Innehållsförteckning

Förord

Det föreliggande kompendiet är i första hand avsett som kurslitteratur i kursen Matematiska programpaket som hör till de inledande studierna i matematik vid Åbo Akademi. Kompendiet innehåller en introduktion till de matematiska programpaketen: *Mathematica*, *Matlab*, *Macsyma* och *Mathcad*. Vår förhoppning är att kompendiet skall inspirera till ökat intresse för användning av dessa hjälpmedel.

Grunden till detta kompendium är lagd av Christer Glader som skrev korta manualer till (de tre förstnämnda) programmen i samband med att han föreläste kursen i Matematiska programpaket åren 1991-1994. Peter Biström kompletterade Matlab delen i och med att version 4 av Matlab kom på hösten 1994. Stefan Emet kompletterade i sin tur materialet vid föreläsning av kursen 1996-1997. *Mathcad* tillsattes hösten 1997 som resultat av dess ökade efterfrågan och användbarhet för lärare.

Förslag till förbättringar och andra synpunkter mottages med tacksamhet av författarna, helst skriftligt per E-mail åt: *Stefan.Emet@abo.fi* eller *Christer.Glader@abo.fi* .

Åbo, december 1997

Författarna

1. Mathematica

Mathematica är ett expertsystem för symbolisk och numerisk problemlösning. Huvudarkitekt för Mathematica är Stephen Wolfram. Programmet utvecklades under andra hälften av 80-talet. För närvarande torde Mathematica vara det populäraste matematiska programpaketet tack vare sin användarvänlighet och mångsidighet. Mathematica har goda rutiner för symboliska och numeriska beräkningar, och erbjuder fina möjligheter för grafisk visualisering av data. Detta gör Mathematica till en mångsidig verktygslåda inom de centrala delområdena för matematisk problemlösning.

Ett sätt att använda Mathematica är som en "symbolisk kalkylator" för att kontrollera resultat som räknats för hand eller för att utföra kalkyler som är för arbetsdryga att göra med papper och penna.

Man kan också skriva egna program i Mathematica och läsa in transformeringsregler som "lär Mathematica" att omforma uttryck. Mathematica erbjuder alltså även goda verktyg för funktionell programmering.

Mathematica erbjuder även goda möjligheter till skrivning av dokument innehållande både "vanlig" text och matematiska beräkningar. Denna sk. Notebook-framställning är mycket användbar i och med att beräkningar och modifikationer lätt kan utföras och infogas i dokumenten.

Mathematica består av två delar: en **kernel** (kärna) och en **front end** (bildskärm och tangentbord). Kerneln sköter om de matematiska manipulationer som Mathematica utför. Kerneln är oberoende av datorkonfiguration. Front end sköter om kommunikationen mellan användaren och kerneln, dvs inläsning av data, utskrift av resultat och förmedling av data till och från kerneln. Front enden varierar beronde på vilken sorts dator och operativsystem man kör Mathematica på.

Mathematica finns tillgängligt för PC, UNIX och Macintosh datorer. I denna framställning går vi igenom Mathematica för PC:n och datorer med UNIX operativsystem. Kommandona i Mathematica är dock likadana oberoende av operativsystem.

1.1. Start av Mathematica

På datorer med UNIX operativsystem startas Mathematica med kommandona **math** eller **mathematica**, kommandorads- resp. *Notebook*-körning. Det senare (användarvänligare) sk. *Notebook*-alternativet fungerar på datorer med X-Windows program t.ex. PC:n med programmet X-Win32 eller på SPARC-arbetsstationer.

