . . . . 78
4.9 Instuderingsfrågor . . . . . 79
4.10 Övningar . . . . . 80

**5 Integraler och differentialekvationer 83**
5.1 Integraler . . . . . 83
5.2 Multipelintegraler . . . . . 86
5.3 Ordinära differentialekvationer . . . . . 88
5.4 System av ordinära differentialekvationer . . . . . 92
5.5 Vidare läsning . . . . . 95
5.6 Instuderingsfrågor . . . . . 95
5.7 Övningar . . . . . 95

**A Lösningar till valda övningar 101**

# Kapitel 1

# Symboliska uttryck

## 1.1 Om programmet för symbolisk matematik

Med tilläggsprogrammet för symbolisk matematik, vilket ingår som standard i studentversionen av MATLAB, får man tillgång till kommandon för algebraiska manipulationer, derivering, integrering, analytisk lösning av differentialekvationer med mera. Tilläggsprogrammet är baserat på kärnan av datoralgebraprogrammet Maple V. För att ta reda på om tilläggspaketet för symbolisk matematik (Symbolic Math Toolbox) är tillgängligt på den aktuella datorn ger man kommandot `ver` vid kommandopromptern. MATLAB svarar då med information om versionen av programmet och ger dessutom en lista på de tilläggsprogram som är installerade. Gå igenom listan och se om Symbolic Math Toolbox är med.

## 1.2 Hjälpkommandon och dokumentation

Genom att klicka på *hjälpknappen* i utvecklingsmiljöns verktygsfält (se figur 1.1HB)[1] kommer MATLABs hjälpfönster upp (se figur 1.4HB) Då man aktiverar boken med *Symbolic Math Toolbox* blir dokumentationen som hör till tilläggspaketet för symbolisk matematik tillgänglig. Dokumentationen innehåller, förutom en allmän beskrivning av tilläggspaketet, alla kommandon listade i kategorier och i alfabetisk ordning.

Precis som vanligt kan man också få hjälp via kommandofönstret. Då man skriver `help` följt av ett kommando, som måste föregås av `sym` för att markera att kommandot hör till tilläggspaketet för symbolisk matematik, får man en detaljerad beskrivning av kommandot och hur det används. Man kan också

[1] HB refererar till huvudboken: Per Jönsson, MATLAB-beräkningar inom teknik och naturvetenskap, Studentlitteratur 2010

få tillgång till Maples underliggande hjälptext genom att skriva `mhelp` följt av kommandot. En lista över alla kommandon som är tillgängliga i tillägsprogrammet fås genom att skriva `help symbolic`.

**Exempel 1.1.** För att få information om kommandot `diff` för symbolisk derivering skriver man

```
help sym/diff
```

Följande förklarande utskrift kommer då upp

```
DIFF   Differentiate.
DIFF(S) differentiates a symbolic expression S with
respect to its free variable as determined by FINDSYM.
DIFF(S,'v') or DIFF(S,sym('v')) differentiates S with
respect to v.
DIFF(S,n), for a positive integer n, differentiates S
n times.
DIFF(S,'v',n) and DIFF(S,n,'v') are also acceptable.

Examples;
   x = sym('x');
   t = sym('t');
   diff(sin(x^2)) is 2*cos(x^2)*x
   diff(t^6,6) is 720.

See also INT, JACOBIAN, FINDSYM.
```

□

**Exempel 1.2.** För att få Maples underliggande hjälptext för kommandot `diff` skriver man

```
mhelp diff
```

vilket ger

```
diff or Diff - Differentiation or Partial Differentiation

Calling Sequence:
     diff(a, x1, x2, ..., xn)
     Diff(a, x1, x2, ..., xn)
     diff(a, [x1, x2, ..., xn])
     Diff(a, [x1, x2, ..., xn])

Parameters:
     a               - an algebraic expression
```

```
        x1, x2, ..., xn - names

    Description:
    - diff computes the partial derivative of the expression
      a with respect to x1,x2, ..., xn, respectively. The
      most frequent use is diff(f(x),x), which computes the
      derivative of the function f(x) with respect to x.
```

Maples hjälptext är normalt väldigt omfattande och vi har här skurit ner den för att inte ta upp alltför mycket plats. □

## 1.3 Inledande exempel

Vi tar några inledande exempel för att visa på användningen av symbolisk matematik. Detaljer och ingående förklaringar till kommandona kommer i de följande avsnitten.

**Exempel 1.3.** För att utveckla uttrycket $(5x + 4)^4$ skriver man

```
syms x
expand((5*x + 4)^4)
```

Det första kommandot anger att $x$ är en symbolisk variabel. Det andra kommandot säger till MATLAB att utveckla det symboliska uttrycket. Vi får följande utskrift på skärmen

```
ans =
625*x^4+2000*x^3+2400*x^2+1280*x+256
```

Algebraiska manipulationer återkommer i kapitel 2. □

**Exempel 1.4.** Ekvationen $t^2 + 3t + 2 = 0$ löses genom kommandona

```
syms t
l = solve(t^2+3*t+2,t)
```

Det första kommandot anger att $t$ är en symbolisk variabel. Det andra kommandot är en uppmaning till MATLAB att lösa ekvationen $t^2 + 3t + 2 = 0$ med avseende på $t$ och spara lösningen i variabeln $l$, vilket ger

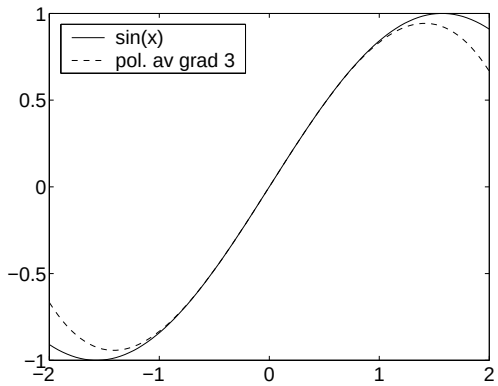
```
l =
 -1
 -2
```

Ekvationslösning behandlas i kapitel 2. □

**Exempel 1.5.** För att beräkna och plotta Taylorpolynomet av grad 3 av funktionen $f(x) = \sin(x)$ kring punkten $x = 0$ ger man kommandona

```
syms x
f = sin(x);
p = taylor(f,4,0);  % Obs 4 ger polynom av grad 3
xv = linspace(-2,2);
fv = subs(f,x,xv); pv = subs(p,x,xv);
plot(xv,fv,xv,pv,'--'), legend('sin(x)','pol. av grad 3')
```

Det första kommandot anger att $x$ är en symbolisk variabel. Andra kommandot definierar funktionen $f(x) = \sin(x)$. Tredje kommandot ger Taylorpolynomet och lagrar det i variabeln $p$. Fjärde kommandot genererar en vektor $xv$ med 100 värden jämnt fördelade mellan $-2$ och 2. Femte och sjätte kommandona ersätter den symboliska variabeln $x$ i uttrycken för $f$ och $p$ med vektorn $xv$ så att vi får vektorer $fv$ och $pv$ med funktionsvärden (se avsnitt 1.6). Dessa vektorer plottas mot $xv$ med hjälp av plotkommandot, vilket ger figur 1.1. □




Figur 1.1: *Funktionen* $f(x) = \sin(x)$ *och Taylorpolynomet av grad 3 kring 0.*

**Exempel 1.6.** Vi har en andra ordningens differentialekvation med begynnelsevillkor

$$\begin{cases} y'' + y' + y = e^{-x} \\ y(0) = 1, \quad y'(0) = -2. \end{cases}$$

Differentialekvationen löses analytiskt med kommandona

```
y = dsolve('D2y+Dy+y=exp(-x)','y(0)=1','Dy(0)=-2','x')
```

och MATLAB svarar

```
y =
-2/3*exp(-1/2*x)*sin(1/2*3^(1/2)*x)*3^(1/2)+exp(-x)
```

För att få en snyggare utskrift skriver vi

```
pretty(y)
```

vilket ger

```
                                   1/2      1/2
- 2/3 exp(- 1/2 x) sin(1/2 3    x) 3    + exp(-x)
```

Kommandon för att derivera, integrera och lösa differentialekvationer behandlas i kapitel 3 och 4. □

**Exempel 1.7.** Det är mycket enkelt att kombinera symboliska manipulationer och programmering. Legendrepolynomet av grad $k$ definieras av

$$P_k(t) = \frac{1}{2^k k!} \frac{d^k}{dt^k}(t^2 - 1)^k.$$

Legendrepolynomen är sinsemellan ortogonala med avseende på skalärprodukten

$$(P_k(t)|P_l(t)) = \int_{-1}^{1} P_k(t)P_l(t)\,dt$$

och dessutom gäller att

$$(P_k(t)|P_k(t)) = \frac{2}{2k+1}.$$

Följande program beräknar Legendrepolynomen $P_1(t), P_2(t), \ldots, P_5(t)$ och lagrar dem i en vektor $p$.

```
syms t
for k = 1:5
  p(k) = expand((1/(2^k*prod(1:k)))*diff((t^2-1)^k,t,k));
  disp(p(k))
end
```

MATLAB svarar

```
t
3/2*t^2-1/2
5/2*t^3-3/2*t
35/8*t^4-15/4*t^2+3/8
63/8*t^5-35/4*t^3+15/8*t
```

I de följande kapitlen återkommer vi flera gånger till programmering. □

## 1.4 Symboliska objekt och variabler

I tilläggsprogrammet för symbolisk matematik inför man symboliska objekt, eller symboliska uttryck, som en ny datatyp. Symboliska objekt (uttryck) är formellt sätt teckensträngar som skapas med kommandot `sym` eller som sätts samman av andra symboliska objekt. En variabel som tilldelas värdet av ett symboliskt objekt kallas en symbolisk variabel. Symboliska objekt kan manipuleras enligt kända algebraiska lagar.

| | |
|---|---|
| `a = sym('k')` | variabeln $a$ tilldelas värdet av det symboliska objektet $k$. |
| `a = sym('k','real')` | variabeln $a$ tilldelas värdet av det symboliska objektet $k$, som förutsätts vara reellt. |
| `a = sym('k','positive')` | variabeln $a$ tilldelas värdet av det symboliska objektet $k$, som förutsätts vara reellt och positivt. |
| `a = sym('k','unreal')` | variabeln $a$ tilldelas värdet av det symboliska objektet $k$, som inte förutsätts vara reell eller positivt. |
| `syms a b c` | variablerna $a, b, c$ tilldelas värdena av de symboliska objekten $a, b, c$. |
| `syms a b c real` | variablerna $a, b, c$ tilldelas värdena av de symboliska objekten $a, b, c$, vilka förutsätts vara reella. |
| `syms a b c positive` | variablerna $a, b, c$ tilldelas värdena av de symboliska objekten $a, b, c$, vilka förutsätts vara reella och positiva. |
| `syms a b c unreal` | variablerna $a, b, c$ tilldelas värdena av de symboliska objekten $a, b, c$, som inte förutsätts vara reella eller positiva. |
| `whos` | ger en detaljerad lista över alla variabler, både numeriska och symboliska. |
| `syms` | ger en lista över de symboliska variablerna. |

**Exempel 1.8.**

(a) Kommandot

```
a = sym('a')
```

tilldelar variabeln $a$ värdet av det symboliska objektet $a$. Variabeln $a$ blir alltså en symbolisk variabel. Om vi vid något senare tillfälle tilldelar $a$ värdet

av ett tal, t.ex. $a = 1.1$, så övergår $a$ till att bli en vanlig numerisk variabel.

(b) Genom kommandona

```
a = sym('a','positive'), b = sym('b','positive')
```

tilldelas variablerna $a$ och $b$ värdena av de symboliska objekten $a$ respektive $b$, vilka förutsätts vara reella och positiva. Ovanstående tilldelningssats kan också skrivas

```
syms a b positive
```

Då det sista skrivsättet är kortare är det ofta att föredra.

(c) Kommandot

```
a = sym('a','unreal')
```

tilldelar variabeln $a$ värdet av det symboliska objektet $a$, som nu inte förutsätts vara reellt och positivt. Kommandot rensar variabeln och objektet $a$ från tidigare förutsättningar.

(d) Kommandot

```
a = sym('sqrt(5)+1')
```

tilldelar variabeln $a$ värdet av det symboliska objektet $\sqrt{5} + 1$. □

**Exempel 1.9.**

(a) Då vi ger kommandot

```
f = sym('a*x^2 + b*x + c')
```

tilldelas variabeln $f$ värdet av det symboliska objektet $ax^2 + bx + c$. Samma symboliska objekt kan också skapas genom kommandona

```
syms a b c x
f = a*x^2 + b*x + c
```

I detta fallet skapas först de symboliska objekten $a, b, c, x$, vilka lagras i motsvarande variabler. De grundläggande symboliska objekten sätts sedan samman till $ax^2 + bx + c$. I det följande kommer vi ofta att använda det senare sättet att skapa mera komplicerade symboliska objekt.

(b) Då vi ger kommandot

```
f = sym('sqrt(2)*sin(a*x+b)')
```

tilldelas variabeln $f$ värdet av det symboliska objektet $\sqrt{2}\sin(ax+b)$. Samma symboliska objekt kan också skapas genom kommandona

```
syms a b x
k = sym('sqrt(2)')
f = k*sin(a*x+b)
```

□

**Exempel 1.10.** Vi har variablerna `a = 3`, `z = 1-3i`, `x = sym('sqrt(5)')` och `y = sym('y','real')`

(a) Kommandot

```
who
```

ger då följande utskrift

```
Name      Size          Bytes  Class

a         1x1               8  double array
x         1x1             138  sym object
y         1x1             126  sym object
z         1x1              16  double array (complex)

Grand total is 12 elements using 288 bytes
```

Förutom namnen får vi storleken, upptaget minnesutrymme i bytes samt typen av de definierade variablerna.

(b) Kommandot

```
syms
```

ger endast de symboliska variablerna

```
'x'    'y'
```

(c) För att ta bort alla variabler, numeriska som symboliska, ger man kommandot

```
clear all
```

Gamla variabler som ligger kvar i minnet orsakar ofta fel. Det är därför en god regel att börja varje ny arbetsuppgift med att rensa variablerna. □

## 1.5 Räkning med symboliska objekt

Symboliska objekt och uttryck visas vänsterjusterade i kommandofönstret medan numeriska objekt visas indragna. Symboliska objekt hanteras också

annorlunda vid beräkningar än MATLABs vanliga numeriska objekt.

**Exempel 1.11.**

(a) Om vi tar sinus av $\pi/3$

```
x = sin(pi/3)
```

svarar MATLAB med

```
x =
   0.86602540378444
```

Vi har fått svaret som ett decimaltal. Om vi definierar $\pi/3$ som ett symboliskt objekt genom kommandot `sym('pi/3')` och tar sinus

```
x = sin(sym('pi/3'))
```

svarar MATLAB med

```
x =
1/2*3^(1/2)
```

Vid tilldelningen blir $x$ en symbolisk variabel som innehåller svaret i den exakta formen. För att få en snyggare utskrift kan man skriva `pretty(x)` vilket ger

```
      1/2
1/2 3
```

För att omvandla det exakta svaret till ett decimaltal i dubbel precision skriver vi `double(x)` varvid MATLAB returnerar 0.86602540378444.

(b) Då vi skriver

```
x = 2; y = 3;
z = 1/x + 1/y
```

svarar MATLAB med

```
z =
   0.83333333333333
```

Vi har fått svaret som ett decimaltal. Om vi definierar 2 och 3 som symboliska objekt och gör samma operationer

```
x = sym('2'); y = sym('3');
z = 1/x + 1/y
```

svarar MATLAB med

```
z =
5/6
```

Vid tilldelningen blir $z$ en symbolisk variabel som innehåller svaret på exakt form. □

## 1.6 Substitution, omvandling till numerisk form

Vid arbetet med symboliska uttryck har man ofta användning för substitutionskommandot `subs`. Kommandot kan bland annat användas för att omvandla symboliska uttryck till numerisk form.

| | |
|---|---|
| `subs(s)` | ersätter symboliska variabler i uttrycket $s$ med motsvarande variabelvärden. |
| `subs(s,old,new)` | ersätter den symboliska variabeln *old* i uttrycket $s$ med *new*, där *new* är en symbolisk eller numerisk variabel eller ett uttryck. Fungerar även för ersättning av flera symboliska variabler, vilka då skall skrivas inom klammerparentes. |

**Exempel 1.12.**

(a) Vi definierar ett symboliskt uttryck som lagras i $y$

```
syms a c t
y = c*exp(-a*t);
```

Genom kommandona

```
c = 4; a = 2;
subs(y)
```

ersätts variablerna $c$ och $a$ i uttrycket för $y$ med variabelvärdena 4 och 2 vilket ger

```
ans =
4*exp(-2*t)
```

(b) Vi definierar ett symboliskt uttryck

```
syms a b d t
y = a*sin(b*t+d);
```

Då vi ger kommandot

```
subs(y,{a b d},{5 2 -1})
```

ersätts variablerna $a, b, d$ i uttrycket för $y$ med värdena $5, 2, -1$, vilket ger

```
ans =
5*sin(2*t-1)
```

(c) Vi definierar ett symboliskt uttryck

```
syms a b t
y = a*t^2 + b
```

Genom kommandona

```
T = 0:5;
Y = subs(y,t,T)
```

ersätts variabeln $t$ i uttrycket för $y$ med vektorn $T = (0, 1, 2, 3, 4, 5)$. Ersättningen sker elementvis och vi får utskriften

```
Y =
[      b,    a+b,  4*a+b,  9*a+b, 16*a+b, 25*a+b]
```

(d) Vi definierar ett symboliskt uttryck

```
syms t x
f = t^2*exp(-t);
```

Genom kommandot

```
g = subs(f,t,sin(x))
```

ersätts variabeln $t$ i uttrycket för $f$ med uttrycket $\sin(x)$, vilket ger

```
g =
sin(x)^2*exp(-sin(x))
```

Substitutionskommandot är alltså mångsidigt och vi kan ersätta symboliska variabler med andra symboliska uttryck, men också med tal och vektorer. □

Som ett sista exempel ska vi använda substitutionskommandot för att omvandla ett symboliskt uttryck till en numerisk vektor, vilken sedan används för att generera en plot.

**Exempel 1.13.**

(a) Vi har ett symboliskt uttryck

```
syms x
f = x*sin(x);
```

Genom kommandona

```
xv = linspace(0,20,500);
fv = subs(f,x,xv);
plot(xv,fv)
xlabel('x'), ylabel('y')
```

ersätts $x$ i det symboliska uttrycket $f$ med den numeriska vektorn *xv* varvid vi får en numerisk vektor *fv*. Observera att ersättningen sker elementvis och att vi inte behöver använda `.*` i uttrycket för $f$. Vektorn *fv* plottas sedan mot *xv* med hjälp av det vanliga plotkommandot. Plotten visas till vänster i figur 1.2.

(b) Vi har ett symboliskt uttryck

```
syms a b c x
f = a*sin(b*x+c);
```

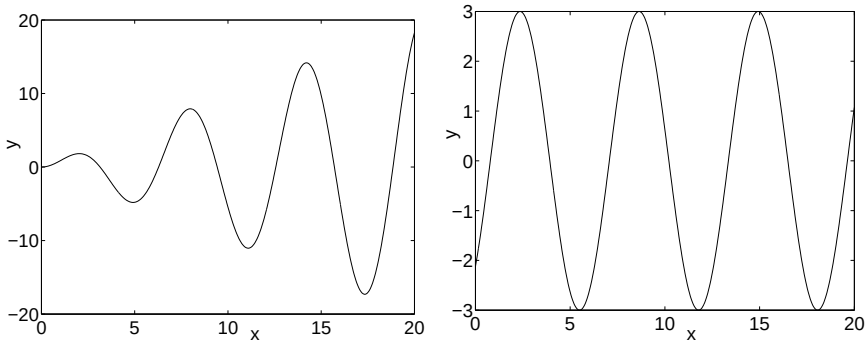
Genom kommandona

```
f = subs(f,{a b c},{3 1 -pi/4});
xv = linspace(0,20,500);
fv = subs(f,x,xv);
plot(xv,fv), xlabel('x'), ylabel('y')
```

ersätts $a, b, c$ i det symboliska uttrycket $f$ med värdena $3, 1, -\pi/4$ varvid vi får ett nytt symboliskt uttryck *f*. I *f* ersätts sedan $x$ med den numeriska vektorn *xv*, vilket ger en numerisk vektor *fv*. Vektorn *fv* plottas slutligen mot *xv*. Den genererade plotten visas till höger i figur 1.2. Observera att ersättningen av variabler sker i två steg, då substitutionskommandot inte kan hantera fall av typen

```
subs(s,{a b},{c d})
```

där $c$ är ett tal och $d$ är en vektor. □




Figur 1.2: *Plottning av symboliska uttryck.*

## 1.7 Ordning mellan symboliska variabler

När man manipulerar matematiska uttryck är variabeln med avseende på vilken man gör operationen ofta känd genom sammanhanget. Om vi till exempel skall derivera det symboliska uttrycket $y = \sin(ax + b)$ så utför vi vanligtvis derivationen med avseende på $x$ och betraktar $a$ och $b$ som konstanter eller parametrar. I symbolisk matematik bör man vara lite försiktigare och noggrant specificera med avseende på vilken variabel operationer skall utföras. Då denna variabel inte specificeras använder MATLAB den så kallade

”findsymregeln” för att avgöra frågan. Enligt findsymregeln görs operationer i första hand med avseende på variabeln $x$. Om $x$ inte finns med i det symboliska uttrycket så utförs operationer på den variabel som ligger närmast $x$ i alfabetet. Om två variabler ligger lika nära $x$ så utförs operationer på den variabel som kommer efter $x$. Variabeln som bestäms enligt findsymregeln kallas defaultvariabeln. Defaultvariabeln skrivs ut med kommandot `findsym`.

| | |
|---|---|
| `findsym(s,1)` | ger variabeln i uttrycket $s$ med avseende på vilken operationer kommer att utföras (defaultvariabeln). |
| `findsym(s)` | listar alla symboliska variabler i uttrycket $s$ i alfabetisk ordning. |

**Exempel 1.14.**

(a) Vi har ett symboliskt uttryck

```
syms a b c x
y = a*sin(b*x+c)
```

Enligt findsymregeln kommer operationer på uttrycket att göras med avseende på variabeln $x$, som alltså är defaultvariabeln.

(b) Vi har ett symboliskt uttryck

```
syms t z
f = (t+z)^3
```

Enligt findsymregeln kommer operationer på uttrycket att göras med avseende på variabeln $z$, som är defaultvariabeln. Detta bekräftas av kommandot

```
findsym(f,1)
```

som ger

```
ans =
z
```

(c) Vi har ett symboliskt uttryck

```
syms s t u
f = 3*s^2 + 2*s*t - 4*s*u^2
```

Då vi skriver

```
findsym(f), findsym(f,1)
```

får vi svaret

```
ans =
s, t, u
ans =
u
```

Uttrycket innehåller de tre symboliska variablerna $s, t, u$. Av dessa är $u$ defaultvariabeln. □

## 1.8 Olika format av symboliska uttryck

För att få symboliska uttryck på en mera lättläst form kan man använda kommandot `pretty`. Genom kommandot `latex` kan man även få uttrycken på LaTeX-form (jfr avsnitt 5.5HB).

| | |
|---|---|
| `pretty(s)` | skriver ett symboliskt uttryck $s$ på lättläst form. |
| `vectorize(s)` | skriver ett symboliskt uttryck $s$ på vektorform, dvs med en punkt framför operatorn för multiplikation, division och upphöjt till (jfr avsnitt 4.11HB). |
| `latex(s)` | ger LaTeX-formen av det symboliska uttrycket $s$. Teckensträngen med uttrycket kan sättas in i ett LaTeX-dokument men även i PowerPoint via TexPoint paketet. |

**Exempel 1.15.** Vi har det symboliska uttrycket

```
syms a x
s = sqrt(a)*x*sin(pi*x)/(x+1)
```

(a) Kommandot

```
pretty(s)
```

ger utskriften

```
 1/2
a    x sin(pi x)
----------------
     x + 1
```

(b) Då vi skriver

```
latex(s)
```

returnerar MATLAB uttrycket $s$ på LaTeX-form

```
ans =
{\frac {\sqrt {a}x\sin(\pi \,x)}{x+1}}}
```

vilket efter insättning i ett LaTeX-dokument ger utskriften

$$\frac{\sqrt{a}x\sin(\pi\,x)}{x+1}$$

(c) Vi har följande symboliska uttryck

```
syms x
f = x^2*sin(2*x)/cos(x)
```

Då vi skriver

```
fv = vectorize(f)
```

returnerar MATLAB uttrycket $f$ på vektorform med operatorerna för upphöjt till, multiplikation och division skrivna som `.^`, `.*` och `./`

```
fv =
x.^2.*sin(2.*x)./cos(x)
```

Uttrycket i vektorform kan till exempel kopieras och klistras in i en M-fil avsedd för numeriska beräkningar. □

## 1.9 Vidare läsning

Jönsson, P. (2008), Matematik med datoralgebrasystem, Studentlitteratur, ISBN 978-91-44-05250-2

En omfattande beskrivning av symbolisk matematik med Maxima, ett program under GNU Public License. Maxima finns för flera operativsystem och kan laddas ner från `http://maxima.sourceforge.net/`

Andersson, G. (1997), Tillämpad matematik med Maple, Studentlitteratur, ISBN 91-44-00503-2

Boken ger grunderna i symbolisk matematik med Maple.

The Mathworks egen dokumentation om tilläggspaketet för symbolisk matematik kan laddas ner från The MathWorks hemsida `www.mathworks.com`.

## 1.10 Instuderingsfrågor

1. Hur skriver man för att få information om ett kommando som hör till tilläggsprogrammet för symbolisk matematik?

2. Hur får man fram en lista över alla kommandon som finns tillgängliga i tilläggspaketet för symbolisk matematik?

3. Vad är ett symboliskt objekt och hur skapas det?

4. Vad är en symbolisk variabel?

5. Vad händer med en symbolisk variabel då den tilldelas ett numeriskt värde?

6. Hur hanteras symboliska objekt jämfört med numeriska objekt? Ge några exempel.

7. Redogör för substitutionskommandot.

8. Då en variabel ersätts med en vektor med hjälp av substitutionskommandot sker ersättningen elementvis. Förklara innebörden av detta. Ge något konkret exempel.

9. Vi har ett symboliskt uttryck

```
syms x, f = x^2*log(1+x)
```

   Hur skriver man för att ersätta $x$ i det symboliska uttrycket $f$ med en numerisk vektor $xv$ med 100 värden jämnt fördelade mellan 0 och 1?

10. Redogör för "findsymregeln".

11. Vad menas med defaultvariabeln?

12. Hur skriver man för att få ett symboliskt uttryck $s$ på lättläst form?

13. Hur kan man få ett symboliskt uttryck $s$ på vektorform? När är detta användbart?

## 1.11 Övningar

1. Ge kommandot `help symbolic` så att du får upp en lista över kommandon. Bekanta dig med listan så att du får en överblick över tilläggsprogrammet för symbolisk matematik.

2. Använd `help` och `mhelp` på kommandot `expand`.

3. Utveckla följande uttryck (se exempel 1.3).

   (a) $(x^3 + 2x^2 - x - 5)^3$ (b) $\sin(x + y)$ (c) $e^{(a+b)^2}$

4. Lös följande ekvationer (se exempel 1.4).

   (a) $x^2 - 4x + 3 = 0$. (b) $x^3 - x^2 - \frac{1}{4}x + \frac{1}{4} = 0$.

5. Betrakta funktionen $f(x) = x^2 e^x$.

   (a) Bestäm tangenten (Taylorpolynomet av grad 1) till $f(x)$ i punkten $x = 1$ (se exempel 1.5).

   (b) Plotta den ursprungliga funktionen och tangenten i samma figur för $0 \leq x \leq 2$.

6. Vi har en differentialekvation

   $$y'(t) = 100 - y(t), \qquad y(0) = 20.$$

   (a) Lös differentialekvationen analytiskt (se exempel 1.6).

   (b) Plotta lösningen för $0 \leq t \leq 10$.

7. Skapa följande symboliska uttryck.

   (a) $2/3$ (b) $\pi/2$ (c) $\sqrt{10} + 1$ (d) $e^2$

   (e) $\ln(2)$ (f) $a\sin(bx + c)$ (g) $\dfrac{(ax+b)}{(cx+d)}$

8. Använd `help` på kommandona `abs`, `conj`, `real` och `imag`.

9. Skapa det symboliska uttrycket $z = x + iy$ där $x$ och $y$ är reella tal. Ge kommandona `abs(z)`, `conj(z)`, `real(z)`, `imag(z)`.

10. Skapa det symboliska uttrycket $z = x + iy$ där $x$ och $y$ inte förutsätts vara reella. Ge kommandona `abs(z)`, `conj(z)`, `real(z)`, `imag(z)`. Jämför resultaten med det du fick från föregående övning. Förklara skillnaden.

11. Beräkna följande uttryck både numeriskt och exakt.

    (a) $\sin(\pi/12)$ (b) $\frac{1}{3} + \frac{1}{2} + \frac{4}{5}$ (c) $\dfrac{1+i}{3-2i}$

12. Vi har ett symboliskt uttryck

    ```
    syms a b c x
    y = a*exp(b*x+c)
    ```

    Använd substitutionskommandot och

    (a) ersätt $a, b, c$ med värdena $-1, 2, 0$

    (b) ersätt $x$ med $x^2 + x + 1$.

13. Ersätt $x$ med vektorn $(0, 1, 2, 3, 4, 5)$, $k$ med 3 och $m$ med $-1$ i uttrycket

```
syms k m x
y = k*x + m
```

14. Vi har ett symboliskt uttryck

```
syms a b x
y = a*x/(1 + b*x)
```

Ersätt $a$ med 3 och $b$ med 2 och plotta uttrycket för $0 \leq x \leq 5$.

1. Bestäm defaultvariabeln i följande uttryck

(a)
```
syms a b x
y = log(a*x) + b
```

(b)
```
syms R I
U = R*I
```

(c)
```
syms a b n R T V
P = n*R*T/(V-n*b) - a*(n/V)^2
```

16. Betrakta det symboliska uttrycket

```
syms x
y = (3 + 2*sin(x))/(x^2 + x*sqrt(x)*cos(x))
```

(a) Skriv uttrycket på lättläst form.

(b) Skriv uttrycket på vektorform. Klipp ut och klistra in uttrycket i en M-fil som används för att plotta uttrycket i intervallet $[0, 10]$. Välj lämplig skala på $y$-axeln.

17. Betrakta programmet i exempel 1.7.

(a) Komplettera programmet så att de fem Legendrepolynomen plottas i intervallet $[-1, 1]$.

(b) Komplettera programmet så att det beräknar och skriver ut alla skalärprodukterna

$$(P_k(t)|P_l(t)) = \int_{-1}^{1} P_k(t)P_l(t)\,dt, \quad k = 1, 2, \ldots, 5, \quad l = 1, 2, \ldots, 5.$$

Ledning: gör `help sym/int` för att få reda på hur man använder kommandot för symbolisk integrering.

18. Legendrepolynomen uppfyller rekursionsformeln

$$kP_k(t) = (2k-1)\,t\,P_{k-1}(t) - (k-1)P_{k-2}(t).$$

Skriv ett program som använder rekursionsformeln och genererar Legendrepolynom upp till $k = 5$. Använd $P_0(t) = 1$ och $P_1(t) = t$. Du kan behöva använda kommandot `simplify` på de genererade polynomen.

# Kapitel 2

# Algebra

I det här kapitlet behandlas grundläggande algebraiska manipulationer och förenklingar. Även ekvationslösning tas upp. Det som görs här ligger till grund för mycket av det som kommer i de följande kapitlen.

## 2.1 Algebraiska manipulationer

Symboliska uttryck kan manipuleras enligt kända algebraiska lagar och regler. Följande kommandon är användbara.

| | |
|---|---|
| `expand(s)` | skriver ett symboliskt uttryck $s$ som en summa av produkter. Används oftast på polynom, men fungerar också på trigonometriska, exponentiella och logaritmiska uttryck. |
| `collect(s,v)` | skriver det symboliska uttrycket $s$ i termer av potenser i variabeln $v$. Då $v$ utelämnas får vi uttrycket som potenser av defaultvariabeln (jfr 1.7). |
| `factor(s)` | faktoriserar det symboliska uttrycket $s$. Om uttrycket är ett heltal fås en uppdelning i primfaktorer. |

**Exempel 2.1.**

(a) Polynomet $(2x+3)^3$ utvecklas genom

```
syms x
y = expand((2*x + 3)^3);
pretty(y)
```

och vi får svaret

```
   3       2
8 x  + 36 x  + 54 x + 27
```

(b) Det går också bra att lagra ett uttryck i en variabel och sedan ge kommandot `expand`. Så ger

```
syms x;
f = (x + 1)^2;
expand(f)
```

svaret

```
ans =
x^2+2*x+1
```

(c) Det trigonometriska uttrycket $\sin(at + b)$ utvecklas genom

```
syms a b t;
f = sin(a*t+b);
expand(f)
```

och vi får svaret

```
ans =
sin(a*t)*cos(b)+cos(a*t)*sin(b)
```

(d) Vi definierar ett symboliskt uttryck $(ax + b)^2 + (a + b)x$

```
syms a b x;
f = (a*x+b)^2 + (a+b)*x;
```

Kommandot

```
pretty(collect(f))
```

ger uttrycket i termer av potenser i defaultvariabeln, vilken är $x$ i detta fallet, och MATLAB svarar

```
 2  2                        2
a  x  + (2 b a + a + b) x + b
```

För att få uttrycket i termer av potenser i variabeln $a$ skriver vi

```
pretty(collect(f,a))
```

vilket ger

```
 2  2                    2
a  x  + (2 b x + x) a + b  + b x
```

(e) Följande kommandon faktoriserar $t^3 - 2t^2 - 5t + 6$

```
syms t
f = t^3-2*t^2-5*t+6; factor(f)
```

MATLAB svarar

```
ans =
(t-1)*(t-3)*(t+2)
```

(f) För att faktorisera heltalet 7475 skriver vi

```
n = sym('7475'); factor(n)
```

MATLAB svarar då

```
ans =
(5)^2*(13)*(23)
```

Kommandot `factor` fungerar även för vanliga numeriska heltal. □

## 2.2 Förenkling av uttryck

En viktig del av matematiken är att förenkla sina uttryck. Problemet är att det inte finns en entydig definition av vad som menas med enkelt. Det som är enkelt och funktionellt i ett sammanhang är inte nödvändigtvis enkelt i ett annat. Vilken form som är önskvärd för ett uttryck beror i hög grad på omständigheterna. I MATLAB finns två grundläggande förenklingskommandon.

| | |
|---|---|
| `simplify(s)` | förenklar det symboliska uttrycket $s$ med hjälp av en rad olika manipulationer som bland annat utnyttjar identiteter för trigonometriska, logaritmiska och exponentiella uttryck. |
| `simple(s)` | använder `simplify` , `collect`, `factor` och andra förenklingsregler i olika kombinationer. Skriver ut alla uttryck som är kortare, räknat i antalet tecken, än det ursprungliga uttrycket. |
| `f = simple(s)` | använder `simplify` , `collect`, `factor` och andra förenklingsregler i olika kombinationer. Det kortaste uttrycket lagras i $f$. |

Många av uttrycken som man får från förenklingskommandona är svårlästa, och det kan vara vettigt att använda kommandot `pretty` för att få snyggare utskrifter.

**Exempel 2.2.**

(a) För att förenkla uttrycket

$$\left(\frac{1}{a^3}+\frac{6}{a^2}+\frac{12}{a}+8\right)^{1/3}$$

skriver vi

```
syms a
f = simplify((1/a^3+6/a^2+12/a+8)^(1/3));
pretty(f)
```

MATLAB svarar då

```
/         3\1/3
|(2 a + 1) |
|----------|
|     3    |
\    a     /
```

Kommandona

```
syms a
f = simple((1/a^3+6/a^2+12/a+8)^(1/3));
pretty(f)
```

ger istället

```
2 a + 1
-------
   a
```

(b) För att förenkla uttrycket

$$\cos^2(x) - 2\sin^2(x)$$

skriver vi

```
syms x
f = simple(cos(x)^2-2*sin(x)^2)
```

vilket ger

```
f =
3*cos(x)^2-2
```

För att få fram alla förenklingsförslag utelämnar vi variabeln till vänster om kommandot `simple` och skriver

```
syms x
simple(cos(x)^2-2*sin(x)^2)
```

MATLAB svarar nu med utskriften

```
simplify:
3*cos(x)^2-2

radsimp:
cos(x)^2-2*sin(x)^2

combine(trig):
3/2*cos(2*x)-1/2

factor:
cos(x)^2-2*sin(x)^2

expand:
cos(x)^2-2*sin(x)^2

combine:
3/2*cos(2*x)-1/2

convert(exp):
(1/2*exp(i*x)+1/2/exp(i*x))^2+1/2*(exp(i*x)-1/exp(i*x))^2

convert(sincos):
cos(x)^2-2*sin(x)^2

convert(tan):
(1-tan(1/2*x)^2)^2/(1+tan(1/2*x)^2)^2-8*tan(1/2*x)^2 ...
/(1+tan(1/2*x)^2)^2

collect(x):
cos(x)^2-2*sin(x)^2

mwcos2sin:
1-3*sin(x)^2

ans =
3*cos(x)^2-2
```

Vi ser att det kortaste uttrycket, räknat i antalet tecken, är $3\cos^2(x) - 2$. Det ur många aspekter enklaste uttrycket är dock $(3/2)\cos(2x) - 1/2$, vilket fås då man använder Maples kommando `combine`. □

## 2.3 Ekvationer

Det går att lösa vissa enklare ekvationer och ekvationssystem. Förutom reella lösningar får man i många fall även komplexa lösningar.

| | |
|---|---|
| `solve(s,v)` | löser ekvationen $s = 0$, där $s$ är ett symboliskt uttryck, med avseende på variabeln $v$. Då $v$ utelämnas löses ekvationen med avseende på defaultvariabeln. |
| `solve(s1,s2,..,sn ,v1,v2,..,vn)` | löser ekvationssystemet $s_1 = 0, s_2 = 0, .., s_n = 0$ med avseende på variablerna $v_1, v_2, \ldots, v_n$. Då variablerna $v_1, v_2, \ldots, v_n$ utelämnas löses systemet med avseende på defaultvariablerna. |

Om exakta symboliska lösningar inte kan bestämmas returneras i en del fall motsvarande numeriska lösning. Observera att man vid lösning av trigonometriska ekvationer inte får alla lösningar utan man måste själv fundera ut och lägga till periodiciteten.

**Exempel 2.3.**

(a) För att lösa andragradsekvationen $x^2 = 2x - 2$ och spara lösningen i variabeln $x$ flyttar vi om så att vi får $x^2 - 2x + 2 = 0$ och skriver

```
syms x
x = solve(x^2-2*x+2,x)
```

varvid MATLAB svarar

```
x =
 1+i
 1-i
```

Vi har fått två komplexa lösningar $1 + i$ och $1 - i$ som är lagrade i $x(1)$ respektive $x(2)$.

(b) För att lösa ut $\kappa$ ur uttrycket

$$\frac{T_1}{T_2} = \left(\frac{P_1}{P_2}\right)^{\frac{\kappa - 1}{\kappa}}$$

ger man kommandona

```
syms T1 T2 P1 P2 k
k = solve(T1/T2-(P1/P2)^((k-1)/k),k);
pretty(k)
```

varvid MATLAB svarar

```
                P1
          log(----)
                P2
- ----------------------
          P1          T1
  -log(----) + log(----)
          P2          T2
```

(c) Det går bra att lagra ett uttryck i en variabel och sedan ge kommandot `solve`. Ekvationen $\sin(x) + \cos(x) = 1$ löses till exempel med

```
syms x
s = sin(x)+cos(x)-1;
solve(s,x)
```

och vi får svaret

```
ans =
 1/2*pi
      0
```

Från våra kunskaper om de trigonometriska funktionerna får vi själv dra slutsatsen att vi ska addera multiplar av $2\pi$ för att få samtliga lösningar.

(d) Ekvationen $\cos(x) = x$ har ingen analytisk lösning. Då vi skriver

```
syms x
solve(cos(x)-x,x)
```

får vi istället det numeriska värdet

```
ans =
.7390851332151606416553120876738 7
```

□

**Exempel 2.4.**

(a) För att lösa ekvationssystemet

$$\begin{cases} ax + y = 3 \\ x - y = -1 \end{cases}$$

med avseende på $x$ och $y$ och spara lösningarna i $x$ och $y$ skriver vi

```
syms a x y
[x,y] = solve(a*x+y-3,x-y+1,x,y)
```

MATLAB svarar då (utskriften har editerats för att ta mindre plats)

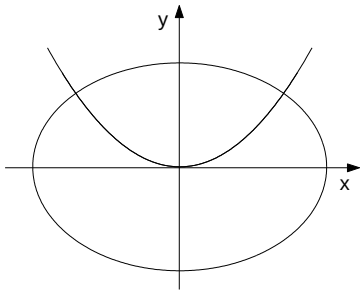
```
x =                 y =
2/(a+1)             (a+3)/(a+1)
```

Från det givna uttrycket ser vi att lösning saknas precis då $a = -1$.

(b) Ekvationssystemet

$$\begin{cases} x^2 = y \\ x^2 + 2y^2 = 1 \end{cases}$$

kan geometrisk tolkas som skärningspunkterna mellan ellipsen $x^2 + 2y^2 = 1$ och parabeln $y = x^2$ (se figur 2.1).




Figur 2.1: *Skärning mellan ellips och parabel.*

Ekvationssystemet löses med

```
syms x y
[x,y] = solve(x^2-y,x^2+2*y^2-1,x,y)
```

och svaret blir

```
x =                         y =
[              i]           [  -1]
[             -i]           [  -1]
[  1/2*2^(1/2)]             [ 1/2]
[ -1/2*2^(1/2)]             [ 1/2]
```

Det finns fyra lösningar $(x, y)$. Två lösningar är reella och två är komplexa.

(c) Då vi har ett system av ekvationer och endast en variabel i vänsterledet fås en struktur som innehåller lösningen. Betrakta systemet

$$\begin{cases} x^2 + xy + y &= 3 \\ x^2 - 4x &= -3 \end{cases}$$

Kommandona

```
syms x y
l = solve(x^2+x*y+y-3,x^2-4*x+3,x,y)
```

ger utskriften

```
l =
    x: [2x1 sym]
    y: [2x1 sym]
```

Lösningen är lagrad i strukturen $l$ som har en $x$- och en $y$-komponent. För att få fram värdena i strukturen skriver man `l.x` respektive `l.y`. □

## 2.4 Vidare läsning

Jönsson, P. (2008), Matematik med datoralgebrasystem, Studentlitteratur, ISBN 978-91-44-05250-2

Förenklingar, algebraiska uttryck och funktioner behandlas i kapitel 4 och 6.

Andersson, G. (1997), Tillämpad matematik med Maple, Studentlitteratur, ISBN 91-44-00503-2

Boken ger grunderna i symbolisk matematik med Maple. Förenklingar och ekvationer behandlas i kapitel 3.

## 2.5 Instuderingsfrågor

1. Vilka är de tre kommandona för grundläggande algebraiska manipulationer? Beskriv var och en av dem.
2. Vilka två kommandon finns för förenkling av uttryck?
3. Låt $s$ vara ett symboliskt uttryck. Vad får vi då vi skriver `simple(s)` respektive `f = simple(s)`
4. Hur löser man ekvationer och ekvationssystem?
5. På vilken form får man lösningarna till ett ekvationssystem om man endast skriver en variabel i vänsterledet?

## 2.6 Övningar

1. Skriv om följande uttryck som ett polynom i $x$

$$\left(x-\frac{1}{a}-\frac{1}{b}\right)\left(x-\frac{1}{b}-\frac{1}{c}\right)\left(x-\frac{1}{c}-\frac{1}{a}\right)$$

2. Multiplicera de två polynomen

$$f(x) = x^4 + 4x^3 - 17x^2 + 20x - 12 \quad \text{och} \quad g(x) = x^2 - 5x + 3$$

och förenkla resultatet.

3. Utveckla

(a) $(x+3)(x-3) - (x+3)^2$ (b) $(3x+5)^2 - (3x-5)^2$

(c) $\tan(3x)$ (d) $\sin(3x)$

4. Fibonaccitalen $0, 1, 1, 2, 3, 5, 8, 13, 21, \ldots$ ges av rekursionsföljden

$$f_k = f_{k-1} + f_{k-2}$$

där $f_0 = 0$, $f_1 = 1$. Man kan visa att Fibonaccitalen också ges av formeln

$$f_k = \frac{1}{\sqrt{5}} \left[ \left( \frac{1+\sqrt{5}}{2} \right)^k - \left( \frac{1-\sqrt{5}}{2} \right)^k \right].$$

Skriv ett program som visar att rekursionsföljden och formeln ger samma tal för $k = 0, 1, 2, \ldots, 20$.