På en SPARC-arbetsstation startas Mathematica enligt följande recept:

- Logga in på arbetsstationen med användarnamn och lösenord.
- Starta X-Windows med kommandot **openwin** eller något motsvarande.
- Aktivera ett kommandofönster (t.ex. *cmdtool*) genom att klicka på det. Följande kommandon skrives sedan till kommandofönstret.
- Logga därefter in på Aton med kommandot: **rlogin aton**
- Styr utskriften till ifrågavarande skärm med kommandot:
  setenv DISPLAY *name*:**0**
  där *name* är namnet på ifrågavarande arbetsstation.
- Mathematica startas nu med kommandot: **mathematica**

På en PC med programmet X-Win32 sker uppstartningen enligt följande:

- Starta X-Win32 och logga därefter in på Aton via denna.
- Ställ in skärmen med kommandot: **setenv DISPLAY** *name.abo.fi*:**0**

  (*name* är namnet på ifrågavarande PC).
- Starta Mathematica med: **mathematica**

En PC-version av Mathematica startas genom att antingen dubbelklicka på dess ikon eller genom att skriva **Mathematica** på kommandopromten (**run** från **File**-menyn).

Vid start av Mathematica enligt *Notebook*-alternativet öppnas ett tomt fönster med namnet Untitled-1. Detta fönster är nu en *Notebook* och kan sparas som en fil med namnförlängningen *.nb*.

Mathematicas kommandoprompt är av formen **In[n]:=**, där **n** står för en löpande numrering startande från 1. Man skriver in sitt kommando och avslutar raden med **Return** eller **Enter**. Resultatet av kommandot skrivs ut på en rad med utskriftsprompten **Out[n]=** . Om man inte önskar utskrift av resultatet avslutar man inläsningen med ett semikolon. Med ett % tecken kan man på en kommandorad hänvisa till närmast föregående resultatrad, med %% till den nästsista resultatraden osv. Med **%n** kan man hänvisa till resultatraden **Out[n]**.

Exempelvis:

In[1]:= 5 + 4/2

Out[1]= 7

In[2]:= % + 4*%

Out[2]= 35

En Mathematica session avslutas med endera **Quit** eller **Exit**.

1.1.1. Notebook- respektive kommandoradskörning

Vid kommandoradskörning av Mathematica (uppstartning enl. kommandot: *math*) skrives kommandona in till prompten. Piltangenterna fungerar ej vid detta användningssätt som för övrigt funnits i alla versioner av Mathematica.

Utvecklingen av gränsnitten har medfört att sk. Notebook-körning av Mathematica har utvecklats. En Notebook består egentligen av en lista av celler som kännetecknas av hakparenteser i högra kanten av Notebook-fönstret. Cellerna kan vara av olika format, t.ex. titel, avsnitt, text, input, output etc. Formatet på en cell ses förutom av dess font även från alternativet **Style** från **Format**-menyn.

Beräkningar kan utföras på celler som är av formatet input, detta format är också inställt som default i Mathematica. En cell aktiveras genom att klicka med vänstra musknappen på dess kant, flera celler aktiveras genom att hålla in musknappen och dra över flera celler. Formatet på en aktiverad cell kan ändras från alternativet **Format** från huvudmenyn. Exempelvis en cell med formatet input kan ändras till "vanlig" text genom att först aktivera denna och därefter välja **Style** → **Text** från **Format**-menyn.

1.2. Hjälpfunktioner

Om man vill få en hjälptext på något av Mathematicas inbyggda kommandon kan detta göras med **?Xxxx** eller **??Xxxx**, där kommandot med ett frågetecken ger en kortare förklaring och kommandot med två frågetecken en mera detaljerad beskrivning av kommandot **Xxxx**. Om man ger **?Xxxx*** fås en uppräkning av alla kommandon som börjar med textsträngen **Xxxx**, om man ger in **?*Xxxx** fås en uppräkning av alla kommandon som avslutas med texsträngen **Xxxx**. Med **?*Xxxx*** fås en uppräkning av alla kommandon som innehåller textsträngen **Xxxx**. Om man då till exempel vill få en uppräkning av alla kommandon i Mathematica kan man ge kommandot **?***.

?Limit

Limit[expr, x -> x0] finds the limit as x -> x0

?*Limit

$IterationLimit Limit $RecursionLimit SequenceLimit

Vid Notebook-körning av Mathematica i X-Windows omgivning kan hjälptext läsas (med hjälp av en *Help Browser*) från alternativet **Help** från huvudmenyn.