5. Faktorisera följande uttryck

(a) $x^4 + 4x^3 + 6x^2 + 4x + 1$ (b) $x^2 - 7x + xy - 7y$

(c) $a^3 + b^3 + c^3 - 3abc$ (d) $a^6 - a^4 + a^2 - 1$

6. Förenkla

(a) $\dfrac{2}{3x+9} + \dfrac{x}{x^2-9} - \dfrac{1}{2x-6}$

(b) $\dfrac{3x-y}{x^2-2xy+y^2} - \dfrac{2}{x-y} - \dfrac{2y}{(x-y)^2}$

(c) $\dfrac{\frac{1}{x+1} + \frac{1}{x-1}}{\frac{1}{x-1} - \frac{1}{x+1}}$

7. Skriv om bråken så att nämnaren blir rotfri

(a) $\dfrac{3+\sqrt{5}}{2+\sqrt{5}}$ (b) $\dfrac{1+2\sqrt{2}}{3-\sqrt{2}}$ (c) $\dfrac{1}{\sqrt{13}+\sqrt{11}}$

8. Lös följande ekvationer

(a) $\dfrac{1}{x-1} - \dfrac{1}{x-2} = \dfrac{1}{x-3} - \dfrac{1}{x-4}$ (b) $\sqrt{x+2} = x$

(c) $x^3 + 10x^2 + 24x = 0$ (d) $\ln(3^x + 3^{x+1}) = 1$

(e) $\sin x \cos x = \dfrac{1}{4}$

9. I den så kallade skalmodellen för atomkärnan förekommer följande uttryck för den elektriska potentialen som funktion av radien $r$

$$V(r) = -\frac{V_0}{1 + \exp(\frac{r-R}{a})}, \quad r > 0.$$

Här är $V_0, R$ och $a$ positiva konstanter.

(a) Beräkna $r_1$ och $r_2$ sådana att $V(r_1) = -0.1V_0$ och $V(r_2) = -0.9V_0$.

(b) Sätt upp ett uttryck för den så kallade skaltjockleken $r_1 - r_2$.

(c) Skissa $V(r)$ i intervallet $[0, 10]$ för $V_0 = 1$, $R = 5$ och $a = 0.5$.

10. Vi har två massor enligt figuren nedan.

Vilken blir hastigheten hos $m$ respektive $M$ efter stöten som antas vara rak, central och elastisk. Ledning: både rörelsemängd och kinetisk energi skall bevaras vid stöten.

# Kapitel 3

# Linjär algebra

## 3.1 Linjära ekvationssystem

Linjära ekvationssystem är vanligt förekommande i olika tillämpningar. Som ett exempel på linjära ekvationsysystem kan vi ta

$$\begin{cases} 2x + 9y + 2z &= 9 \\ 3x + 7y - 3z &= -3 \\ 4x - 5y + 4z &= -11 \end{cases}.$$

I detta fallet är antalet ekvationer lika med antalet obektanta. Man kan även ha underbestämda system, där antalet obekanta är fler än antalet ekvationer, och överbestämda ekvationssystem, där antalet ekvationer är fler än antalet obekanta. Kommandot `solve` hanterar alla linjära ekvationssystem och man behöver inte förutsätta att antalet ekvationer är lika med antalet obekanta.

**Exempel 3.1.**

(a) Vi har ett $2 \times 2$ ekvationssystem

$$\begin{cases} ax + 2y &= 8 \\ 5x + 8y &= b \end{cases}.$$

Kommandona

```
syms a b x y
[x,y] = solve(a*x+2*y-8,5*x+8*y-b,x,y)
```

ger utskriften (vilken har editerats för att ta mindre plats)

```
x =                          y =
-(-32+b)/(-5+4*a)             1/2*(-40+a*b)/(-5+4*a)
```

Nämnarna i lösningen blir 0 då $a = 5/4$. För att se hur detta fallet hanteras skriver vi

```
syms a b x y
a = sym('5/4');
[x,y] = solve(a*x+2*y-8,5*x+8*y-b,x,y)
```

MATLAB returnerar

```
x =                    y =
[ empty sym ]               []
```

vilket visar att lösning saknas i detta fallet.

(b) Vi har ett $3 \times 3$ ekvationssystem

$$\begin{cases} 3x_1 - 2x_2 + 4x_3 &= 8 \\ 5x_1 + 8x_2 - 6x_3 &= -5 \\ 9x_1 - 2x_2 + 7x_3 &= -17 \end{cases}$$

Systemet löses genom

```
syms x1 x2 x3
s1 = 3*x1-2*x2+4*x3-8;
s2 = 5*x1+8*x2-6*x3+5;
s3 = 9*x1-2*x2+7*x3+17;
[x1,x2,x3] = solve(s1,s2,s3,x1,x2,x3)
```

vilket ger lösningen

```
x1 =          x2 =          x3 =
-331/9        1283/18       587/9
```

(c) Betrakta följande underbestämda system

$$\begin{cases} x - y - 2z &= -8 \\ 3x - 3y - 4z &= -20 \end{cases}$$

För att lösa systemet skriver vi

```
syms x y z
s1 = x-y-2*z+8;
s2 = 3*x-3*y-4*z+20;
[x,y,z] = solve(s1,s2,x,y,z)
```

MATLAB svarar

```
x =        y =        z =
-4+y       y          2
```

Detta ska tolkas som att lösningen beror av en parameter $y$ som kan anta vilket värde som helst (det finns alltså oändligt många lösningar). Det spelar ingen roll vilken symbol vi använder för parametern. Väljer vi t.ex. att beteckna parametern med bokstaven $t$ så ser lösningen ut på följande sätt

$$x = -4 + t, \quad y = t, \quad z = 2, \quad t \in \mathbb{R}.$$

Vi säger att lösningen har en frihetsgrad. □

## 3.2 Vektorer

En vektor är en samling ordnade reella eller komplexa tal

$$v = (v_1, v_2, \ldots, v_n).$$

I MATLAB fås vektorer genom att skriva elementen inom hakparentes

```
v = [v1 v2 ... vn]
```

eller genom att tilldela elementen ett och ett

```
v(1) = v1, v(2) = v2, ... v(n) = vn
```

**Exempel 3.2.**

(a) För att generera vektorn $v = (a, b, c)$ kan man skriva

```
syms a b c
v = [a b c]
```

eller

```
syms a b c
v(1) = a, v(2) = b, v(3) = c
```

(b) Numeriska vektorer kan omvandlas till symbolisk form. Vektorn

```
v = [0.25 -2.3 1.5]
```

fås på symbolisk form genom

```
v = sym(v)
```

och MATLAB svarar

```
v =
[    1/4, -23/10,     3/2]
```

I den symboliska formen har alla decimaltal ersatts med rationella tal. □

## 3.3 Skalär- och vektorprodukt

Skalär- och vektorprodukt är viktiga begrepp inom vektorgeometrin. Låt $u = (u_1, u_2, u_3)$ och $v = (v_1, v_2, v_3)$ vara två vektorer i rummet. Med den skalära produkten $u \cdot v$ av vektorerna menas det reella talet

$$|u||v| \cos\theta,$$

där $\theta$ är den minsta vinkeln mellan vektorerna (se vänstra delen av figur 3.1). Två vektorer är vinkelräta (ortogonala) precis då skalärprodukten är lika med noll. I en ortonormerad bas (ON-bas) får vi följande enkla uttryck i koordinaterna

$$u \cdot v = u_1 v_1 + u_2 v_2 + u_3 v_3.$$

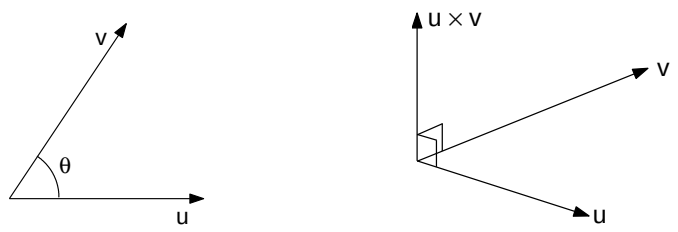
Genom att sätta $u = v$ fås ett uttryck för längden av $u$ i kvadrat

$$|u|^2 = u \cdot u = u_1^2 + u_2^2 + u_3^2.$$

Vektorprodukten $u \times v$ är definierad som vektorn med längd

$$|u||v| \sin\theta$$

som är ortogonal mot både $u$ och $v$ och som ligger orienterad så att $u$, $v$, $u \times v$ är positivt orienterade (se högra delen av figur 3.1).




Figur 3.1: *Skalärprodukt $u \cdot v$ och vektorprodukt $u \times v$.*

I en högerorienterad ortonormerad bas (HON-bas) får vi följande uttryck i koordinaterna

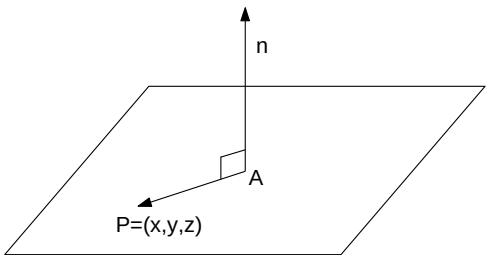
$$u \times v = (u_2 v_3 - u_3 v_2, u_3 v_1 - u_1 v_3, u_1 v_2 - u_2 v_1).$$

I MATLAB finns kommandona `dot` och `cross` som fungerar för både numeriska och symboliska vektorer. Kommandot för skalärprodukt, `dot`, kan användas för vektorer med godtycklig dimension. Det kan också användas också för komplexa vektorer, varvid man får $\overline{u}_1 v_1 + \overline{u}_2 v_2 + \ldots + \overline{u}_n v_n$. Normalt arbetar

vi dock med reella vektorer och man skall därför inte glömma att definiera motsvarande symboliska objekt som reella. Kommandot för vektorprodukt, `cross`, är endast definierat för reella vektorer i rummet.

| | |
|---|---|
| `dot(u,v)` | beräknar skalärprodukten $\overline{u}_1 v_1 + \overline{u}_2 v_2 + \ldots + \overline{u}_n v_n$ mellan två vektorer $u = (u_1, u_2, \ldots, u_n)$ och $v = (v_1, v_2, \ldots, v_n)$. |
| `cross(u,v)` | beräknar vektorprodukten $u \times v = (u_2 v_3 - u_3 v_2, u_3 v_1 - u_1 v_3, u_1 v_2 - u_2 v_1)$ mellan $u = (u_1, u_2, u_3)$ och $v = (v_1, v_2, v_3)$. |

**Exempel 3.3.** Ett plan bestäms av en normalvektor $n$ och en punkt $A$ (se figur 3.2). Punkten $P = (x, y, z)$ ligger i planet precis då vektorn $\overline{AP}$ är vinkelrät mot normalvektorn, dvs då $n \cdot \overline{AP} = 0$. Detta definierar planets ekvation.




Figur 3.2: *Plan bestämt av en normalvektor $n$ och en punkt $A$.*

Vi betraktar fallet då vi har en normalvektor $n = (1, -2, 3)$ och en punkt $A = (1, 0, 4)$. Vänsterledet i planets ekvation beräknas då genom

```
syms x y z
n = [1 -2 3]; n = sym(n);
A = [1 0 4];  A = sym(A);
P = [x y z];  AP = P-A;
dot(n,AP)
```

MATLAB svarar

```
ans =
x-13-2*y+3*z
```

Planets ekvation är alltså

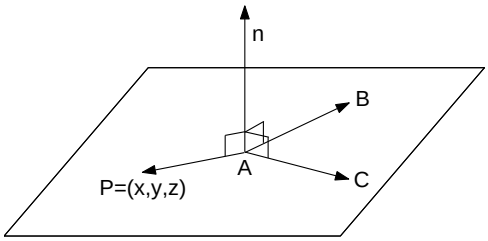
$$x - 2y + 3z - 13 = 0.$$

□

**Exempel 3.4.** Tre punkter $A$, $B$ och $C$ definierar ett plan. En normalvektor till planet ges av

$$n = \overline{AB} \times \overline{AC}.$$

En punkt $P = (x, y, z)$ ligger i planet precis då $n \cdot \overline{AP} = 0$. Det sista villkoret ger planets ekvation.




Figur 3.3: *Ett plan bestämt av tre punkter $A$, $B$ och $C$.*

Vi har tre punkter $A = (0, 1, 2)$, $B = (3, 2, 1)$ och $C = (4, -1, 0)$. Följande kommandon bestämmer vänsterledet i planets ekvation

```
syms x y z
A = [0 1 2]; A = sym(A);
B = [3 2 1]; B = sym(B);
C = [4 -1 0]; C = sym(C);
AB = B-A; AC = C-A;
n = cross(AB,AC);
P = [x y z];
AP = P - A;
dot(n,AP)
```

Svaret blir

```
ans =
-4*x+2*y+18-10*z
```

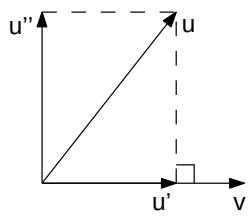
Planets ekvation blir alltså

$$-4x + 2y - 10z + 18 = 0.$$

En kontroll visar att de tre punkterna $A$, $B$, $C$ uppfyller ekvationen och alltså ligger i planet. □

**Exempel 3.5.** Låt $u'$ vara projektionen av vektorn $u$ på $v$ (se figur till höger). Enligt projektionsformeln är

$$u' = \frac{u \cdot v}{|v|^2}\, v.$$




När väl projektionen är bestämd fås det ortogonala komplementet som $u'' = u - u'$.

Vi betraktar vektorerna $u = (1, -2, 3)$ och $v = (-2, 5, 1)$. Projektionen $u'$ och det ortogonala komplementet $u''$ beräknas genom

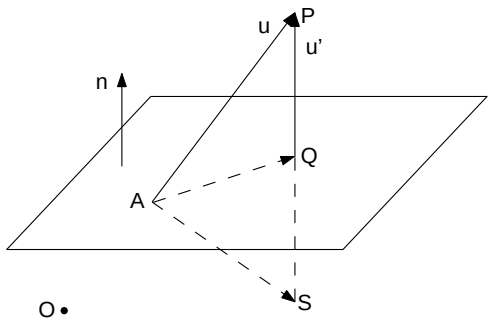
```
u = [1 -2 3]; u = sym(u);
v = [-2 5 1]; v = sym(v);
up = (dot(u,v)/dot(v,v))*v
uk = u - up
```

vilket ger

```
up =
[   3/5,  -3/2, -3/10]
uk =
[   2/5,  -1/2, 33/10]
```

En kontroll visar att *up* och *uk* är ortogonala. □

**Exempel 3.6.** Ett plan är givet av normalvektorn $n$ och punkten $A$. Vi ska bestämma projektionen $Q$ och spegelpunkten $S$ av punkten $P$ (se figur 3.4).




Figur 3.4: *Projektion och spegling av en punkt i ett plan.*

Vektorn $u'$ fås genom att projicera $u = \overline{AP}$ på normalvektorn $n$. Punkterna $Q$ och $S$ fås sedan som

$$Q = \overline{OP} - u' \quad \text{och} \quad S = \overline{OP} - 2u'.$$

Vi betraktar $P = (4, 2, -1)$ och planet givet av ekvationen

$$2x + 3y - z + 5 = 0.$$

Planet har normalvektorn $n = (2, 3, -1)$. Genom insättning ser man att till exempel punkten $A = (0, 0, 5)$ ligger i planet. Följande kommandon ger punkterna $Q$ och $S$

```
n = [2 3 -1]; n = sym(n);
P = [4 2 -1]; P = sym(P);
A = [0 0 5]; A = sym(A);
u = P-A;
up = (dot(u,n)/dot(n,n))*n;
Q = P-up
S = P-2*up
```

Vi får svaret

```
Q =
[   8/7, -16/7,   3/7]
S =
[ -12/7, -46/7,  13/7]
```

□

**Exempel 3.7.** Linjärt oberoende vektorer kan ortogonaliseras i en så kallad Gram-Schmidt process. Om $v_1, v_2, \ldots, v_n$ är linjärt oberoende så gäller att

$$\begin{aligned}
v_1' &= v_1 \\
v_2' &= v_2 - \frac{v_2 \cdot v_1'}{v_1' \cdot v_1'} v_1' \\
v_3' &= v_3 - \frac{v_3 \cdot v_1'}{v_1' \cdot v_1'} v_1' - \frac{v_3 \cdot v_2'}{v_2' \cdot v_2'} v_2' \\
&\vdots \\
v_n' &= v_n - \frac{v_n \cdot v_1'}{v_1' \cdot v_1'} v_1' - \frac{v_n \cdot v_2'}{v_2' \cdot v_2'} v_2' - \ldots - \frac{v_n \cdot v_{n-1}'}{v_{n-1}' \cdot v_{n-1}'} v_{n-1}'
\end{aligned}$$

är sinsemellan ortgonala vektorer. Om de ortogonaliserade vektorerna normaliseras fås en uppsättning ortonormala vektorer.

Följande M-fil ortogonaliserar kolonnvektorerna $v_1, v_2, \ldots, v_n$ lagrade i matrisen $V$. Efter ortogonaliseringen normaliseras vektorerna och lagras som kolonner i matrisen $U$.

```
function U = gramschmidt(V)
[m,n] = size(V);
U(:,1) = V(:,1);
for i = 2:n
  U(:,i) = V(:,i);
  for j = 1:i-1
    A = dot(V(:,i),U(:,j));
    B = dot(U(:,j),U(:,j));
    U(:,i) = U(:,i) - (A/B)*U(:,j);
  end
end
for i = 1:n
  U(:,i) = U(:,i)/sqrt(dot(U(:,i),U(:,i)));
end
```

Vi testar ortonormaliseringen på vektorerna

$$v_1 = \begin{pmatrix} 1 \\ 2 \\ 0 \\ 2 \end{pmatrix}, \quad v_2 = \begin{pmatrix} 1 \\ -2 \\ 1 \\ 0 \end{pmatrix}, \quad v_3 = \begin{pmatrix} 0 \\ 2 \\ -1 \\ 2 \end{pmatrix}$$

Då vi ger kommandona

```
v1 = [1 ; 2 ; 0 ; 2];
v2 = [1 ; -2 ; 1 ; 0];
v3 = [0 ; 2 ; -1 ; 2];
V = [v1 v2 v3]
V = sym(V);
U = gramschmidt(V); pretty(U)
```

svarar MATLAB med

```
[                1/2                1/2]
[1/3     4/15 5         - 2/15 5    ]
[                                    ]
[                1/2               1/2 ]
[2/3    - 4/15 5        - 1/5 5     ]
[                                    ]
[               1/2                1/2]
[ 0      1/5 5          - 4/15 5    ]
[                                    ]
[                1/2               1/2 ]
[2/3     2/15 5          4/15 5     ]
```

En kontroll visar att kolonnvektorerna i $U$ är ömsesidigt ortogonala och att de alla har längden ett. □

## 3.4 Matriser

En matris är ett rektangulärt schema av tal. I MATLAB fås matriser genom att skriva elementen radvis inom hakparentes

```
A = [a11 a12 ... a1n
     a21 a22 ... a2n
             ...
     am1 am2     amn]
```

Man kan även ge alla element på samma kommandorad, där semikolon skiljer raderna åt

```
A = [a11 a12 ... a1n ; a21 a22 ... a2n ; ...]
```

Matriser kan multipliceras med tal (skalärer). Matriser med samma storlek kan också adderas och subtraheras. Dessa operationer sker elementvis. Under vissa förutsättningar kan man också definiera produkten av två matriser. I MATLAB har vi kommandona `+`, `-`, `*` för addition, subtraktion och matrismultiplikation. Notera att multiplikation betecknas med `*` och *inte* med punktstjärna `.*`, som ger elementvis multiplikation (jfr avsnitt 4.11HB).

**Exempel 3.8.**

(a) Matriserna

$$A = \begin{pmatrix} a & b & c \\ -1 & a & b \end{pmatrix} \quad \text{och} \quad B = \begin{pmatrix} a \\ b \end{pmatrix}$$

definieras genom kommandona

```
syms a b c
A = [a b c ; -1 a b]; B = [a ; b];
```

(b) Numeriska matriser kan omvandlas till symbolisk form. Matrisen

```
A = [0.25 -2.1 3.2
     0.8   5.2 4.0]
```

fås på symbolisk form genom

```
A = sym(A)
```

och MATLAB svarar

```
A =
[    1/4, -21/10,   16/5]
[    4/5,   26/5,      4]
```

I den symboliska formen har alla decimaltal ersatts med rationella tal. □

**Exempel 3.9.** Vi har två matriser $A$ och $B$

```
syms a b
A = [a 1/4 ; 0 -1];
B = [-2 -8 ; b 2];
```

(a) Matriserna adderas genom kommandot

```
C = A + B
```

och vi får resultatet

```
C =
[   a-2, -31/4]
[     b,     1]
```

(b) Multiplikation $AB$ fås genom

```
C = A*B
```

vilket ger

```
C =
[ -2*a+1/4*b,   -8*a+1/2]
[         -b,         -2]
```

□

## 3.5 Transponering

Omkastning av rader och kolonner i en matris ger den transponerade matrisen $A^T$. För komplexa matriser inför man också adjunkten, vilken fås genom transponering och komplexkonjugering. Adjunkten betecknas ibland $A^*$. För att transponera en matris $A$ i MATLAB skriver man `A.'`. Adjunkten fås genom `A'`.

**Exempel 3.10.**

(a) Vi har en reell matris $A$

```
syms a b c real
A = [ a b 3
     -c 4 5];
```

Transponatet $A^T$ fås genom

```
A.'
```

vilket ger

```
ans =
[  a, -c]
[  b,  4]
[  3,  5]
```

(b) Betrakta följande komplexa matris

```
syms x y real
A = [x+y*i 3+5*i
     2-2*i x-y*i]
```

För att få transponatet $A^T$ och adjunkten $A^*$ ger vi kommandona

```
B = A.'
C = A'
```

MATLAB svarar

```
B =
[ x+i*y, 2-2*i]
[ 3+5*i, x-i*y]
C =
[ x-i*y, 2+2*i]
[ 3-5*i, x+i*y]
```

□

## 3.6 Invers

Med enhetsmatrisen $I$ av ordning $n$ menar man en $n \times n$ matris med ettor på diagonalen och nollor för övrigt. Låt $A$ vara en $n \times n$ matris. Om det finns en matris $B$ sådan att

$$AB = I \quad \text{och} \quad BA = I$$

så säger man att $A$ är inverterbar med den inversa matrisen $B$. För en inverterbar matris $A$ betecknas den inversa matrisen $B$ med $A^{-1}$. Kvadratiska matriser som saknar invers kallas singulära.

I MATLAB beräknas den inversa matrisen till $A$ genom kommandot `inv(A)`. I de fall matrisen $A$ är singulär skriver MATLAB ut ett varningsmeddelande.

**Exempel 3.11.** Betrakta matrisen

```
syms a b c
A = [a b
     c 1]
```

Inversen fås genom kommandot

```
B = inv(A)
```

och resultatet blir

```
B =
[  1/(a-b*c), -b/(a-b*c)]
[ -c/(a-b*c),  a/(a-b*c)]
```

Man kan kontrollera att $B$ verkligen är en invers genom att utföra multiplikationerna $AB$ och $BA$ och se att resultatet blir enhetsmatrisen. □

**Exempel 3.12.** Den så kallade Givensmatrisen

$$G = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix},$$

beskriver en linjär avbildning i form av en rotation med vinkeln $\theta$ radianer (jfr avsnitt 9.8). Avbildningsmatrisen ges av kommandona

```
syms theta real
G = [cos(theta) -sin(theta)
     sin(theta)  cos(theta)]
```

(a) Produkten av två avbildningsmatriser beräknas genom

```
G2 = G*G
```

och MATLAB svarar

```
G2 =
[ cos(theta)^2-sin(theta)^2,  -2*cos(theta)*sin(theta)]
[   2*cos(theta)*sin(theta), cos(theta)^2-sin(theta)^2]
```

Förenkling av uttrycket

```
G2 = simple(G2)
```

```
G2 =
[  cos(2*theta), -sin(2*theta)]
[  sin(2*theta),  cos(2*theta)]
```

vilket är förenligt med tolkningen att två rotationer med vinkeln $\theta$, beskrivet av produkten $G^2$ av matriserna, är lika med en rotation med vinkeln $2\theta$.

(b) Inversen $G^{-1}$ av avbildningsmatrisen fås genom

```
Ginv = inv(G);
```

Inversen förenklas med kommandot

```
Ginv = simple(Ginv)
```

vilket ger

```
Ginv =
[  cos(theta),  sin(theta)]
[ -sin(theta),  cos(theta)]
```

Detta kan, med utnyttjande av att cosinusfunktionen är jämn och sinusfunktionen är udda, också skrivas

$$G^{-1} = \begin{pmatrix} \cos(-\theta) & -\sin(-\theta) \\ \sin(-\theta) & \cos(-\theta) \end{pmatrix},$$

vilket är konsistent med tolkningen att inversen (omvändningen) av en rotation med vinkeln $\theta$ är en rotation med den motsatta vinkeln $-\theta$. □

## 3.7 Determinant

Determinanten till en matris $A$ betecknas med $\det(A)$ eller $|A|$. I MATLAB beräknas determinanten till den kvadratiska matrisen $A$ genom kommandot `det(A)`. Följande exempel illustrerar användandet av kommandot.

**Exempel 3.13.** Betrakta matrisen

$$A = \begin{pmatrix} a & 3 & 4 \\ 1 & 2 & 2 \\ 5 & 0 & 3 \end{pmatrix}$$

Bestäm för vilka $a$ matrisen är singulär.

Matrisen och determinanten fås genom kommandona

```
syms a
A = [a 3 4 ; 1 2 2 ; 5 0 3];
det(A)
```

MATLAB svarar

```
ans =
6*a-19
```

Matrisen är singulär då determinanten är noll, dvs. precis då $a = 19/6$. □

**Exempel 3.14.** Matrisen

$$V = \begin{pmatrix} 1 & t_1 & t_1^2 & t_1^3 \\ 1 & t_2 & t_2^2 & t_2^3 \\ 1 & t_3 & t_3^2 & t_3^3 \\ 1 & t_4 & t_4^2 & t_4^3 \end{pmatrix}$$

kallas Vandermondematrisen av ordning 4. Matrisen och motsvarande determinant beräknas med kommandona

```
syms t1 t2 t3 t4
V = [1 t1 t1^2 t1^3
     1 t2 t2^2 t2^3
     1 t3 t3^2 t3^3
     1 t4 t4^2 t4^3]
detV = simple(det(V))
```

MATLAB ger svaret

```
detV =
(t4-t2)*(-t2+t3)*(t3-t4)*(-t2+t1)*(t1-t4)*(t1-t3)
```

□

## 3.8 Egenvärden och egenvektorer

Låt $A$ vara en $n \times n$ kvadratisk matris. En vektor $x \neq 0$ sådan att

$$Ax = \lambda x$$

för något tal $\lambda$, kallas en egenvektor till $A$ med egenvärdet $\lambda$. I MATLAB finns följande kommandon för att bestämma egenvärden och egenvektorer.

| | |
|---|---|
| `eig(A)` | ger en kolonnvektor som innehåller egenvärdena till matrisen $A$. |
| `[X,D] = eig(A)` | ger en diagonalmatris $D$ som innehåller egenvärdena till $A$ och en matris $X$ med motsvarande egenvektorer. Om $X$ är av samma storlek som $A$ så har man en full uppsättning av linjärt oberoende egenvektorer. |
| `[X,D,P] = eig(A)` | som ovan fast nu returneras också en vektor $P$ med index. Längden av $P$ är lika med antalet linjärt oberoende egenvektorer. Indexen talar om till vilket egenvärde de olika kolonnerna i $X$ hör. |

**Exempel 3.15.** Vi har en matris

```
A = [1 -1  4
     3  2 -1
     2  1 -1];
A = sym(A)
```

(a) Egenvärdena fås genom

```
lambda = eig(A)
```

vilket ger utskriften

```
lambda =
  1
 -2
  3
```

(b) För att få både egenvärdena och egenvektorerna skriver vi

```
[X,D] = eig(A)
```

och MATLAB svarar

```
X =                    D =
[  1, -1,  1]          [  3,  0,  0]
[  2,  1, -4]          [  0, -2,  0]
[  1,  1, -1]          [  0,  0,  1]
```

□

**Exempel 3.16.** Vi har en matris

```
A = [ 1 0 1
     -1 2 1
     -1 1 2];
A = sym(A)
```

(a) Egenvärdena och egenvektorerna beräknas genom

```
[X,D] = eig(A)
```

vilket ger

```
X =                D =
[ 1, 1]            [ 1, 0, 0]
[ 1, 1]            [ 0, 2, 0]
[ 0, 1]            [ 0, 0, 2]
```

De två egenvektorerna bildar ingen bas i $\mathbb{R}^3$.

(b) Då vi skriver

```
[X,D,P] = eig(A)
```

blir svaret

```
X =            D =              P =
[ 1, 1]        [ 1, 0, 0]       1     2
[ 1, 1]        [ 0, 2, 0]
[ 0, 1]        [ 0, 0, 2]
```

Vektorn $P$ indikerar att den första egenvektorn hör till det första egenvärdet medan den andra egenvektorn hör till det andra egenvärdet. □

## 3.9 Tillämpningsexempel; ortogonala polynom

Mängden av alla polynom $p(t) = a_0 + a_1 t + a_2 t^2 + \ldots + a_n t^n$ på intervallet $[-1, 1]$ bildar ett linjärt rum. Med skalärprodukten

$$(p(t)|q(t)) = \int_{-1}^{1} p(t)q(t)\,dt$$

blir rummet ett Euklidiskt rum. För att konstruera en ortonormerad bas i detta rum tillämpar vi Gram-Schmidts metod på basen $1, t, t^2, \ldots, t^n$. Följande M-fil ortogonaliserar $1, t, t^2, \ldots, t^n$ lagrade i vektorn $V$. Efter ortogonaliseringen normaliseras polynomen och lagras i vektorn $U$. Enda skillnaden jämfört med exempel 3.7 är att skalärprodukter nu beräknas som integraler med kommandot `int(p,q,-1,1)` (se avsnitt 5.1 för mera information om integrationskommandot).

```
function U = gramschmidtpolynom(V)
U(1) = V(1);
n = length(V)
for i = 2:n
  U(i) = V(i);
  for j = 1:i-1
    A = int(V(i)*U(j),-1,1);
    B = int(U(j)*U(j),-1,1);
    U(i) = U(i) - (A/B)*U(j);
  end
end
% normera
for i = 1:n
   U(i) = U(i)/sqrt((int(U(i)^2,-1,1)));
end
```

Vi testar att beräkna en ortonormerad bas för det Euklidiska rummet.

```
syms t
V = [1 t t^2 t^3 t^4];
U = gramschmidtpolynom(V);
pretty(U)
```

vilket ger

```
[     1/2          1/2         2          1/2
[1/2 2    , 1/2 t 6    , 3/4 (t  - 1/3) 10
[

        3            1/2   105   4                2   1/2 ]
, 5/4 (t  - 3/5 t) 14    , --- (t  + 3/35 - 6/7 t ) 2    ]
                           16                             ]
```

De genererande polynomen är, så när som på en multiplikativ konstant, lika med Legendrepolynomen. Ortogonala polynom är av stor betydelse inom approximationsteorin och de behandlas vidare i övningarna.

## 3.10 Vidare läsning

Jönsson, P. (2008), Matematik med datoralgebrasystem, Studentlitteratur, ISBN 978-91-44-05250-2

Linjär algebra gås igenom i kapitel 7.

Andersson, G. (1997), Tillämpad matematik med Maple, Studentlitteratur, ISBN 91-44-00503-2

Boken ger grunderna i symbolisk matematik med Maple. Linjär algebra behandlas i kapitel 7.

Anton, H. and Busby, R.C. (2003), Contemporary linear algebra, Wiley, ISBN 0-471-16362-7

En modern framställning med en mängd olika tillämpningar.

Andersson, K.G. (2000), Linjär algebra, Studentlitteratur, ISBN-9144016085

Mycket klar framställning av den linjära algebran. Innehåller ett avsnitt om Gram-Schmidts ortogonaliseringsmetod.

## 3.11 Instuderingsfrågor

1. Vad menas med att ett linjärt ekvationssystem är underbestämt? Ge ett exempel på ett sådant system.

2. Vad menas med att ett linjärt ekvationssystem är överbestämt?

3. Vad menas med att en lösning till ett linjärt ekvationssystem har en frihetsgrad?

4. Vad är en vektor och hur skrivs en sådan i MATLAB?

5. Hur skriver man för att omvandla den numeriska vektorn `v = [0.6 0.8 1.5]` till symbolisk form?

6. Vilken är den geometriska definitionen av skalärprodukt?

7. Förutsatt att vi jobbar i en ON-bas, hur ser skalärprodukten ut i koordinatform?

8. Vilken är den geometriska definitionen av vektorprodukt?

9. Förutsatt att vi jobbar i en HON-bas, hur ser vektorprodukten ut i koordinatform?

10. Vilka är kommandona för skalär- respektive vektorprodukt?

11. Du känner en normalvektor $n$ och en punkt $A$ i ett plan. Hur kan planets ekvation då uttryckas?

12. Hur lyder projektionsformeln?

13. Vad är det för skillnad på transponering $A^T$ och adjungering $A^*$? Hur skrivs de två operationerna i MATLAB?

14. Vilka egenskaper har en inverterbar matris $A$?

15. Vilka egenskaper har determinanten till en matris $A$? Vad är kommandot för att beräkna determinanten?

16. Vad menas med egenvärde resp. egenvektor till en kvadratisk matris $A$?

## 3.12 Övningar

1. Bestäm samtliga lösningar till följande system

$$\begin{cases} x_1 - x_2 + 2x_3 + x_4 + 3x_5 = 1 \\ 2x_1 + x_3 + 2x_4 + 4x_5 = -2 \\ x_1 + 2x_2 - 2x_3 + 3x_5 = -3 \end{cases}.$$

2. För vilka värden på konstanten $a$ har ekvationssystemet en entydig lösning? När finns oändligt många lösningar och när saknas lösning?

$$\begin{cases} x_1 + x_2 + x_3 = 0 \\ x_1 + 2x_2 + ax_3 = 1 \\ x_1 + ax_2 + 2x_3 = -1 \end{cases}.$$

3. Bestäm ekvationen för planet definierad av normalvektorn $n = (1, 2, -3)$ och punkten $A = (-2, -1, 4)$.

4. Bestäm projektionen $Q$ och spegelpunkten $S$ av punkten $P = (1, -2, 3)$ i planet definierat av de tre punkterna $(-1, -3, -4)$, $(-5, -5, 1)$ och $(3, 1, 2)$.

5. Beräkna det kortaste avståndet från punkten $(1, 2, 3)$ till den räta linjen

$$\begin{cases} x = 1 - t \\ y = -4 + 2t \\ z = 3 - t \end{cases}.$$

6. Bestäm kortaste avståndet mellan linjerna

$$(x, y, z) = (1, 2, 3) + t\,(0, 1, 1) \quad \text{och} \quad (x, y, z) = (1, 1, 1) + t\,(2, 3, 1).$$

7. Låt $\mathbf{u} = (u_1, u_2, u_3)$, $\mathbf{v} = (v_1, v_2, v_3)$ och $\mathbf{w} = (w_1, w_2, w_3)$ vara vektorer. Visa att

$$(\mathbf{u} \times \mathbf{v}) \times \mathbf{w} = (\mathbf{u} \cdot \mathbf{w})\,\mathbf{v} - (\mathbf{v} \cdot \mathbf{w})\,\mathbf{u}.$$

8. Visa att om $\mathbf{u} + \mathbf{v} + \mathbf{w} = \mathbf{0}$ så är $\mathbf{u} \times \mathbf{v} = \mathbf{v} \times \mathbf{w} = \mathbf{w} \times \mathbf{u}$. Ledning: använd substitutionskommandot och ersätt $u_1$ med $-v_1 - w_1$, $u_2$ med $-v_2 - w_2$ och $u_3$ med $-v_3 - w_3$.

9. Vektorerna $(1, 0, 2)$, $(-1, 2, 3)$ och $(4, 2, -1)$ är linjärt oberoende och bildar en bas för $\mathbb{R}^3$. Använd Gram-Schmidtprogrammet i exempel 3.7 för att generera en ON-bas.

10. Låt

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \quad \text{och} \quad B = \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix}.$$

Verifiera att följande likheter gäller

(a) $\det(AB) = \det(A)\det(B)$ (b) $(AB)^{-1} = B^{-1}A^{-1}$.

(c) $\det(A^T) = \det(A)$ (d) $(AB)^T = B^T A^T$

11. Testa dig fram och försök hitta en formel som du tror gäller för $A^n$ om

$$A = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}.$$

12. Hilbertmatriser $H$ har element $h_{ij} = 1/(i+j-1)$ och är kända för att vara mycket illa konditionerade (jfr exempel 9.9HB). Bestäm den exakta inversen till Hilbertmatrisen av ordning $n = 5$. Approximera den exakta Hilbertmatrisen med en matris $H_0$ där element $h_{ij} = 1/(i+j-1)$ har omvandlats till reella tal med sex decimaler. Invertera $H_0$ numeriskt och jämför resultatet med den exakta inversen till $H$.

13. Visa att egenvärdena till matrisen

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

ges av formeln $\lambda = [(a+d) \pm \sqrt{(a-d)^2 + 4bc}]/2$.

14. Bestäm egenvärdena och egenvektorerna till matrisen

$$A = \begin{pmatrix} 1 & a+bi \\ a-bi & 1 \end{pmatrix}$$

där $a$ och $b$ är reella tal.

15. I avsnitt 3.9 införde vi skalärprodukten mellan två polynom. Mera generellt kan vi definiera skalärprodukten genom

$$(p(t)|q(t)) = \int_{-1}^{1} p(t)q(t)w(t)\,dt,$$

där $w(t)$ är en icke-negativ viktfunktion. Låt $w(t) = (1-t^2)^{-1/2}$ och konstruera en ortogonal bas $T_0(t), T_1(t), \ldots, T_n(t)$ i motsvarande Euklidiska rum genom att tillämpa Gram-Schmidts metod på basen $1, t, t^2, \ldots, t^n$. Skala resultatet så att

$$(T_k(t)|T_k(t)) = \int_{-1}^{1} T_k(t)T_k(t)w(t)\,dt = \pi/2, \qquad k = 1, 2, \ldots, n.$$

De genererade polynomen kallas Chebyshevpolynom av första ordningen. Plotta de sex första Chebyshevpolynomen $T_0(t), T_1(t), \ldots, T_5(t)$ i samma figur för $t$ i intervallet $[-1, 1]$.

# Kapitel 4

# Gränsvärden, derivator och summor

Med tilläggspaketet för symbolisk matematik kan man beräkna gränsvärden och derivator analytiskt. Man kan även bestämma integraler och lösa differentialekvationer. Funktionerna man studerar kan antingen ges direkt i termer av de elementära matematiska funktionerna eller så kan de ges som funktions M-filer. För att få en lista över de elementära matematiska funktionerna skriver man `help elfun` vid kommandopromptern.

## 4.1 Funktioner

Merparten av de elementära matematiska funktionerna accepterar symboliska uttryck som argument och resultatet blir också ett symboliskt funktionsuttryck. Då vi till exempel skriver

```
syms x
f = exp(-x)*sin(x);
```

får vi det symboliska funktionsuttrycket $e^{-x}\sin x$ som lagras i variabeln $f$. Funktioner kan också definieras i funktions M-filer. En funktions M-fil inleds alltid med ordet function, sedan kommer invariabeln $x$, funktionsnamnet och funktionsuttrycket $y$ på följande sätt

```
function y = funktionsnamn(x)
y = funktionsuttryck
```

Under den första raden ger man regeln för hur funktionsuttrycket skall beräknas. Då man sparar en funktionsfil ska man ge den namnet som står på första raden följt av extensionen **.m** .

**Exempel 4.1.** För att definiera funktionen $y = e^{-x} \sin x$ i en M-fil öppnar vi MATLABs editor och skriver

```
function y = fun(x)
y = exp(-x).*sin(x);
```

Då vi sparar funktionsfilen skall den ges namnet **fun.m**. Notera att vi använde punktstjärna `.*` vid multiplikationen. Vi kan nu anropa funktionen antingen med en numerisk vektor eller med en symbolisk invariabel.

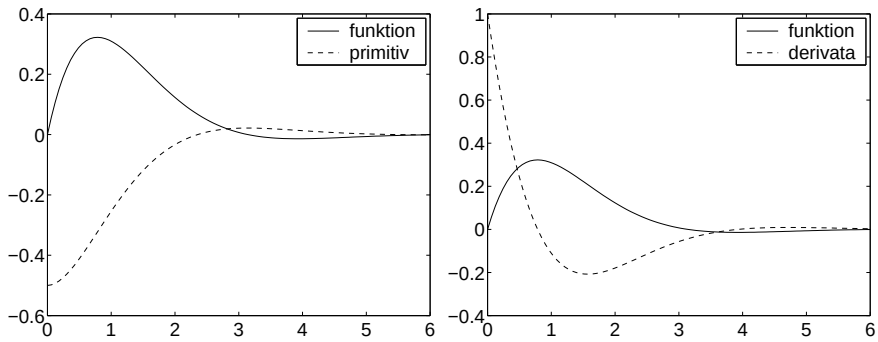
Vi kallar på funktionen genom att skriva funktionsnamnet följt av invariabeln. För att plotta funktionen och den primitiva funktionen i intervallet $[0, 6]$ ger vi kommandona

```
syms x
f = fun(x);
F = int(f,x);       % primitiv funktion
xv = linspace(0,6,100);
fv = subs(f,x,xv); Fv = subs(F,x,xv);
plot(xv,fv,xv,Fv,'--'), legend('funktion','primitiv')
```

vilket ger plotten till vänster i figur 4.1. Funktionen och dess derivata plottas genom kommandona

```
syms x
f = fun(x);
df = diff(f,x);     % derivata
xv = linspace(0,6,100);
fv = subs(f,x,xv); dfv = subs(df,x,xv);
plot(xv,fv,xv,dfv,'--'), legend('funktion','derivata')
```

vilket ger plotten till höger i figur 4.1. Derivationskommandona återkommer i sektion 4.4. Primitiva funktioner behandlas i kapitel 5. □




Figur 4.1: *En funktion tillsammans med primitiv funktion och derivata.*

## 4.2 Speciella matematiska funktioner

Med tilläggsprogrammet för symbolisk matematik (obs gäller ej studentversionen) får man tillgång till ett antal speciella matematiska funktioner som ibland kommer upp som lösningar till integraler eller differentialekvationer. De viktigaste av dessa specialfunktioner är listade nedan.

| | |
|---|---|
| `BesselI, BesselJ` | Besselfunktioner |
| `BesselK, BesselY` | |
| `Beta` | betafunktionen |
| `Binomial` | binomialfunktionen |
| `EllipticF, EllipticK` | elliptiska integraler |
| `erf, erfc` | felfunktioner |
| `FresnelC, FresnelS` | Fresnels sinus- och cosinusintegraler |
| `GAMMA` | gammafunktionen |
| `zeta` | Riemann zetafunktion |
| `Dirac, Heaviside` | Delta- och stegfunktion (distributioner) |

Till skillnad från de elementära matematiska funktionerna som $\sin(x), \ln(x)$ etc så kan dessa speciella funktioner inte hanteras av substitutionskommandot. För att beräkna funktionsvärden för dessa speciella funktioner måste man i stället använda kommandot `mfun`. För att lista de speciella funktionerna ger man kommandot `mfunlist`.

| | |
|---|---|
| `mfunlist` | ger en lista över de speciella matematiska funktionerna. För att få mera information om var och en av funktionerna i listan ge kommandot `mhelp` följt av funktionsnamnet. |
| `mfun('fun', p1,p2,...,pn)` | evaluerar den speciella funktionen med namn *fun* för parametrarna $p1, p2, \ldots, pn$. Här kan den sista parametern vara en vektor. |

**Exempel 4.2.**

(a) För att få mer information om Fresnels sinus- och cosinusintegraler ger vi kommandot

```
mhelp FresnelS
```

och MATLAB svarar (utskriften har editerats och kortats ned)

```
FresnelC - The Fresnel Cosine Integral
FresnelS - The Fresnel Sine Integral
Fresnelf, Fresnelg - The Fresnel Auxiliary Functions
```

```
Calling Sequence:
     FresnelC(x)
     FresnelS(x)
     Fresnelg(x)
     Fresnelf(x)

Parameters:
     x - an expression

Description:
- The Fresnel cosine integral is defined as follows:
      FresnelC(x) =  int(cos(Pi/2*t^2), t=0..x);

Description:
- The Fresnel sine integral is defined as follows:
      FresnelS(x) =  int(sin(Pi/2*t^2), t=0..x);
      .....
```

(b) För att plotta Fresnels sinus- och cosinusintegraler i intervallet $[0, 6]$ ger vi kommandona

```
x = linspace(0,6,200);
FC = mfun('FresnelC',x);
FS = mfun('FresnelS',x);
plot(x,FC,x,FS,'--')
legend('FresnelC','FresnelS')
```

vilket ger plotten till vänster i figur 4.2.