1.3. Editering av kommandon

Editering av kommandon under en interaktiv session görs enklast med **emacs**-editorn †. Ett urval av editeringskommandon som i Mathematica automatiskt öppnar emacs editorn är:

Edit[], inläsning av nytt kommando;

EditIn[], editering av senaste kommandot;

EditIn[n], Editering av kommandot **In[n]**;

Edit[%], editering av senaste resultat;

Edit[Out[n]], editerar resultat **Out[n]**;

EditDefinition[f], editerar definitionen av f.

När man givit något av ovanstående kommandon kommer man in i emacs editorn och kan där utföra sina ändringar. Man kan bland annat röra sig med pilarna. När editeringen är utförd håller man **Control**- tangenten nedtryckt och trycker därefter turvis på **x** och **c** tangenten, släpper därefter upp Control tangenten och trycker på **y** tangenten.

1.3.1. Evaluering av kommandon

Kommandon evalueras vid Notebook-körning endera genom att trycka **Shift Enter** eller genom att välja **Evaluation→ Evaluate Cells** från **Kernel**-menyn. Celler med formatet input kan evalueras genom att först aktivera ifrågavarande cell/celler och därefter trycka **Shift Enter**.

Vid "kommandorads"-körning evalueras kommandona genom att trycka på **Enter**.

† gäller endast vid kommandorads-körning i UNIX-omgivning

1.3.2. Symbol- och kommandopaletter

I Mathematica version 3.0 kan grekiska bokstäver och matematiska symboler och funktioner väljas genom klickning från paletter. Dessa hittas från **Palettes** under **File**-menyn, bl.a. följande paletter finns tillgängliga under **Palettes** alternativ: **BasicTypesetting**:

Palett för indexering:

Grekiska alfabetet:

Under **Palettes** alternativ: **BasicInput** finns bl.a. följande paletter:

Relationssymboler och dylikt:

Funktioner:

Symbol eller funktion väljes från en palett genom klickning, varefter dess tomma (svarta) fält ifylls. Alla funktioner och symboler kan även användas genom att skriva motsvarande kommandon. Det är också möjligt att skapa egna paletter dit man kan sätta funktioner/symboler, mera information om detta hittas från avsnitt 1.10.2 i help-browsern el. Mathematica-manualen [5].

1.3.3. Logfil

Om man, vid kommandoradskörning på Aton, vill spara sin Mathematica session på en fil kan detta enklast göras på följande vis. Innan man går in i Mathematica programmet öppnar man med kommandot % **script** en logfil med namnet **typescript**. När man avslutat sin Mathematica session ger man kommandot % **exit** varvid logfilen stängs. Logfilen kan sedan printas ut på en laserskrivare exempelvis med kommandot % **txt2ps typescript | lpr -Pnamn**, där **namn** är namnet på laserskrivaren. Om man har utfört grafiska kommandon under sin Mathematica session bör man notera att typescript filen innehåller "skräptecken" som man kan stryka i en lämplig editor, (t.ex emacs), innan man tar ut filen på laserskrivare.

1.3.4. Batchkörning

Om man inte vill ge in en följd av kommandon interaktivt, eller om man har tidskrävande beräkningar, kan man bilda en inputfil med kommandon. Resultaten av kommandona skrivs ut på en specificerad outputfil. Ta som exempel Integrering av $f(x) = log(x^2)$. Följande inputfil har namnet **infil**.

```
AppendTo[$Echo, "stdout"]
Integrate[Log[x^ 2], x]
```

Efter sista raden bör man trycka på return tangenten för att Mathematica skall exekvera den. Nu kan vi utföra batchkörningen med kommandot

% math -batchinput -batchoutput <infil> utfil,

och resultatet skrivs ut i filen **utfil**, som då ser ut på följande vis:

```
{"stdout"}
Integrate[Log[x^ 2], x]

-2*x + x*Log[x^ 2]
```

Egna filer kan laddas in med << -tecknen, t.ex. en fil *infil* läses in enligt kommandot:

In[1]:= << infil;

Detta är mycket användbart vid evaluering av många och långa uttryck.