(c) Mera information om Besselfunktionerna fås igenom

```
mhelp BesselJ
```

Utskriften (som på nytt har editerats och kortats ned) blir

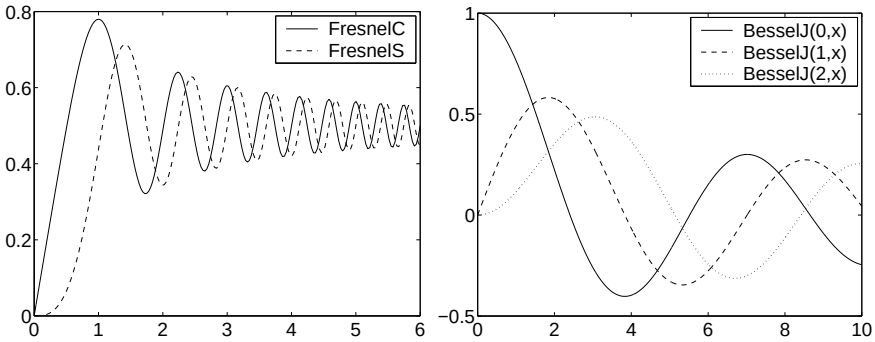
```
BesselI, BesselJ - The Bessel functions of the first kind
BesselK, BesselY - The Bessel functions of the second kind

Calling Sequence:
     BesselI(v, x)
     BesselJ(v, x)
     BesselK(v, x)
     BesselY(v, x)

Parameters:
```

```
        v - an algebraic expression (the order or index)
        x - an algebraic expression (the argument)

    Description:
    - BesselJ and BesselY are the Bessel functions of the first
      and second kinds, respectively. They satisfy Bessel's
      equation:

    2                 2    2
   x  y'' + x y' + (x  - v ) y = 0

    - BesselI and BesselK are the modified Bessel functions of
      the first and second kinds, respectively. They satisfy the
      modified Bessel equation:

    2                 2    2
   x  y'' + x y' - (x  + v ) y = 0
      .....
```

(d) Besselfunktionerna av första slaget med index $v = 0, 1, 2$ plottas i intervallet $[0, 10]$ genom

```
x = 0:0.01:10;
BJ0 = mfun('BesselJ',0,x);
BJ1 = mfun('BesselJ',1,x);
BJ2 = mfun('BesselJ',2,x);
plot(x,BJ0,x,BJ1,'--',x,BJ2,':')
legend('BesselJ(0,x)','BesselJ(1,x)','BesselJ(2,x)')
```

vilket ger plotten till höger i figur 4.2. □




Figur 4.2: *Speciella matematiska funktioner plottade med hjälp av kommandot* `mfun`.

## 4.3 Gränsvärden

Gränsvärdesbegreppet är av fundamental betydelse vid matematisk analys. I MATLAB beräknas gränsvärden med hjälp av kommandot `limit`.

| | |
|---|---|
| `limit(f,x,a)` | bestämmer gränsvärdet av det symboliska uttrycket $f$ då $x \to a$. Då $x$ utelämnas beräknas gränsvärdet med avseendet på defaultvariabeln. |
| `limit(f,x,a,'right')` | bestämmer gränsvärdet av det symboliska uttrycket $f$ då $x \to a^+$ (högergränsvärde). Då $x$ utelämnas beräknas gränsvärdet med avseendet på defaultvariabeln. |
| `limit(f,x,a,'left')` | bestämmer gränsvärdet av det symboliska uttrycket $f$ då $x \to a^-$ (vänstergränsvärde). Då $x$ utelämnas beräknas gränsvärdet med avseendet på defaultvariabeln. |

I kommandot `limit` kan $a$ också vara `-inf` eller `inf`, varvid vi får gränsvärdena av $f$ då $x \to -\infty$ och då $x \to \infty$.

**Exempel 4.3.**

(a) Gränsvärdet

$$\lim_{x \to 0} \frac{\sin(ax)}{bx}$$

beräknas genom

```
syms x a b
f = sin(a*x)/(b*x);
flim = limit(f,x,0)
```

vilket ger

```
flim =
a/b
```

(b) Betrakta funktionen

$$f(t) = 3 + e^{-at}, \quad a > 0$$

För att beräkna gränsvärdet då $t \to \infty$ ger vi kommandona

```
syms t
syms a positive
f = 3 + exp(-a*t);
flim = limit(f,t,inf)
```

Vi får svaret

```
flim =
3
```

Observera att om man glömmer att definiera $a$ som positiv så blir gränsvärdet obestämt. MATLAB markerar detta genom att returnerade det inmatade uttrycket obehandlat på följande sätt

```
flim =
limit(3+exp(-a*t),t = inf)
```

(c) Vi har funktionen

$$f(x) = \cos(x^2)\,(x-3)\sqrt{1+\frac{1}{(x-3)^2}},$$

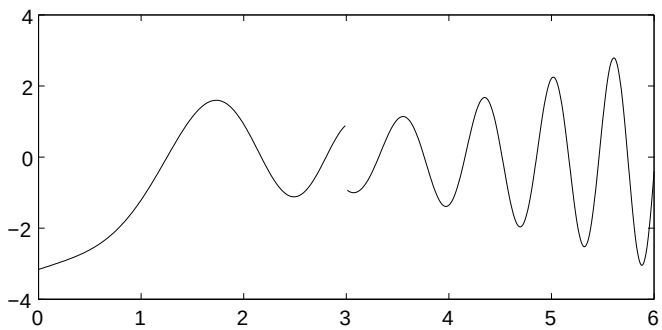
som är definierad för alla $x$ utom $x = 3$. Grafen till funktionen visas i figur 4.3. Funktionen saknar gränsvärde då $x$ går mot 3, men har både höger- och vänstergränsvärden. Dessa beräknas genom

```
syms x
f = cos(x^2)*(x-3)*sqrt(1 + 1/(x-3)^2);
h = limit(f,x,3,'right');
v = limit(f,x,3,'left');
disp(h), disp(v)
```

vilket ger resultatet

```
cos(9)
-cos(9)
```

□




Figur 4.3: *Höger- och vänstergränsvärden av en diskontinuerlig funktion.*

## 4.4 Derivator

Derivator användes i en mängd olika sammanhang. Ofta för att karakterisera funktioners utseende men också för att lösa optimeringsproblem.

| | |
|---|---|
| `diff(f,x)` | deriverar den symboliska funktionen $f$ med avseende på $x$. Då $x$ utelämnas deriveras uttrycket med avseende på defaultvariabeln. |
| `diff(f,x,n)` | deriverar den symboliska funktionen $n$ stycken gånger med avseende på $x$. Då $x$ utelämnas deriveras uttrycket med avseende på defaultvariabeln. |
| `jacobian(f,v)` | beräknar funktionalmatrisen (Jacobianen) av den skalära eller vektoriella funktionen $f$ med avseende på vektorn $v$. Elementet $(i, j)$ av den resulterande funktionalmatrisen är $\partial f_i/\partial v_j$. Notera att när $f$ är en skalär så blir funktionalmatrisen lika med gradienten av $f$. |

**Exempel 4.4.** Betrakta funktionen

$$f(x) = \frac{ax}{1 + \cos(bx)}.$$

(a) Derivatan med avseende på $x$ fås genom

```
syms x a b
f = a*x/(1 + cos(b*x));
dfx = diff(f,x);
pretty(dfx)
```

Utskriften från MATLAB blir

```
     a         a x sin(b x) b
------------ + ---------------
1 + cos(b x)                2
               (1 + cos(b x))
```

(b) För att derivera funktionen två gånger ger vi kommandona

```
syms x a b
f = a*x/(1 + cos(b*x));
d2fx = diff(f,x,2);
pretty(d2fx)
```

vilket ger

```
                                   2  2                  2
   a sin(b x) b        a x sin(b x)  b     a x cos(b x) b
2 --------------- + 2 ---------------- + ---------------
                 2                   3                  2
  (1 + cos(b x))      (1 + cos(b x))      (1 + cos(b x))
```
□

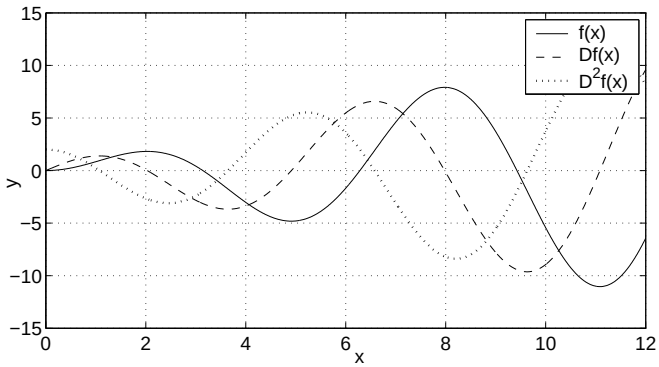
**Exempel 4.5.** Vi har funktionen

$$f(x) = x\sin(x)$$

Funktionen, förstaderivatan och andraderivata plottas med kommandona

```
syms x                % def. x som en symbolisk variabel
f1 = x*sin(x);        % funktionen f(x)
f2 = diff(f1,x);      % förstaderivatan f'(x)
f3 = diff(f1,x,2);    % andraderivatan f''(x)
xv = 0:0.01:12;       % vektor med värden från 0 till 12
f1v = subs(f1,x,xv);  % vektor med värden på f(x)
f2v = subs(f2,x,xv);  % vektor med värden på f'(x)
f3v = subs(f3,x,xv);  % vektor med värden på f''(x)
plot(xv,f1v,'-',xv,f2v,'--',xv,f3v,':') % plotta
xlabel('x'), ylabel('y')
legend('f(x)','Df(x)','D^2f(x)')  % teckenförklaring
grid on               % gridlinjer
```

Den genererade plotten visas i figur 4.4. Notera att funktionen $f(x)$ har inflexionspunkter (punkter där funktionen övergår från att vara konkav till konvex) precis då andraderivatan är noll. □




Figur 4.4: *Funktionen $f(x) = x\sin(x)$ tillsammans med derivatan och andraderivatan. Funktionen $f(x)$ har inflexionspunkter där andraderivatan är noll.*

**Exempel 4.6.**

(a) Betrakta den skalära funktionen

$$f(x,y) = x\sin(x/y).$$

För att beräkna gradienten

$$\mathrm{grad} f(x,y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$$

ger vi kommandot

```
syms x y
f = x*sin(x/y);
gradf = jacobian(f,[x y])
```

MATLAB svarar då

```
gradf =
[ sin(x/y)+x*cos(x/y)/y,      -x^2*cos(x/y)/y^2]
```

(b) Vi har funktionen

$$f(x,y) = \frac{x^2}{\ln(xy)}.$$

Riktningsderivatan i punkten $(1,8)$ i riktningen $v = (1,2)$ ges av $\mathrm{grad} f(1,8) \cdot \frac{v}{|v|}$. Det analytiska uttrycket fås med kommandona

```
syms x y
f = x^2/log(x*y);
gradf = jacobian(f,[x y])
v = [1 2]/norm([1 2]);
df_v = dot(subs(gradf,{x,y},{sym('1'),sym('8')}),v);
pretty(simplify(df_v))
```

vilket ger

```
        1/2
       5    (24 log(2) - 5)
1/180 --------------------
                   2
               log(2)
```

För att få motsvarande numeriska värde ger man kommandot `double(df_v)`.

(c) Övergången från rätvinkliga till rymdpolära koordinater defineras av

$$\begin{cases} x = r\sin\theta\cos\varphi \\ y = r\sin\theta\sin\varphi \\ z = r\cos\theta \end{cases}.$$

Funktionalmatrisen av den vektorvärda funktionen beräknas genom

```
syms r t p
x = r*sin(t)*cos(p);
y = r*sin(t)*sin(p);
z = r*cos(t);
df = jacobian([x y z],[r t p])
```

och vi får

```
df =
[    sin(t)*cos(p),  r*cos(t)*cos(p), -r*sin(t)*sin(p)]
[    sin(t)*sin(p),  r*cos(t)*sin(p),  r*sin(t)*cos(p)]
[           cos(t),        -r*sin(t),                0]
```

För att få funktionaldeterminanten skriver vi

```
simplify(det(df))
```

och MATLAB svarar

```
ans =
sin(t)*r^2
```

(d) Vi har en funktion $f(x, y)$. Flervariabelmotsvarigheten till andraderivata ges av den så kallade Hessianen

$$\begin{pmatrix} \dfrac{\partial^2 f}{\partial x^2} & \dfrac{\partial^2 f}{\partial y \partial x} \\ \dfrac{\partial^2 f}{\partial x \partial y} & \dfrac{\partial^2 f}{\partial y^2} \end{pmatrix}.$$

För att beräkna Hessianen till $f(x, y) = \sin(x\,y + 1)$ använder vi kommandot `jacobian` två gånger

```
syms x y
H = jacobian(jacobian(sin(x*y+1),[x y]),[x y])
```

Matrisen blir

```
H =
[              -sin(x*y+1)*y^2, -sin(x*y+1)*x*y+cos(x*y+1)]
[ -sin(x*y+1)*x*y+cos(x*y+1),             -sin(x*y+1)*x^2]
```

□

## 4.5 Taylorutvecklingar

En funktion $f(x)$ som är tillräckligt många gånger deriverbar kring en punkt $x = a$ kan approximeras med ett Taylorpolynom

$$f(x) \approx f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \ldots + \frac{f^{(n-1)}(a)}{(n-1)!}(x-a)^{(n-1)}.$$

Felet i approximationen ges av en restterm som kan skrivas på formen

$$R_n(x) = \frac{f^{(n)}(a + \theta(x-a))}{n!}(x-a)^n,$$

där $0 \leq \theta \leq 1$. Taylorutvecklingar och resttermsuppskattningar spelar en mycket viktig roll både i den matematiska och den numeriska analysen.

| | |
|---|---|
| `taylor(f,n,x,a)` | ger Taylorpolynomet av ordning $n-1$ till $f$, som är en funktion av $x$, kring en punkt $a$. Då $n$ utelämnas används defaultvärdet 6. Då $x$ utelämnas används defaultvariabeln. Då $a$ utelämnas fås Taylorpolynomet kring punkten 0. |

**Exempel 4.7.** Vi har funktionen

$$f(x) = e^{-x}$$

(a) Taylorpolynomet av ordning 3 kring punkten 0 fås genom

```
syms x
f = exp(-x);
p = taylor(f,4); pretty(p)
```

MATLAB svarar

```
                2          3
1 - x + 1/2 x  - 1/6 x
```

(b) Taylorpolynomet av ordning 2 kring punkten 1 fås genom

```
syms x
f = exp(-x);
p = taylor(f,3,1); pretty(p)
```

vilket ger

```
                                                    2
exp(-1) - exp(-1) (x - 1) + 1/2 exp(-1) (x - 1)
```

□

**Exempel 4.8.** Bestäm gränsvärdet av funktionen

$$f(x) = \frac{\sin(x) - x}{x\cos(x) - x}$$

då $x$ går mot noll.

Uppgiften kan lösas genom att studera Taylorutvecklingarna kring noll av både täljaren och nämnaren. Vilken ordning av utvecklingarna som behövs beror på funktionsuttrycken, och i allmänhet får man pröva sig fram. Följande kommandon bestämmer kvoten mellan Taylorutvecklingarna

```
syms x
t = sin(x) - x;       % täljare
n = x*cos(x) - x;     % nämnare
pt = taylor(t,6);     % Taylorutveckling av ordning 5
pn = taylor(n,6);     % Taylorutveckling av ordning 5
pretty(pt/pn)
```

Vi får utskriften

```
        3           5
- 1/6 x   + 1/120 x
-------------------
        3          5
- 1/2 x   + 1/24 x
```

Då $x$ är nära noll kan $x^5$ termerna försummas i förhållande till $x^3$ termerna. Gränsvärdet blir alltså lika med $(-1/6)/(-1/2) = 1/3$. □

**Exempel 4.9.** En kropp med vilomassa $m_0$ som rör sig med farten $v$ har enligt Einsteins relativitetsteori en kinetisk energi som ges av uttrycket

$$W_{kin} = \frac{m_0 c^2}{\sqrt{1-(v/c)^2}} - m_0 c^2.$$

Vi utvecklar uttrycket i termer av $(v/c)$

```
syms m0 v c x
Wkin = taylor(m0*c^2/sqrt(1-x^2) - m0*c^2,x,3);
pretty(subs(Wkin,x,v/c))
```

vilket ger

```
          2
1/2 m0 v
```

Newtons klassiska uttryck för den kinetiska energin hos en kropp i rörelse kommer alltså ut som den första termen i Taylorutvecklingen av det relativistiska uttrycket. □

**Exempel 4.10.** Följande kommandon plottar funktionen

$$f(x) = e^x$$

tillsammans med Taylorpolynom av ordning 1 till 4 kring punkten $x = 0$

```
syms x
f = exp(x);
xv = -2:0.01:2;
fv = subs(f,x,xv);
for i = 2:5
  p = taylor(f,i,0);
  pv = subs(p,x,xv);
  subplot(2,2,i-1)
  plot(xv,fv,xv,pv,'--')
  ylim([-2 8])
  string = num2str(i-1,'%2.0f');
  title(['e^x och Taylorpolynom av grad ' string])
end
```

Plottarna visas i figur 4.5. □

## 4.6 Summor

Summor förekommer i en mängd olika sammanhang. Med toolboxen för symbolisk matematik finns möjlighet att utföra symbolisk summering.

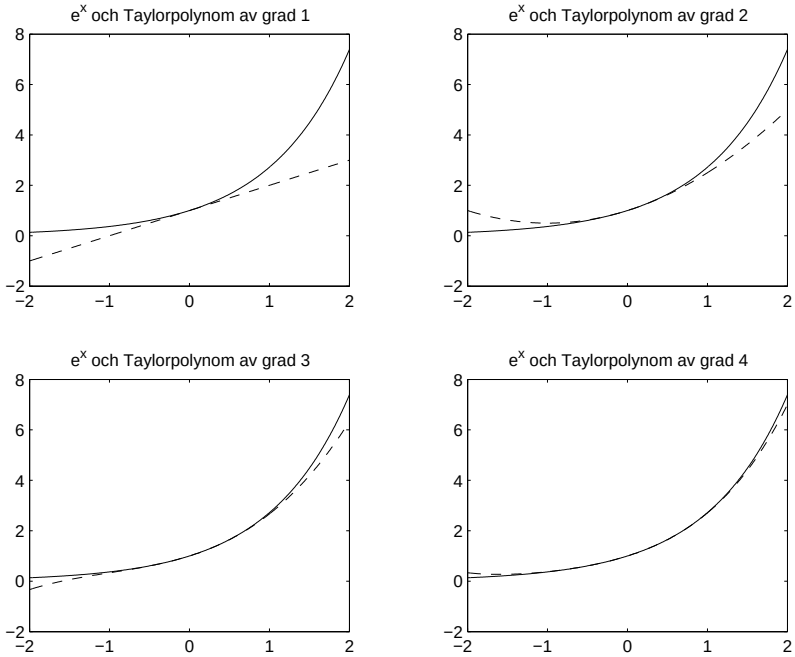
| | |
|---|---|
| `symsum(s,v,a,b)` | ger summan av det symboliska uttrycket $s$ där summationsindex $v$ går mellan $a$ och $b$. Då $a$ och $b$ utelämnas sker summationen automatiskt från 0 till $v - 1$. |

I kommandot `symsum` kan $a$ och $b$ också vara `-inf` respektive `inf` så att vi beräknar oändliga summor.

**Exempel 4.11.**

(a) En summa av formen

$$\sum_{k=0}^{n-1} ax^k = a + ax + ax^2 + \ldots + ax^{n-1}$$




Figur 4.5: *Funktionen $e^x$ tillsammans med Taylorpolynom av grad 1 till 4.*

kallas en geometrisk summa. Summan beräknas med

```
syms a x k n
pretty(symsum(a*x^k,k,0,n-1))
```

och svaret blir

```
   n
a x        a
----- - -----
x - 1   x - 1
```

(b) Den oändliga geometriska summan

$$\sum_{k=0}^{\infty} ax^k = a + ax + ax^2 + \dots$$

beräknas genom

```
syms a x k
pretty(symsum(a*x^k,k,0,inf))
```

vilket ger svaret

```
     a
- -----
  x - 1
```

Notera att MATLAB förutsätter att $|x| < 1$ så att summan är konvergent.

(c) Vi undersöker den så kallade teleskopsumman

$$\sum_{v=1}^{n} \frac{1}{v(v+1)} = \frac{1}{1 \cdot 2} + \frac{1}{2 \cdot 3} + \ldots + \frac{1}{n \cdot (n+1)}.$$

Kommandona

```
syms v n
pretty(symsum(1/(v*(v+1)),v,1,n))
```

ger uttrycket

```
     1
- ----- + 1
  n + 1
```

(d) Vi har den oändliga summan

$$\sum_{k=0}^{\infty} \frac{1}{k!} = \frac{1}{0!} + \frac{1}{1!} + \frac{1}{2!} + \ldots$$

där $k! = 1 \cdot 2 \cdot \ldots \cdot k$ är $k$-fakulteten. Summan beräknas genom

```
syms k
symsum(1/sym('k!'),k,0,inf)
```

MATLAB svarar

```
exp(1)
```

I uttrycket för summan skapade vi det symboliska uttrycket $k!$ med hjälp av kommandot `sym`. Alternativt skulle vi kunna använda `prod(1:k)`. □

## 4.7 Tillämpningsexempel; optimering

En av fördelarna med tilläggspaketet för symbolisk matematik är att det är enkelt att använda analytiskt beräknade uttryck i numeriska metoder. Som illustration skall vi titta på minimering av en funktion av flera variabler. För enkelhetens skull begränsar vi oss till en funktion av två variabler $f(x_1, x_2)$.

I Newtons metod utgår man från ett gissat värde $x^0 = (x_1^0, x_2^0)$ på minpunkten. Taylorutvecklingen av funktionen $f(x_1, x_2)$ i punkten $(x_1^0, x_2^0)$ ges av

$$f(x_1, x_2) \approx f(x^0) + \frac{\partial f(x^0)}{\partial x_1} dx_1 + \frac{\partial f(x^0)}{\partial x_2} dx_2 + \frac{1}{2}\left( \frac{\partial^2 f(x^0)}{\partial x_1^2} dx_1^2 + 2\frac{\partial^2 f(x^0)}{\partial x_1 \partial x_2} dx_1 dx_2 + \frac{\partial^2 f(x^0)}{\partial x_2^2} dx_2^2 \right)$$

där $dx_1 = (x_1 - x_1^0)$ och $dx_2 = (x_2 - x_2^0)$. Då startvärdet ligger nära den exakta minpunkten definierar Taylorutvecklingen en uppåtböjd paraboloid. Minimum av paraboloiden med avseende på $dx_1$ och $dx_2$ fås genom att derivera och sätta derivatan till noll. Detta leder till

$$\begin{pmatrix} \frac{\partial^2 f(x^0)}{\partial x_1^2} & \frac{\partial^2 f(x^0)}{\partial x_1 \partial x_2} \\ \frac{\partial^2 f(x^0)}{\partial x_1 \partial x_2} & \frac{\partial^2 f(x^0)}{\partial x_2^2} \end{pmatrix} \begin{pmatrix} dx_1 \\ dx_2 \end{pmatrix} = - \begin{pmatrix} \frac{\partial f(x^0)}{\partial x_1} \\ \frac{\partial f(x^0)}{\partial x_2} \end{pmatrix}.$$

I ekvationssystemet ges koefficientmatrisen av Hessianen till $f(x_1, x_2)$ medan högerledet är den negativa gradienten. Punkten $x^1 = (x_1^1, x_2^1) = (x_1^0 + dx_1, x_2^0 + dx_2)$ är, förutsatt att vi startade tillräckligt nära den exakta minpunkten, i allmänhet en bättre approximation till den sökta minpunkten än startvärdet $(x_1^0, x_2^0)$. Vi tar $(x_1^1, x_2^1)$ som nytt startvärde och upprepar proceduren tills det att avståndet

$$\|x^1 - x^0\|_2 = \sqrt{dx_1^2 + dx_2^2}.$$

understiger en given tolerans *tol*.

Följande funktionsfil bestämmer minimum till en funktion med hjälp av Newtons metod. Inparameter till funktionsfilen är *fun* som är namnet på funktionsfilen som beräknar $f(x_1, x_2)$, $x^0$ som är startvektorn och *tol*. Utparameter är det sist beräknade $x^1$ och antalet iterationer $n$. Kommandona `df = jacobian(f,x)` och `H = jacobian(df,x)` används för att analytiskt beräkna gradienten och Hessianen. Funktionsvärdena i punkten $x^0$ beräknas med hjälp av substitutionskommandot `subs`. Notera att gradienten `df` är en radvektor och att den måste transponeras vid lösningen av ekvationssystemet. På samma sätt så måste kolonnvektorn `dx` transponeras innan den adderas till radvektorn `x0`.

```
function [x1,n] = optimering(fun,x0,tol)
% Denna funktionsfil bestämmer minimum till f(x).
% fun är namnet på filen som beräknar f(x).
% I df lagras den symboliskt bestämda
% gradientvektorn. I H lagras Hessianen.
```

```
syms x1 x2
x = [x1 x2];
f = feval(fun,x);
df = jacobian(f,x);
H = jacobian(df,x);
n = 0;
norm_dx = 1e30;
while norm_dx > tol & n < 50
   dfx0 = subs(df,x,x0);
   Hx0 = subs(H,x,x0);
   dx = Hx0\(-dfx0)';    % transponera dfx0
   x1 = x0 + dx';        % transponera dx
   x0 = x1;
   norm_dx = norm(dx,2);
   n = n + 1;
end
```

Vi testar optimeringsprogrammet på ett exempel.

**Exempel 4.12.** Genom att göra en konturplott ser man att funktionen

$$f(x_1, x_2) = (x_1^2 - x_2)^2 + (0.9 - x_1 - 0.1x_2)^2;$$

har en minpunkt nära origo. För att bestämma minpunkten med Newtons metod skriver vi en funktionsfil med $f(x_1, x_2)$

```
function y = fun(x)
y = (x(1).^2-x(2)).^2 + (0.9-x(1)-0.1*x(2)).^2;
```

Vi tar toleransen $10^{-12}$. En uppskattning av minpunkten fås genom

```
[x,n] = optimering('fun',[0 0],1e-12)
```

och MATLAB svarar

```
x =
   0.83095189484530   0.69048105154700
n =
     6
```

□

## 4.8 Vidare läsning

Jönsson, P. (2008), Matematik med datoralgebrasystem, Studentlitteratur, ISBN 978-91-44-05250-2

Gränsvärden, derivator och summor diskuteras ingående i kapitel 9. Flervariabeltillämpningar tas upp i kapitel 13.

Persson, A. och Böiers L.-C, A (2001), Analys i en variabel, Studentlitteratur, ISBN 91-44-02056-2

Ger grunderna i den matematiska analysen. Begrepp som gränsvärde och derivata tas upp. Även Taylorutvecklingar och summor ingår.

Persson, A. och Böiers L-C, A (2005), Analys i flera variabler, Studentlitteratur, ISBN 91-44-03869-0

Boken behandlar flervariabelanalys. Gradient, funktionalmatris och funktionaldeterminant behandlas ingående.

Heath, M (2002), Scientific computing, an introductory survey. McGraw-Hill. ISBN 0-070112229-X

Newtons metod för optimering diskuteras i kapitel 6.

## 4.9 Instuderingsfrågor

1. Hur beräknar man speciella funktioner som t.ex. Besselfunktioner?
2. Vad menas med att en funktion saknar gränsvärde men har ett högergränsvärde?
3. Vilket är kommandot för att beräkna ett gränsvärde då $x \to \infty$?
4. Vilket är kommandot för att beräkna gränsvärdet då $x \to a^+$?
5. Vad menas med en inflexionspunkt?
6. Hur beräknar man första- respektive andraderivatan av en funktion?
7. Vi har en funktion $f(x, y)$. Hur definieras gradient respektive Hessianmatris. Hur beräknar man dessa kvantiteter?
8. Vi har en vektorvärd funktion $(f_1(x, y), f_2(x, y))$. Vad menas med funktionalmatrisen och hur beräknas den?
9. Vad menas med ett Taylorpolynom? Hur skrivs feltermen?
10. Vi har en funktion $f(x)$. Hur beräknar man Taylorpolynomet av ordning 7 kring punkten 2?
11. Vilket är kommandot för att beräkna en symbolisk summa?
12. Hur ser en geometrisk summa ut?

13. Hur skriver man $k!$ (k-fakultet) på symbolisk form?

14. Beskriv Newtons metod för minimering av en funktion $f(x_1, x_2)$.

## 4.10 Övningar

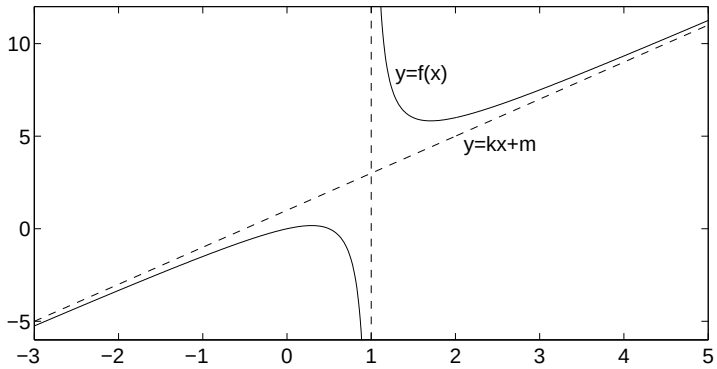
1. Skriv `mhelp dawson` för att få reda på mer om Dawsons integral. Plotta Dawsons integral i intervallet [0,10].

2. Beräkna följande gränsvärden.

   (a) $\lim_{x\to\infty}\left(1-\frac{1}{2x}-\frac{1}{2x^2}\right)^{2x}$ (b) $\lim_{x\to\infty}\frac{(x+1)^x}{x^{x+1}}$ (c) $\lim_{x\to 1/2}\frac{3-x}{x^2+1}$

   (d) $\lim_{x\to 0^+} x^x$ (e) $\lim_{x\to 0}\sin(1/x)$

3. Som en tillämpning på gränsvärden ska vi titta på sneda asymptoter till en funktion. Linjen $y = kx + m$ är en sned asymptot till funktionen $f(x)$ om avståndet från kurvpunkten till den punkt på asymptoten, som har samma $x$-koordinat, går mot noll då $x \to \infty$ eller $x \to -\infty$.




Figur 4.6: *Funktion med sned asymptot*

Vi undersöker alltså

$$g(x) = f(x) - kx - m.$$

Linjen $y = kx + m$ är asymptot då och endast då $g(x) \to 0$, då $x \to \infty$ eller $x \to -\infty$. Vi betraktar i fortsättningen endast fallet $x \to \infty$. Om

$y = kx + m$ är en asymptot får vi $k$ genom att dela $f(x)$ med $x$ och låta $x \to \infty$

$$k = \lim_{x\to\infty} \frac{f(x)}{x}.$$

När väl $k$ är bestämt fås $m$ genom

$$m = \lim_{x\to\infty} (f(x) - kx).$$

Om något av gränsvärdena inte existerar saknas sned asymptot.

Undersök nedanstående funktioner med avseende på sneda asymptoter.

(a) $f(x) = 3 - x/3 + \ln(e^x + 1)$ (b) $\dfrac{x^3 + 2x^2}{x^2 - 1}$

Plotta asymptoterna och funktionen i samma figur.

4. Bestäm derivatan och andraderivatan av följande funktioner.

(a) $f(x) = \dfrac{\sin(x)}{\cos(x) + x^3}$ (b) $f(x) = e^{a\sin(x^2)}$ (c) $f(x) = \dfrac{1 + ax}{b + cx}$

5. Plotta funktionen $f(x) = -1 + \dfrac{x}{x + \sin(x)}$ tillsammans med första- och andraderivatan i intervallet $[0, 10]$.

6. Beräkna gradienten och Hessianen till följande funktioner.

(a) $f(x, y) = \dfrac{e^{ax^2+by^2+2xy}}{x^2 + y^2}$ (b) $f(x, y, z) = \dfrac{x^2 + y^2 + \sin(xy)}{1 + z^2}$

7. Bestäm funktionalmatrisen och funktionaldeterminanten till

$$\begin{cases} x = r\cos\theta \\ y = r\sin\theta \end{cases}.$$

8. Betrakta funktionen $y = \ln(x)$. Beräkna Taylorpolynomen av ordning 1 till 4 kring punkten $x = 1$. Plotta den ursprungliga funktionen och Taylorpolynomen i samma figur.

9. Beräkna följande summor

(a) $\displaystyle\sum_{k=5}^{n-1} ax^k$ (b) $\displaystyle\sum_{k=0}^{n} k^2$ (c) $\displaystyle\sum_{l=1}^{\infty} 1/l^2$

10. Funktionen $f(x_1, x_2) = (x_1^2 + 2x_2^2 - 1)^2 + (\,4(x_1 - 0.4)^2 + x_2^2 - 1)^2$ har en minpunkt i närheten av $(0, 0.7)$. Bestäm minpunkten mera exakt med Newtons metod.

11. Skriv ett program som bestämmer lösningar till ekvationen $f(x) = 0$ med hjälp av Newton-Raphsons metod (se avsnitt 11.4HB). Inparameter till funktionsfilen är; *fun*, som är en teckensträng innehållande namnet på funktionsfilen som beräknar $f(x)$, startvärdet $x_0$ och toleransen *tol*. Utparameter är det sist beräknade värdet $x_1$ och antalet iterationer $n$. Kommandot `dfun = diff(f,x)` används för att analytiskt beräkna derivatan $f'(x)$. Funktionsvärdena i punkten $x_0$ beräknas med hjälp av substitutionskommandot `subs`. Testa ditt program på lite olika problem och visa att det fungerar

12. Skriv ett program som beräknar nollställena till ett system av ekvationer med hjälp av Newton-Raphsons metod (jfr avsnitt 11.4HB). Använd kommandot `jacobian` för att beräkna funktionalmatrisen analytiskt. Funktionsvärdena i punkten $x_0$ beräknas med hjälp av substitutionskommandot `subs`. Använd programmet för att lösa ekvationssystemet i exempel 11.8HB.

# Kapitel 5

# Integraler och differentialekvationer

Vi ska nu se hur man kan låta MATLAB beräkna integraler och lösa differentialekvationer. Speciellt kommandona för att bestämma integraler är kraftfulla och undanröjer många av svårigheterna som annars är förknippade med detta område av matematiken.

## 5.1 Integraler

Vi har följande kommandon för att beräkna integraler.

| | |
|---|---|
| `int(f,x)` | beräknar den obestämda integralen av $f$ (primitv funktion) med avseende på variabeln $x$. Då $x$ utelämnas beräknas integralen med avseende på defaultvariabeln. |
| `int(f,x,a,b)` | beräknar den bestämda integralen av $f$ från $a$ till $b$ med avseende på variabeln $x$. Då $x$ utelämnas beräknas integralen med avseende på defaultvariabeln. |

**Exempel 5.1.**

(a) Den obestämda integralen (primitiva funktionen)

$$\int x^2 \sin(x)\, dx$$

fås genom

```
syms x
```

```
int(x^2*sin(x),x)
```

MATLAB skriver ut funktionen

```
ans =
-x^2*cos(x)+2*cos(x)+2*x*sin(x)
```

(b) För att beräkna integralen

$$\int_0^{\pi} x^2 \sin(x)\, dx$$

ger vi kommandona

```
syms x
int(x^2*sin(x),x,0,pi)
```

vilket ger svaret

```
ans =
pi^2-4
```

□

**Exempel 5.2.** Det går även bra att beräkna generaliserade integraler.

(a) Integralen

$$\int_0^{\infty} \sin(x) e^{-ax}\, dx, \quad a > 0$$

beräknas med kommandona

```
syms x
syms a positive   % obs a måste definieras som positiv
I = int(sin(x)*exp(-a*x),x,0,inf);
pretty(I)
```

och resultatet blir

```
                        2     1/2
                    (a~  + 1)
---------------------------------------------------
                  /    2                           \
   /     1 \1/2 |  a~         2            1       |
a~ |1 + ---|    |------- + ------- + -------------|
   |      2|    |     1         1      2 /     1 \|
   \    a~ /    |1 + ---   1 + ---   a~  |1 + ---||
                |      2         2       |      2||
                \    a~        a~        \    a~ //
```

Uttrycket ser komplicerat ut. Då vi förenklar det med `simple` får vi det betydligt angenämare uttrycket

```
I =
1/(a^2+1)
```

(b) Betrakta integralen

$$\int_0^1 \frac{1}{\sqrt{x}+x}\,dx.$$

I detta fallet växer integranden obegränsat då vi närmar oss 0 från höger. Den generaliserade integralen beräknas med

```
syms x
int(1/(sqrt(x)+x),x,0,1)
```

och svaret blir

```
ans =
2*log(2)
```

□

**Exempel 5.3.** Betrakta rotationskroppen genererad av funktionskurvan $y = \cos x$, $0 \le x \le \pi/4$ (jfr avsnitt 13.2HB och 13.3HB). Funktionskurvans längd ges av

```
syms x
y = cos(x);
integrand = sqrt(1+diff(y,x)^2);
L = int(integrand,x,0,pi/4);
pretty(L)
```

MATLAB returnerar

```
 1/2                 1/2     1/2                 1/2       1/2
2    EllipticE(1/2 2   ) - 2    EllipticE(1/2 2   , 1/2 2   )
```

Vi får längden uttryckt i termer av specialfunktionen `EllipticE`. För att omvandla till ett numeriskt värde skriver vi `double(L)` vilket ger $L = 0.8520$.

Rotationskroppens volym och mantelyta beräknas genom

```
syms x
y = cos(x);
V = pi*int(y^2,x,0,pi/4);
pretty(V)
integrand = y*sqrt(1+diff(y,x)^2);
Y = 2*pi*int(integrand,x,0,pi/4);
pretty(Y)
```

vilket ger

```
pi (1/4 + 1/8 pi)

                 1/2  1/2                          1/2    1/2  1/2
1/4 pi 2    6    - pi log(2) + pi log(2    + 2    3   )
```

Önskas värdena på decimalform kan kommandot `double` användas. □

## 5.2 Multipelintegraler

Integrationskommandot ger oss ett kraftfullt verktyg för att beräkna multipelintegraler.

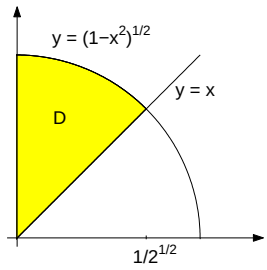
**Exempel 5.4.**

(a) Beräkna

$$I = \iint_D \frac{y}{1+\sqrt{2}\,x}\,dxdy,$$

där $D$ är området definierat av olikheterna $0 \le x \le y \le \sqrt{1-x^2}$.

Vi börjar med att plotta integrationsområdet. Från plotten ser man att integralen kan beräknas genom itererad integration på följande sätt




$$I = \int_{x=0}^{1/\sqrt{2}} \left( \int_{y=x}^{\sqrt{1-x^2}} \frac{y}{1+\sqrt{2}\,x}\,dy \right) dx.$$

Uppgiften löses med följande kommandon

```
syms x y
Iy = int(y/(1 + sqrt(2)*x),y,x,sqrt(1-x^2));
I = int(Iy,x,0,1/sqrt(2))
```

och vi får resultatet

```
I =
1/8*2^(1/2)
```

(b) Vi ska beräkna dubbelintegralen

$$I = \iint_D (x-R)y^2\,dxdy,$$

där $D$ är cirkelskivan med medelpunkt i origio och radie $R$.

Vi har $x^2 + y^2 = R^2 \Leftrightarrow y = \pm\sqrt{R^2 - x^2}$ och integralen fås genom

$$I = \int_{x=-R}^{x=R} \left( \int_{y=-\sqrt{R^2-x^2}}^{y=\sqrt{R^2-x^2}} (x-R)y^2 \, dy \right) dx$$

Då vi skriver

```
syms x y
syms R positive
Iy = int((x-R)*y^2,y,-sqrt(R^2-x^2),sqrt(R^2-x^2));
I = int(Iy,x,-R,R)
```

svarar MATLAB

```
I =
-1/4*R^5*pi
```

□

**Exempel 5.5.** Vi ska titta på hur man kan göra variabelbyten. Vi har trippelintegralen

$$\iiint_D \frac{x^2y^2}{1 + x^2 + y^2 + z^2} \, dxdydz,$$

där $D$ är klotet med medelpunkt i origo och radie $R$.

Vi övergår till rymdpolära integraler

$$\begin{cases} x = r \sin\theta \cos\varphi \\ y = r \sin\theta \sin\varphi \\ z = r \cos\theta \end{cases}$$

varvid integrationsområdet blir $0 \le r \le R$, $0 \le \theta \le \pi$, $0 \le \varphi \le 2\pi$. Integralen beräknas med följande kommandon (notera hur vi multiplicerar integranden med $r^2 \sin\theta$ för att kompensera för areaförändringen vid variabelbytet)

```
syms x y z r theta phi
syms R positive
a = r*sin(theta)*cos(phi);
b = r*sin(theta)*sin(phi);
c = r*cos(theta);
f = subs(x^2*y^2/(1+x^2+y^2+z^2),{x,y,z},{a,b,c})
Iphi =  int(f*r^2*sin(theta),phi,0,2*pi);
Iphitheta = int(Iphi,theta,0,pi);
I = int(Iphitheta,r,0,R);
pretty(I)
```

och vi får resultatet

```
        5                  3
4/75 R~  pi - 4/45 pi R~  + 4/15 pi R~ - 4/15 pi atan(R~)
```

□

## 5.3 Ordinära differentialekvationer

En ordinär differentialekvation (ODE) anger ett samband mellan en funktion och dess derivator. För att få en entydig lösning måste man lägga på villkor på lösningen i en eller flera punkter. Om man har villkor på lösningen och dess derivator i en enda punkt brukar man prata om ett begynnelsevärdesproblem.

| | |
|---|---|
| `dsolve('ekv', 'villkor',t)` | löser den ordinära differentialekvationen given i *ekv*. Defaultvärdet för den oberoende variabeln är $t$. Derivering med avseende på den oberoende variabeln betecknas med `D`. Högre ordningens derivator betecknas med `D` följt av ett tal som ger ordningen. Då ingen explicit lösning till differentialekvationen hittas försöker programmet bestämma en lösning på implicit form. Vid lösning på implicit form fås alltid ett varningsmeddelande. Om ingen explicit eller implicit lösning hittas returneras en tom symbolisk variabel. Villkor på lösningen ges i *villkor* genom ekvationer av typen `y(a)=b` eller `Dy(a)=b` där $a$ och $b$ är konstanter. |

**Exempel 5.6.**

(a) Den allmänna lösningen till differentialekvationen

$$y' + y = xe^x$$

bestäms genom kommandona

```
syms x
y = dsolve('Dy+y=x*exp(x)','x')
```

och vi får

```
y =
1/2*x*exp(x)-1/4*exp(x)+exp(-x)*C1
```

Här är *C1* en godtycklig konstant. För att plotta lösningen då $-1 \leq x \leq 3$ och med *C1* $= 2$ skriver vi

```
C1 = 2; y = subs(y);
xv = linspace(-1,3);
yv = subs(y,x,xv);
plot(xv,yv), xlabel('x'), ylabel('y')
```

vilket ger plotten till vänster i figur 5.1. Notera att vi här är tvungna att använda substitutionskommandot två gånger: en gång för att ersätta *C1* och en gång för att ersätta $x$ med en vektor med värden mellan $-1$ och 3.

(b) Vi har följande ekvation för en odämpad oscillation med en extern periodiskt varierande kraft

$$y'' + 144\,y = \cos(11t)$$

där lösningen skall uppfylla $y(0) = y'(0) = 0$. Differentialekvationen med villkor löses genom

```
syms t
y = dsolve('D2y+144*y=cos(11*t)','y(0)=0','Dy(0)=0','t')
```

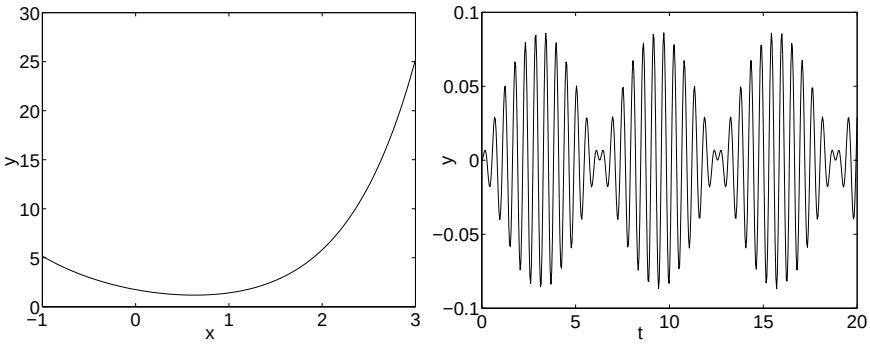
MATLAB ger utskriften

```
y =
-1/23*cos(12*t)+1/23*cos(11*t)
```

Lösningen plottas i intervallet $[0, 20]$ genom

```
tv = linspace(0,20,2000);
yv = subs(y,t,tv);
plot(tv,yv), xlabel('t'), ylabel('y')
```

vilket ger plotten till höger i figur 5.1. □




Figur 5.1: *Lösningar till ekvationerna* $y' + y = xe^x$ *och* $y'' + 144y = \cos(11t)$.