1.4. Aritmetik i Mathematica

För aritmetik har vi följande operatorer:

x^ y, exponentiering;

-x, minus;

x/y, division;

x y z, multiplikation;

x*y*z, multiplikation;

x+y+z, addition.

De vanliga precedensrelationerna gäller. De kan dock upphävas med parenteser, (). Vi tar några exempel.

In[3]:= 2 3*4

Out[3]= 24

In[4] := (3+4)^ 2-2(3+1)

Out[4]= 41

Om de i beräkningarna ingående talen är exakt givna är även svaret exakt. Om decimaltal ingår i beräkningarna kan svaret vara avrundat. Mathematicas funktion **N[expr, n]** kan användas för att erhålla ett närmevärde av **expr** med **n** decimalers noggrannhet.

In[5]:= 1.5^ 45

Out[5]= $8.39666\ 10^{7}$

In[6] := 2^ 100

Out[6]= 1267650600228229401496703205376

In[7]:= N[2^ 100]

Out[7]= $1.26765\ 10^{30}$

Några inbyggda konstanter i Mathematica:

Pi = 3.14159....

E = 2.71828....

I = imaginära enheten,

Infinity = oändligheten.

In[7]:= N[Pi, 40]

Out[7]= 3.1415926535897932384626433832795028841 97

Relationsopertorerna i Mathematica är följande:

x == y, likhet,

x != y, olikhet,

x > y, x större än y,

x < y, x mindre än y,

x >= y, x större än eller lika med y,

x <= y, x mindre än eller lika med y.

Man kan också räkna exakt med komplex aritmetik i Mathematica. Här följer ett urval av Mathematicas funktioner för manipulering av komplexa tal:

x + I y, det komplexa talet x + iy;

Re[z], realdelen av z;

Im[z], imaginära delen av z;

Conjugate[z], det komplexa konjugatet av z;

Abs[z], beloppet av z;

Arg[z], argumentet av z.

Vi exemplifierar den komplexa aritmetiken:

```
In[11]:= z = (4+3I)/(2-I)
Out[11]= 1 + 2 I
In[12]:= Re[z]
Out[12]= 1
In[13]:= Im[z]
Out[13]= 2
In[14]:= Conjugate[z]
Out[14]= 1 - 2 I
In[15]:= Abs[z]
Out[15]= Sqrt[5]
```

1.5. Definiering av egna variabler

Man kan definiera egna variabler i Mathematica, dessa behåller det definierade värdet tills de omdefinieras eller stryks. Det är god politik att definiera de egna variablerna i små bokstäver för att inte riskera sammanblandning med Mathematicas egna konstanter eller funktioner, som alltid inleds med stor bokstav.

x = value, tilldelar variabeln x ett värde,

x = y = value, tilldelar variablerna x och y samma värde,

x =., stryker variabeln x.

1.6. Funktioner

1.6.1. Några inbyggda funktioner

Mathematica har alla standardfunktioner inbyggda. Dessutom finns en massa specialfunktioner tillgängliga. Man bör notera att alla Mathematicas egna funktioner, och även alla kommandon, inleds med stor bokstav efterföljt av små bokstäver. Om funktionsnamnet eller kommandot är ett sammansatt ord så inleds varje delord med stor bokstav. Låt oss se på några av de färdigt definierade funktionerna.