**Exempel 5.7.**

(a) Vi har differentialekvationen

$$y' + y = \delta(t).$$

där lösningen skall uppfylla $y(-1) = 0$. Högerledet ges av en Diracfunktion $\delta(t)$ som har egenskapen att den är noll för alla $t$ utom för $t = 0$, där den är oändlig. Diracfunktionen är ingen funktion i vanlig bemärkelse utan en så kallad distribution. Då vi skriver

```
syms t
y = dsolve('Dy + y = dirac(t)','y(-1)=0','t')
```

får vi

```
y =
exp(-t)*heaviside(t)
```

Lösningen är plottad till vänster i figur 5.2.

(b) Betrakta differentialekvationen

$$y' + y = \text{heaviside}(t)\, e^{-t}.$$

med villkoret $y(-1) = 0$. Högerledet kan ses som en impuls som slås på vid $t = 0$ och som sedan dör ut exponentiellt. Differentialekvationen löses med

```
syms t
y = dsolve('Dy + y = heaviside(t)*exp(-t)','y(-1)=0','t')
```

och MATLAB svarar

```
y =
exp(-t)*heaviside(t)*t
```

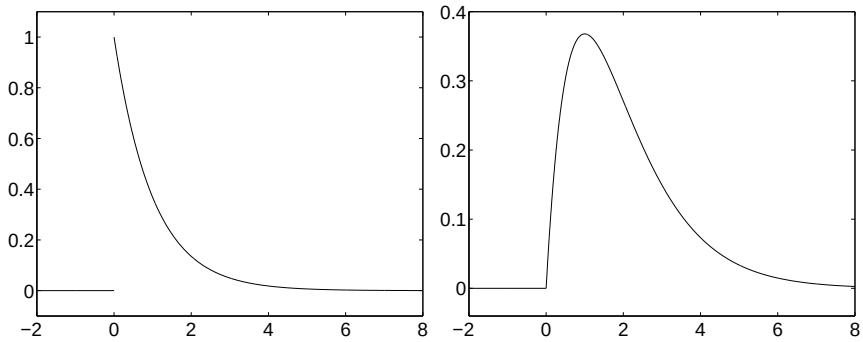
Lösningen finns plottad till höger i figur 5.2. □

**Exempel 5.8.** Ibland kan MATLAB inte finna explicita lösningar (den beroende variabeln ges som ett explicit uttryck av den oberoende variabeln). I en del fall fås i stället lösningen på implicit form. Betrakta följande begynnelsevärdesproblem

$$(1 - \sin y)\, y' = x, \qquad y(0) = 0$$

För att lösa begynnelsevärdesproblem skriver vi

```
syms x
y = dsolve('(1-sin(y))*Dy=x','y(0)=0','x')
```




Figur 5.2: *Lösningar till* $y' + y = \delta(t)$ *och* $y' + y = heaviside(t)e^{-t}$.

och får svaret

```
??? Error using ==> dsolve
Error, (in dsolve/IC) The 'implicit' option is not available
when giving Initial Conditions.
```

MATLAB inte kunde finna en explicit lösning utan gick vidare för att finna en implicit lösning. Implicita lösningar kan dock inte bestämmas då man ger villkor på lösningen. Vi testar därför att bestämma en lösning utan villkor

```
syms x
dsolve('(1-sin(y))*Dy=x','x')
```

och får svaret

```
Warning: Explicit solution could not be found; implicit
solution returned.
> In dsolve at 312
ans =
1/2*x^2-y-cos(y)+C1 = 0
```

Den allmänna lösningen till differentialekvationen ges alltså av det implicita uttrycket

$$\frac{1}{2}x^2 - y - \cos(y) + C1 = 0.$$

För att begynnelsevillkoret $y(0) = 0$ skall vara uppfyllt ser man enkelt att konstanten *C1* måste vara 1. Den implicita lösningen kan plottas med kommandot `contour` (jfr avsnitt 5.9HB). □

## 5.4 System av ordinära differentialekvationer

För att lösa system av ordinära differentialekvationer i MATLAB kan man som tidigare använda kommandot `dsolve`.

| | |
|---|---|
| `dsolve('ekv1','ekv2', ...,'villkor',t)` | löser systemet av differentialekvationen givet i *ekv1, ekv2,...*. Defaultvärdet för den oberoende variabeln är $t$. Den oberoende variabeln kan ändras från $t$ till någon annan symbolisk variabel genom att skriva denna variabel inom citationstecken som sista argument.<br>Villkor på lösningen ges i *villkor* genom ekvationer av typen `y(a)=b` eller `Dy(a)=b` där $a$ och $b$ är konstanter. |

**Exempel 5.9.**

(a) Vi har följande system av differentialekvationer

$$\begin{cases} y_1' = -2y_1 - 3y_2 \\ y_2' = 3y_1 - 2y_2 \end{cases}$$

För att bestämma lösningen ger vi kommandona

```
syms t
[y1,y2] = dsolve('Dy1=-2*y1-3*y2','Dy2=3*y1-2*y2','t')
```

och MATLAB svarar med

```
y1 =
exp(-2*t)*(C1*sin(3*t)+C2*cos(3*t))
y2 =
-exp(-2*t)*(C1*cos(3*t)-C2*sin(3*t))
```

(b) Betrakta följande system av tre kopplade differentialekvationer

$$\begin{cases} y_1' = y_2 \\ y_2' = y_3 \\ y_3' = y_1 \end{cases}$$

med begynnelsevillkoren $y_1(0) = 1$, $y_2(0) = 0$, $y_3(0) = 0$. Systemet löses med

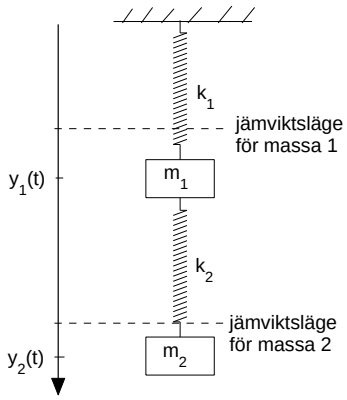
```
syms t
ekv1 = 'Dy1 = y2'; ekv2 = 'Dy2 = y3'; ekv3 = 'Dy3 = y1';
v1 = 'y1(0) = 1';  v2 = 'y2(0) = 0';  v3 = 'y3(0) = 0';
[y1,y2,y3] = dsolve(ekv1,ekv2,ekv3,v1,v2,v3,'t')
```

vilket ger

```
y1 =
1/3*exp(t)+2/3*exp(-1/2*t)*cos(1/2*3^(1/2)*t)
y2 =
1/3*exp(t)-1/3*exp(-1/2*t)*cos(1/2*3^(1/2)*t) ...
            -1/3*exp(-1/2*t)*sin(1/2*3^(1/2)*t)*3^(1/2)
y3 =
1/3*exp(t)-1/3*exp(-1/2*t)*cos(1/2*3^(1/2)*t) ...
            +1/3*exp(-1/2*t)*sin(1/2*3^(1/2)*t)*3^(1/2)
```

□

**Exempel 5.10.** Vi kan även ha system av högre ordningens differentialekvationer. Vi har två kopplade fjädrar enligt figur 5.3 (dämpningen försummas).




Figur 5.3: *Kopplade fjädrar.* $y_1(t)$ *är avståndet till jämviktsläget för massa 1 och* $y_2(t)$ *är avståndet till jämviktsläget för massa 2.*

Enligt Hooks lag är den återställande kraften från en fjäder proportionell mot uttänjningen. Kraften på massa 1, som påverkas av båda fjädrarna, är

$$F_1 = -k_1 y_1 + k_2(y_2 - y_1).$$

Kraften på massa 2, som endast påverkas av den andra fjädern, är

$$F_2 = -k_2(y_2 - y_1).$$

Den totala kraften är lika med massan gånger accelerationen och vi har

$$\begin{cases} m_1 y_1'' = -k_1 y_1 + k_2(y_2 - y_1) \\ m_2 y_2'' = -k_2(y_2 - y_1) \end{cases}.$$

Som ett konkret exempel tar vi $m_1 = 1$ och $m_2 = 1$ samt fjäderkonstanter $k_1 = 1$ och $k_2 = 1$. Vi löser differentialekvationerna för begynnelsevärdena $y_1(0) = 0$, $y_1'(0) = 0$, $y_2(0) = 1$, $y_2'(0) = 0$.

```
syms t
ekv1 = 'D2y1 = - y1 + (y2-y1)';
ekv2 = 'D2y2 = -(y2-y1)';
v1 = 'y1(0)=0'; v2 = 'Dy1(0)=0';
v3 = 'y2(0)=1'; v4 = 'Dy2(0)=0';
[y1,y2] = dsolve(ekv1,ekv2,v1,v2,v3,v4,'t');
y1 = simple(y1); y2 = simple(y2);
pretty(y1), pretty(y2)
```

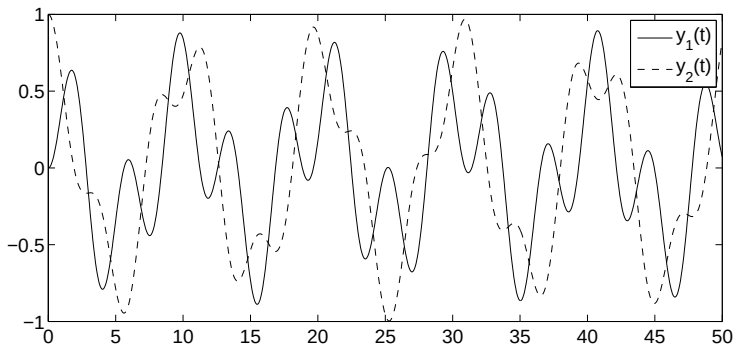
och MATLAB svarar med

```
     1/2            1/2
2/5 5   sin(1/2 t 5   ) sin(1/2 t)

     1/2            1/2                                1/2
1/5 5   sin(1/2 t 5   ) sin(1/2 t) + cos(1/2 t 5   ) cos(1/2 t)
```

Svängningarna plottas i tidsintervallet $[0, 50]$ genom kommandona

```
tv = linspace(0,50,1000);
y1v = subs(y1,t,tv); y2v = subs(y2,t,tv);
plot(tv,y1v,tv,y2v,'--'), legend('y_1(t)','y_2(t)')
```

Den genererade plotten visas i figur 5.4. □




Figur 5.4: *Svängningar för kopplat fjädersystem.*

## 5.5 Vidare läsning

Jönsson, P. (2008), Matematik med datoralgebrasystem, Studentlitteratur, ISBN 978-91-44-05250-2

Integraler behandlas i kapitel 11 med ordinära differentialekvationer tas upp i kapitel 12.

Persson, A. och Böiers L.-C, A (2001), Analys i en variabel, Studentlitteratur, ISBN 91-44-02056-2

Integraler och differentialekvationer behandlas i kapitel $6, 7$ och 8.

Persson, A. och Böiers L-C, A (2005), Analys i flera variabler, Studentlitteratur, ISBN 91-44-03869-0

Multipelintegraler och variabelbyten beskrivs i kapitel 6–8.

## 5.6 Instuderingsfrågor

1. Hur beräknar man den obestämda integralen av $f(x)$?
2. Hur beräknar man integralen av $f(x)$ över intervallet $[a, b]$?
3. Vad är en generaliserad integral?
4. Vilken geometrisk tolkning har en dubbelintegral $\iint_D f(x, y)\, dxdy$?
5. Hur skriver man för att beräkna integralen $\displaystyle\int_{x=0}^{x=1} \left( \int_{y=-x}^{y=x} f(x, y)\, dy \right) dx$?
6. Vilket är kommandot för att lösa en ordinär differentialekvation? Hur betecknar man derivatan i differentialekvationen?
7. Vad menas med implicit lösning till en differentialekvation?
8. Hur kan man plotta en implicit lösning?
9. Hur löser man ett system av differentialekvationer?

## 5.7 Övningar

1. Bestäm integralerna

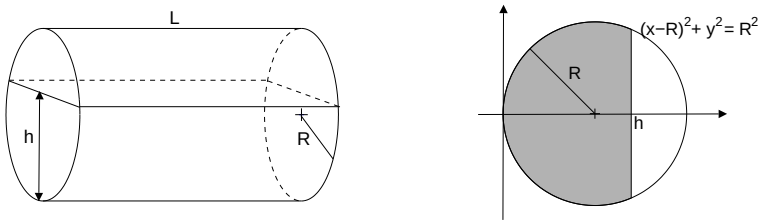
   (a) $\displaystyle\int \frac{ax + b}{x^2 + 5x + 6}\, dx$, (b) $\displaystyle\int \ln(x)^2\, dx$, (c) $\displaystyle\int_0^{\pi/4} \frac{1 - \tan(x)}{1 + \tan(x)}\, dx$

   (d) $\displaystyle\int_0^{\pi/8} \frac{dx}{1 + \sin(x)\cos(x)}$

2. Bestäm de generaliserade integralerna
   (a) $\displaystyle\int_1^\infty \frac{dx}{e^x - e^{-x}}$, (b) $\displaystyle\int_1^\infty \frac{\ln(2x-1)}{x^2}\,dx$ (c) $\displaystyle\int_0^\infty \frac{1}{1+e^{ax}}\,dx$, $a > 0$

3. Kurvan $y = 2\sqrt{x}$, $0 \le x \le 3$ roterar kring $x$-axeln. Bestäm volymen och mantelytans area för rotationskroppen.

4. Vi har en oljetank med radie $R$ och längden $L$ enligt figuren. Hur stor volym olja innehåller tanken då oljenivån går upp till $h$? Ledning: betrakta figuren nedan till höger.




5. Bestäm integralerna
   (a) $\displaystyle\iint_D \frac{dxdy}{(x^2+y^2)^2}$ där $D = \{(x,y)|\ 1 \le x \le 2,\ 0 \le y \le x\}$.
   (b) $\displaystyle\iint_D \frac{dxdy}{1+x+y}$ där $D = \{(x,y)|\ 0 \le y \le 1,\ y \le x \le 2y\}$.

6. Låt $N(t)$ beteckna antalet individer i en population vid tiden $t$. Enligt den logistiska modellen för populationstillväxt är

$$\frac{dN}{dt} = r\left(1 - \frac{N}{K}\right)N.$$

   Här är $r$ populationens födelseöverskott per individ och år (tillväxten per individ). $K$ bärarkapaciteten som talar om hur många individer miljön kan upprätthålla.

   a) Lös differentialekvationen under förutsättning att det finns $N_0$ individer vid $t = 0$.

   b) Bestäm tiden $t$ för vilken vi har maximal tillväxthastighet. Ledning; derivera lösningen $N(t)$ två gånger och sätt derivatan lika med noll.

   c) Hur stor är den maximala tillväxthastigheten uttryckt i $K$ och $r$?

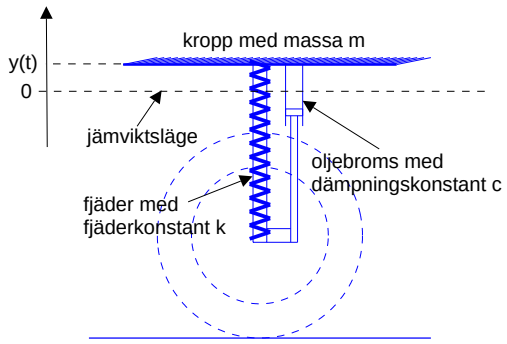
   d) Hur stor är populationen (uttryckt i $K$ och $r$) då tillväxthastigheten är maximal?

7. Vi har en kropp med temperaturen $T$ som befinner sig i ett omgivande medium med konstant temperatur $\tau$. Enligt Newtons avsvalningslag gäller att temperaturändringen per tidsenhet är proportionell mot temperaturdifferensen

$$\frac{dT}{dt} = -k(T - \tau).$$

Lös differentialekvationen. Antag att kroppen har temperaturen $T_0$ vid $t = 0$. Beräkna gränsvärdet av lösningen då $t \to \infty$.

8. Vi ska nu sätta upp och analysera några modeller för mekaniska svängningar. För den sakens skull tänker vi oss en kropp (bilchassit) med massa $m$ fäst i en fjäder enligt figuren 5.5




Figur 5.5: *Dämpat fjädersystem på en bil.*

Kroppen är kopplad till en viskös dämpare (trögflytande olja) vilken utövar en kraft som motverkar rörelsen. Fjädern och dämparen utgör ett så kallat dämpat fjädersystem. Dämpade fjädersystem har bland annat tillämpningar som stötdämpare i bilar. Låt $y(t)$ vara bilchassits position i förhållande till fjädersystemets jämviktsläge. Kvantiteterna $y'(t)$ och $y''(t)$ kan då tolkas som chassits vertikala hastighet och acceleration. Enligt Hooks lag är den återställande kraften från fjädern proportionell mot avståndet $y(t)$ till jämviktsläget och vi har

$$F_{\text{fjäder}} = -ky,$$

där den positiva konstanten $k$ är den så kallade fjäderkonstanten. Vi har ett minustecken framför eftersom kraften är motsatt riktad i förhållande till $y$. Noggranna experiment visar att kraften från dämparen är proportionell mot hastigheten $y'(t)$ och vi har

$$F_{\text{dämpare}} = -cy',$$

där den positiva konstanten $c$ är systemets dämpningskonstant. Den dämpande kraften är motsatt riktad i hastigheten. I det allmänna fallet kan vi också tänka oss att bilen också påverkas av en extern kraft $F_{\text{ext}}$. Enligt Newtons andra lag är kraftsumman $F$ lika med massan gånger accelerationen och vi har

$$\underbrace{-ky}_{F_{\text{fjäder}}} \underbrace{-cy'}_{F_{\text{dämpare}}} + F_{\text{ext}} = my''.$$

Omordning ger

$$my'' + cy' + ky = F_{\text{ext}}.$$

Vi startar med fallet att den externa kraften är noll.

(a) Lös ekvationen i fallet då dämpning saknas ($c = 0$). Vilken är systemets egenfrekvens?

(b) Lös ekvationen då vi har dämpning.

Vi tänker oss nu att bilen kör över ojämn mark så att den påverkas av en extern periodisk kraft $F_{\text{ext}} = F_0 \cos(at)$.

(c) Låt dämpningen $c$ vara noll. Lös ekvationen då den externa frekvensen $a$ är lika stor som systemets egenfrekvens (resonans) och skild från systemets egenfrekvens. Hur skiljer sig lösningarna åt?

(d) Låt dämpningen $c$ vara noll. Tag $m = k = F_0 = 1$ och låt $y(0) = y'(0) = 0$. Plotta lösningarna i uppgift (c) både då den externa frekvensen är lika med och då den skiljer sig från egenfrekvensen. Studera speciellt vad som händer då den externa frekvensen kommer nära egenfrekvensen (svävning).

(e) Gör om uppgift (d) men nu med $c = 0.1$.

En bils fjädersystem justeras så att bilen mjukt går tillbaka till jämviktsläget då man kör över ett gupp. Vi ska simulera ett gupp genom att låta den externa kraften $F_{\text{ext}}$ ha ett konstant värde $F_0$ i tidsintervallet $[t_1, t_2]$ medan den är noll för övrigt. Kraften överför en vertikal impuls $F_0(t_2 - t_1)$ till bilen. En funktion som har värdet $F_0$ i $[t_1, t_2]$ men som är noll för övrigt fås som

$$F_{\text{ext}} = F_0(\text{heaviside}(t - t_1) - \text{heaviside}(t - t_2)).$$

(f) Låt kraften verka i tidsintervallet $[1, 2]$. Tag $m = k = F_0 = 1$ och låt $y(0) = y'(0) = 0$. Lös ekvationen differentialekvationen för lite olika positiva värden på $c$ och plotta lösningen. Hur tycker du att man bör välja $c$ för att få ett vettigt fjädersystem?

9. Betrakta systemet av differentialekvationer

$$\begin{cases} y_1'(t) &= -4y_1(t) + 5y_2(t) \\ y_2'(t) &= -4y_1(t) + 4y_2(t) \end{cases}$$

(a) Skriv upp den allmänna lösningen till systemet.

(b) Bestäm lösningen som uppfyller $y_1(0) = -1$ och $y_2(0) = 1$.

(c) Plotta lösningen för $0 \leq t \leq 10$ (jfr uppgift 14 i kapitel 9HB).

10. Vi ska titta på en radioaktiv sönderfallskedja. Isotopen $^{210}$Bi är radioaktivt och sönderfaller under $\beta$ strålning till $^{210}$Po vilken i sin sönderfaller under $\alpha$ strålning till den stabila isotopen $^{206}$Pb. Låt $x_1(t)$ och $x_2(t)$ beteckna mängden av $^{210}$Bi respektive $^{210}$Po. Då gäller att

$$\begin{cases} \underbrace{\dfrac{dx_1}{dt}}_{\substack{\text{ändring av } x_1 \\ \text{per tidsenhet}}} &= - \underbrace{\lambda_1 x_1}_{\substack{\text{antal sönderfall} \\ \text{av } x_1 \text{ per tidsenhet}}} \\ \underbrace{\dfrac{dx_2}{dt}}_{\substack{\text{ändring av } x_2 \\ \text{per tidsenhet}}} &= \underbrace{\lambda_1 x_1}_{\substack{\text{antal bildade} \\ x_2 \text{ per tidsenhet}}} - \underbrace{\lambda_2 x_2}_{\substack{\text{antal sönderfall} \\ \text{av } x_2 \text{ per tidsenhet}}}. \end{cases}$$

Antag att vi har 1 mol $^{210}$Bi vid tiden $t = 0$ och att mängden $^{210}$Po är noll.

(a) Lös ekvationssystemet.

(b) Vi har $\lambda_1 \approx 0.138$ och $\lambda_2 \approx 0.0050090$. Sätt in dessa värden och och plotta lösningarna i intervallet $[0, 200]$.

(c) Bestäm tiden för vilken mängden $^{210}$Po är maximal.

(d) Plotta det totala antalet sönderfall per tidsenhet (antalet sönderfall från $^{210}$Bi och $^{210}$Po) i tidsintervallet $[0, 200]$. Plotta först med vanlig skala och sen med logaritmisk skala på $y$-axeln.

# Bilaga A

# Lösningar till valda övningar

## Kapitel 1

```
3.  (a)   syms x
         expand((x^3+2*x^2-x-5)^3)

4.  (b)   syms x
         solve(x^3-x^2-x/4+1/4,x)

5.  syms x
    f = x^2*exp(x)
    p = taylor(f,2,1)
    xv = 0:0.01:2;
    fv = subs(f,x,xv);
    pv = subs(p,x,xv);
    plot(xv,fv,xv,pv,'--')
    legend('y = x^2e^x','tangent')

6.  syms t y
    y = dsolve('Dy = 100-y','y(0)=20','t')
    tv = 0:0.01:10;
    yv = subs(y,t,tv);
    plot(tv,yv)
    xlabel('t'), ylabel('y')

14. syms a b x
    y = a*x/(1 + b*x);
    y = subs(y,{a b},{3 2})
    xv = 0:0.01:5;
```

```
    yv = subs(y,x,xv);
    plot(xv,yv)

17. syms t
    tv = -1:0.01:1;
    for k = 1:5
       p(k)=expand((1/(2^k*prod(1:k)))*diff((t^2-1)^k,t,k));
       disp(p(k))
       pv = subs(p(k),t,tv);
       plot(tv,pv), hold on
    end

    for k = 1:5
       for l = 1:5
          integ = int(p(k)*p(l),t,-1,1);
          disp([k l integ])
       end
    end

18. syms t
    P0 = sym('1');
    P1 = t;
    for k = 2:5
       P2 = simplify(((2*k-1)*t*P1 - (k-1)*P0)/k);
       disp(k), disp(P2)
       P0 = P1;
       P1 = P2;
    end
```

## Kapitel 2

```
1.  syms x a b c
    p = collect((x-1/a-1/b)*(x-1/b-1/c)*(x-1/c-1/a),x);
    pretty(p)

2.  syms x
    f = x^4+4*x^3-17*x^2+20*x-12; g = x^2-5*x+3;
    collect(f*g)

3. (d) syms x
       expand(sin(3*x))
```

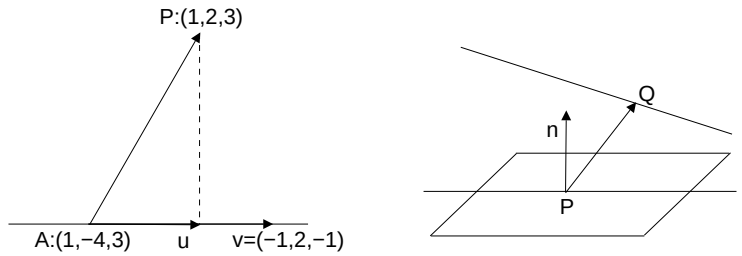
```
4.  f0 = 0; f1 = 1;
    for k = 2:20
       f2 = f1 + f0;
       f0 = f1;
       f1 = f2;
       disp([k f2])
    end

    a = sym('sqrt(5)');
    for k = 0:20
       f = simple((1/a)*( ((1+a)/2)^k - ((1-a)/2)^k ));
       disp([k f])
    end

7.  factor(sym('(3 + sqrt(5))/(2 + sqrt(5))'))

    factor(sym('(1+2*sqrt(2))/(3-sqrt(2))'))

    factor(sym('1/(sqrt(13) + sqrt(11))'))

8. (b) syms x
       l = solve(sqrt(x + 2) - x,x)

9.  syms V0 R a r positive
    V = - V0/(1 + exp((r-R)/a));
    r1 = solve(V + 0.1*V0,r)
    r2 = solve(V + 0.9*V0,r)
    r1 - r

    rv = 0:0.01:10;
    V = subs(V,{V0 R a},{1 5 0.5});
    Vv = subs(V,r,rv);
    plot(rv,Vv)

10. % vi kallar hastigheterna efter kollisionen för u1 och u2
    syms m M v1 v2 u1 u2
    % bevaring av rörelsemängd
    ekv1 = m*v1 + M*v2 - (m*u1 + M*u2);
    % bevaring av energi
    ekv2 = m/2*v1^2 + M/2*v2^2 - (m/2*u1^2 + M/2*u2^2);
    [u1,u2] = solve(ekv1,ekv2,u1,u2)

    Vi får två lösningar. Den första är ointressant och
```

```
beskriver fallet då det inte blev någon stöt (v2 > v1).
```

# Kapitel 3

```
1.  syms x1 x2 x3 x4 x5
    s1 = x1 - x2 + 2*x3 + x4 + 3*x5 - 1;
    s2 = 2*x1 + x3 + 2*x4 + 4*x5 + 2;
    s3 = x1 + 2*x2 - 2*x3 + 3*x5 + 3;
    [x1,x2,x3,x4,x5] = solve(s1,s2,s3,x1,x2,x3,x4,x5)
    % lösningen beror av två parametrar

2.  syms a x1 x2 x3
    s1 = x1 + x2 + x3;
    s2 = x1 + 2*x2 + a*x3 - 1;
    s3 = x1 + a*x2 + 2*x3 + 1;
    [x1,x2,x3] = solve(s1,s2,s3,x1,x2,x3)
    % lösning saknas då a = 2

3.  syms x y z
    n = [1 2 -3]; n = sym(n);
    A = [-2 -1 4]; A = sym(A);
    P = [x y z];
    AP = P - A;
    dot(n,AP)

4.  A = [-1 -3 -4]; A = sym(A);
    B = [-5 -5 1];  B = sym(B);
    C = [3 1 2]; C = sym(C);
    AB = B - A; AC = C - A;
    n = cross(AB,AC);   % normalvektor till planet
    P = [1 -2 3]; P = sym(P);
    u = P-A;
    up = (dot(u,n)/dot(n,n))*n;
    Q = P-up, S = P-2*up

5.  Vi inför beteckningar enligt figur A.1.
    A = (1,-4,3) punkt på linjen
    v = (-1,2,-1) linjens riktningsvektor
    Vektorn u fås genom att projicera AP på v. Sökt
    avstånd till linjen fås som längden av vektorn P-U

    A = [1 -4 3]; A = sym(A);
```

```
v = [-1 2 -1]; v = sym(v);
P = [1 2 3]; P = sym(P);
AP = P - A;
u = (dot(AP,v)/dot(v,v))*v;
sqrt(dot(P-u,P-u))
```




Figur A.1: *Avstånd mellan punkt och linje och avstånd mellan två linjer.*

6. Betrakta figur A.1.
   Vi konstruerar ett plan som är parallellt med båda linjerna.
   Planets normalvektor ges av n = v1 x v2 där v1 och v2 är
   linjernas riktningsvektorer. Vi tar sedan och projicerar
   PQ på n. Avståndet fås som längden av den projicerade vektorn.

```
v1 = [0 1 1]; v1 = sym(v1);
v2 = [2 3 1]; v2 = sym(v2);
P = [1 2 3]; P = sym(P);
Q = [1 1 1]; Q = sym(Q);
n = cross(v1,v2); PQ = Q - P;
u = (dot(PQ,n)/dot(n,n))*n;
sqrt(dot(u,u))
```

7.
```
syms u1 u2 u3 v1 v2 v3 w1 w2 w3 real
u = [u1 u2 u3]; v = [v1 v2 v3]; w = [w1 w2 w3];
simplify( cross(cross(u,v),w) - (dot(u,w)*v - dot(v,w)*u) )
```

8.
```
syms u1 u2 u3 v1 v2 v3 w1 w2 w3 real
u = [u1 u2 u3]; v = [v1 v2 v3]; w = [w1 w2 w3];
a = cross(u,v) - cross(v,w);
simplify( subs(a,{u1 u2 u3},{-v1-w1 -v2-w2 -v3-w3}) )
```

De andra likheterna visas på samma sätt

11.
```
t = sym('theta')
A = [cos(t) sin(t) ; - sin(t) cos(t)];
for i = 1:10
   disp(i), disp(simple(A^i))
end
```

från resultatet är det inte svårt att gissa sig till formeln

13.
```
syms a b c d
A = [a b ; c d]
eig(A)
```

14.
```
clear all
syms a b real
A = [1 a+b*i ; a-b*i 1];
eig(A)
```

15.
```
function U = gramschmidtpolynom(V)
% modifierad med w(t) = (1-t^2)^(-1/2)
syms t
U(1) = V(1);
n = length(V)
for i = 2:n
  U(i) = V(i);
  for j = 1:i-1
    A = int(V(i)*U(j)*(1-t^2)^(-1/2),-1,1);
    B = int(U(j)*U(j)*(1-t^2)^(-1/2),-1,1);
    U(i) = U(i) - (A/B)*U(j);
  end
end
% skala
for i = 2:n
  A = sqrt(sym('pi')/2);
  B = sqrt(int(U(i)^2*(1-t^2)^(-1/2),-1,1));
  U(i) = A*U(i)/B;
end

% huvudprogram
syms t
V = [1 t t^2 t^3 t^4 t^5];
U = gramschmidtpolynom(V);
pretty(U)
```

```
tv = linspace(-1,1);
for i = 1:6
  UV = subs(U(i),t,tv)
  plot(tv,UV), hold on
end
```

# Kapitel 4

```
1.  x = 0:0.01:10;
    y = mfun('dawson',x);
    plot(x,y)

2. (a) syms x
       limit((1 - 1/(2*x) - 1/(2*x^2))^(2*x),x,inf)

   (d) syms x
       limit(x^x,x,0,'right')

3. (a) syms x
       f = 3-x/3+log(exp(x)+1);
       k1 = limit(f/x,x,inf)
       m1 = limit(f - k1*x,inf)
       k2 = limit(f/x,x,-inf)
       m2 = limit(f - k2*x,-inf)
       k1 = double(k1); m1 = double(m1);
       k2 = double(k2); m2 = double(m2);
       xv = -4:0.01:4;
       fv = subs(f,x,xv);
       plot(xv,fv,xv,k1*xv+m1,'--',xv,k2*xv+m2,'--')

4. (a) syms x
       f = sin(x)/(cos(x) + x^3)
       pretty(diff(f,x)), pretty(diff(f,x,2))

5.  syms x
    f = -1 + x/(x + sin(x));
    df = diff(f,x); df2 = diff(f,x,2);
    xv = 0:0.01:10;
    fv = subs(f,x,xv);
    dfv = subs(df,x,xv);
```

```
df2v = subs(df2,x,xv);
plot(xv,fv,xv,dfv,'--',xv,df2v,':')
legend('f','Df','D^2f')
```

6. (a)
```
syms a b x y
f = exp(a*x^2 + b*y^2 + 2*x*y)/(x^2+y^2);
gradf = jacobian(f,[x y]);
H = jacobian(gradf,[x y]);
pretty(gradf)
pretty(H)
```

```
I uttrycken står %1 för kvantiteten
       2      2
exp(a x  + b y  + 2 x y)
```

7.
```
syms r theta
x = r*cos(theta); y = r *sin(theta)
df = jacobian([x y],[r theta])
simplify(det(df))
```

8.
```
syms x
f = log(x);
xv = 0.5:0.01:2;
fv = subs(f,x,xv);
for i = 2:5
   p = taylor(f,i,1);
   pv = subs(p,x,xv);
   subplot(2,2,i-1)
   plot(xv,fv,xv,pv,'--')
   string = num2str(i-1,'%2.0f');
   title(['ln(x) och Taylorpolynom av grad ' string])
end
```

9. (a)
```
syms a x k n
pretty(symsum(a*x^k,k,5,n-1))
```

(b)
```
syms k n
pretty(symsum(k^2,k,0,n))
```

(c)
```
syms l
pretty(symsum(1/l^2,l,1,inf))
```

10.
```
function y = fun(x)
```

```
y=(x(1).^2+2*x(2).^2-1).^2+(4*(x(1)-0.4).^2+x(2).^2-1).^2;

[x,n] = optimering('fun',[0 0.7],1e-12)
```

11.
```
function [x1,n] = newtonsym(fun,x0,tol)
% Denna funktionsfil beräknar en approximativ rot till
% ekvationen f(x) = 0. fun innehåller namnet på
% funktionsfilen som beräknar f(x). I variabeln
% dfun lagras den symboliskt bestämda derivatan av f(x)
syms x             % x skall definieras symbolisk
f = feval(fun,x);  % anropa funktionsfilen fun
df = diff(f,x);    % beräkna derivatan, lagra i df
n = 0;
dx = 1e30;         % tag dx stort så vi börjar iterera
while abs(dx) > tol & n < 50
   dx =  subs(f,x,x0)/subs(df,x,x0);
   x1 = x0 - dx;
   x0 = x1;        % tag x1 som nytt startvärde
   n = n + 1;
end

function y = fun(x)
y = cos(x) - x;

[x,n] = newtonsym('fun',1,1e-12)
```

12.
```
function [x1,n] = newtonsyssym(fun,x0,tol)
% Denna funktionsfil beräknar en approximativ rot till
% ekvationssystemet  f(x) = 0. fun innehåller namnet på
% funktionsfilen som beräknar f(x). I variabeln
% dfun lagras den symboliskt bestämda funktionalmatrisen
syms x1 x2
x = [x1 x2]
f = feval(fun,x);   % anropa funktionsfilen fun
df = jacobian(f,x); % beräkna funktionalmatris, lagra i df
n = 0;
norm_dx = 1e30;
while norm_dx > tol & n < 50
   dx = subs(df,x,x0)\subs(f,x,x0);
   x1 = x0 - dx';      % OBS transponat
   x0 = x1;
   norm_dx = norm(dx,2);
   n = n + 1;
```

```
end
% Notera att i detta fallet så är subs(f,x,x0) en
% kolonnvektor så vi behöver ingen transponering här


function y = fun(x)
y = [x(1).^2 + x(2)
     x(1).^2 + x(2).^2 -1];

[x,n] = newtonsyssym('fun',[0.5  -0.5],1e-12)
```

# Kapitel 5

```
1. (a) syms a b x
       int((a*x+b)/(x^2+5*x+6),x)

   (c) syms x
       int((1-tan(x))/(1+tan(x)),x,0,pi/4)

2. (a) syms x
       int(1/(exp(x) - exp(-x)),x,1,inf)

   (c) syms x
       syms a positive
       int(1/(1+exp(a*x)),x,0,inf)

3. syms x
   y = 2*sqrt(x);
   V = pi*int(y^2,x,0,3)
   Y = 2*pi*int(y*sqrt(1+diff(y,x)^2),x,0,3)

4. syms R L h x positive
   A = int(2*sqrt(R^2 - (x-R)^2),x,0,h)
   V = A*L

5.(a) syms x y
      Iy = int(1/(x^2 + y^2)^2,y,0,x);
      I = int(Iy,x,1,2)

  (b) syms x y
      % Vi integrerar först i x-led
      Ix = int(1/(1+x+y),x,y,2*y);
```

```
    I = int(Ix,y,0,1)

6.  syms t N
    syms r K N0 positive
    N = dsolve('DN = r*(1-N/K)*N','N(0)=N0')
    dN = diff(N,t)
    dN2 = diff(N,t,2)
    tmax = solve(dN2,t)
    subs(dN,t,tmax)
    subs(N,t,tmax)

7.  syms T T0 tau t
    syms k positive
    T = dsolve('DT = -k*(T - tau)','T(0) = T0')
    limit(T,t,inf)

8.  syms y t
    syms a F0 m c k positive

   (a)
    dsolve('m*D2y + k*y = 0')

    %Systemets egenfrekvens är (k*m)^(1/2)/m

   (b)
    dsolve('m*D2y + c*Dy + k*y = 0')

   (c)
    % Resonans
    dsolve('m*D2y + k*y = F0*cos((k*m)^(1/2)/m*t)')

    % Inte resonans
    dsolve('m*D2y + k*y = F0*cos(a*t)')

    % Då vi har resonans växer amplituden på den sista
    % termen med t.

   (d)
    tv = 0:0.01:150;
    y = dsolve('D2y + y = cos(t)','y(0)=0','Dy(0)=0')
    yv = subs(y,t,tv);
    plot(tv,yv)
```

```
y = dsolve('D2y + y = cos(1.2*t)','y(0)=0','Dy(0)=0')
yv = subs(y,t,tv);
plot(tv,yv)

(e)
tv = 0:0.01:150;
y = dsolve('D2y+0.1*Dy+y = cos(t)','y(0)=0','Dy(0)=0')
yv = subs(y,t,tv);
plot(tv,yv)

y=dsolve('D2y+0.1*Dy+y=cos(1.2*t)','y(0)=0','Dy(0)=0')
yv = subs(y,t,tv);
plot(tv,yv)

(f)
tv = 0:0.01:50;

% c = 0
y = dsolve('D2y + y = heaviside(t-1) -
            heaviside(t-2)','y(0)=0','Dy(0)=0')
yv = subs(y,t,tv);
plot(tv,yv)

% c = 1
y = dsolve('D2y + 1*Dy + y = heaviside(t-1) - ...
            heaviside(t-2)','y(0)=0','Dy(0)=0')
yv = subs(y,t,tv);
plot(tv,yv)

% c = 5
y = dsolve('D2y + 5*Dy + y = heaviside(t-1) - ...
            heaviside(t-2)','y(0)=0','Dy(0)=0')
yv = subs(y,t,tv);
plot(tv,yv)
```

9.
```
syms t
[y1 y2] = dsolve('Dy1=-4*y1+5*y2','Dy2=-4*y1+4*y2')
[y1 y2] = dsolve('Dy1=-4*y1+5*y2','Dy2=-4*y1+4*y2',...
             'y1(0)=-1','y2(0)=1')
tv = 0:0.01:10;
y1v = subs(y1,t,tv); y2v = subs(y2,t,tv);
plot(tv,y1v,tv,y2v,'--')
```

```
10. syms t
    syms l1 l2 positive
    [x1 x2] = dsolve('Dx1 = -l1*x1','Dx2 = l1*x1 - l2*x2',...
                'x1(0)=1','x2(0)=0')
    x1 = subs(x1,{l1 l2},{0.138 0.005009});
    x2 = subs(x2,{l1 l2},{0.138 0.005009});
    % tid för Po maximal
    dx2 = diff(x2,t);
    tmax = double(solve(dx2,t))

    tv = 0:0.01:200;
    x1v = subs(x1,t,tv);
    x2v = subs(x2,t,tv);
    plot(tv,x1v,tv,x2v,'--')

    % Antalet sönderfall (både Bi och Po) fås som
    % l1*x1 + l2*x2
    sv = subs(0.138*x1 + 0.0050090*x2,t,tv);
    figure(1), plot(tv,sv)
    figure(2), semilogy(tv,sv)
```

# Sakregister

**MATLAB-kommandon**


`*`, 48
`+`, 48
`-`, 48
`abs`, 23
`bernoulli`, 63
`BesselI`, 63
`BesselJ`, 63
`BesselK`, 63
`BesselY`, 63
`Beta`, 63
`binomial`, 63
`Chi`, 63
`Ci`, 63
`clear all`, 12
`collect`, 27
`conj`, 23
`cross`, 43
`dawson`, 63
`det(A)`, 52
`diff`, 68
`dilog`, 63
`Dirac`, 63
`dot`, 43
`double`, 15
`dsolve`, 88
`Ei`, 63
`eig(A)`, 53
`elfun`, 61
`EllipticCE`, 63
`EllipticCK`, 63
`EllipticCPi`, 63
`EllipticE`, 63
`EllipticF`, 63
`EllipticK`, 63
`EllipticPi`, 63
`erfc`, 63
`erf`, 63
`euler`, 63
`expand`, 27
`factor`, 27
`findsym`, 19
`FresnelC`, 63
`FresnelS`, 63
`GAMMA`, 63
`harmonic`, 63
`Heaviside`, 63
`help symbolic`, 8
`help`, 7
`imag`, 23
`int`, 83
`inv(A)`, 50
`jacobian`, 68
`latex`, 20
`Li`, 63
`limit`, 66
`lnGAMMA`, 63
`mfunlist`, 63
`mfun`, 63
`mhelp`, 8
`pretty`, 20
`Psi`, 63
`real`, 23
`Shi`, 63
`simple`, 29
`simplify`, 29
`solve`, 39
`Ssi`, 63

`subs`, 16
`symsum`, 74
`syms`, 12
`sym`, 12
`taylor`, 72
`vectorize`, 20
`ver`, 7
`W`, 63
`whos`, 12
`Zeta`, 63

**A**
adjunkt, 49
asymptot, 80

**B**
begynnelsevärdesproblem
    analytisk lösning, 88

**C**
Chebyshevpolynom, 59

**D**
datatyp, 12
datoralgebra, 7
defaultvariabel, 19
Diracs deltafunktion, 90
Distribution, 90

**E**
egenvektor, 53
egenvärde, 53
ekvation för plan, 43
elementära matematiska funktioner, 61
enhetsmatris, 50

**F**
fakultet, 76
Fibonacci, 36
findsymregeln, 19
fjädersystem, 97
frihetsgrad, 41
funktionalmatris, 68
funktions M-filer, 61
förenkling av uttryck, 29

**G**
geometrisk summa, 75
Givensmatris, 51
gradient, 68
Gram-Schmidt ortogonalisering, 46
gränsvärde, 66

**H**
Heavisidefunktionen, 90
Hessianen, 71
hjälpfönster, 7
hjälpknappen, 7
hjälpkommandon, 7
Hooks lag, 93

**I**
invers matris, 50
inverterbar matris, 50

**J**
Jacobian, 68

**K**
kopplade fjädrar, 93

**L**
LaTeX, 20
Legendrepolynom, 55
linjärt ekvationssystem, 39

**M**
Maple, 7
matris
    addition, 48
    egenvärde, 53
    enhets, 50
    invers, 50
    multiplikation, 48
    multiplikation med skalär, 48
    produkt, 48
    subtraktion, 48
multipelintegraler, 86

**N**
Newtons avsvalningslag, 97
normalvektor, 43

**O**
ODE
analytisk lösning, 88
system analytisk lösning, 92
omvandla symboliska uttryck, 16
omvandling till decimaltal, 15
ortogonala polynom, 55
ortonormerad bas, 42

**P**
plotta symboliska uttryck, 17
PowerPoint, 20
primfaktor, 27, 29
projektionsformeln, 45

**R**
radioaktiv sönderfallskedja, 99
rekursionsformel, 36

**S**
singulär, 50
singulär matris, 50
skalärprodukt, 42, 43
speciella funktioner, 63
stegfunktion, 90
stötdämpare, 97
substitution, 16
summor, 74
Symbolic Math Toolbox, 7
symbolisk
derivata, 68
gränsvärde, 66
integral, 83
primitiv funktion, 83
symbolisk matematik, 7
symbolisk variabel, 12
symboliska uttryck
plotta, 17
symboliskt objekt, 12
symboliskt uttryck, 12
latex-form, 20
lättläst form, 20
vektorform, 20
system av differentialekvationer, 92

**T**
talföljd, 36
Taylorutveckling, 72
teleskopsumma, 76

**U**
underbestämt system, 39

**V**
Vandermondematris, 53
variabel
numerisk, 13
variabel, default, 19
variabel, symbolisk, 12
vektor, 41
vektorprodukt, 42, 43
vinkelrät projektion, 45

**Ö**
överbestämt system, 39