Sqrt[x], kvadratroten av x,

Exp[x], exponentialfunktionen,

Log[x], naturliga logaritmen,

Log[b, x], logaritmen med basen b,

Sin[x], **Cos[x]**, **Tan[x]**, några trigonometriska funktioner,

ArcSin[x], **ArcCos[x]**, **ArcTan[x]**, samt deras inversa funktioner,

n!, fakulteten av n,

Abs[x], absoluta beloppet av x,

Round[x], avrundning till närmaste heltal,

Floor[x], ger det största heltalet mindre än eller lika med x

Ceiling[x], ger det minsta heltalet som är större än eller lika med x,

Mod[n, m], n modulo m (resten av n/m),

Random[], likformigt fördelat slumptal i intervallet (0,1),

Max[x, y, ...], **Min[x, y, ...]**, maximum resp. minimum av talen x, y, ... ,

FactorInteger[n], primtalsfaktorering av talet n.

In[1]:= x = 6.8

Out[1]= 6.8

In[2]:= y = -3.3

Out[2]= -3.3

In[3]:= 4!

Out[3]= 24

In[4]:= Abs[y]

Out[4]= 3.3

In[5]:= Round[x]

Out[5]= 7

In[6]:= Max[1,2,4,x]

Out[6]= 6.8

In[7]:= FactorInteger[99]

Out[7]= {{3, 2}, {11, 1}}

1.6.2. Definiering av egna funktioner

Man kan behändigt definiera egna funktioner i mathematica, varvid man med fördel beaktar konventionen att de egna funktionerna definieras med små bokstäver. Syntaxen ser ut som följer:

f[x_]:= x^ 2, definierar f(x) som en parabel,

g2[x_, y_]:= x - Sin[y], definierar g2(x,y) som en funktiona av två variabler,

?f, visar definitionen av funktionen f,

Clear[f], stryker definitionen av funktionen f.

Observera att funktioners argument alltid innesluts med hakparenteser, och att understreckningen efter argumentet, (x_), endast används då man definierar funktionen, inte när man i fortsättningen använder den. Vidare använder man tilldelningstecknet, := , när man definierar en funktion, och likhetstecknet, = , då man definierar en variabel.

In[20]:= g[a_]:= a^ 2

In[21]:= g[5]

Out[21]= 25

In[22]:= f[x_ ,y_]:= 5 x^ 2 - 3 y

In[23]:= f[3, 4]

Out[23]= 33

In[24]:= f[g[x], y]

Out[24]= $5x^4 - 3y$

1.6.3. Rekursivt definierade funktioner

Man kan definiera funktioner rekursivt i Mathematica. Låt oss ta beräkningen av $n!$ som exempel. Vi vet att $1! = 1$ och att $n! = n(n-1)!$, och därmed kan vi rekursivt beräkna $n!$. Detta kan i Mathematica göras på följande vis:

```
In[39]:= fac[1]:= 1
In[40]:= fac[n_]:= n fac[n-1 ]
In[41]:= ?fac
fac
fac/ : fac[1]:= 1
fac/ : fac[n_]:= n fac[n - 1]
In[42]:= fac[5]
Out[23]= 120
```

1.6.4. Styckevis definierade funktioner

Om man vill definiera en funktion styckevis i Mathematica, så görs det enklast på följande vis. Säg att man vill definiera $f(x)$ så att $f(x) = -1$ då $x < 0$, $f(x) = 0$ då $x = 0$ och $f(x) = 1$ då $x > 0$.

```
In[1]:= f[x_]:= -1/ ; x < 0
In[2]:= f[x_]:= 0/ ; x == 0
In[3]:= f[x_]:= 1/ ; x > 0
In[4]:= f[-4.9]
Out[4]= -1
In[5]:= f[12.1]
Out[5]= 1
In[6]:= f[0]
Out[6]= 0
```

1.7. Förenkling av uttryck

För förenkling av algebraiska uttryck finns ett flertal kommandon tillgängliga. Här presenteras några av de mest användbara:

Expand[expr], multiplicerar ut produkter och potenser i täljaren och utvecklar ett bråk som en summa av delbråk,

ExpandAll[expr], som föregående förutom att nämnaren också utvecklas,

Factor[expr], reducerar till en produkt av faktorer,

Together[expr], gör liknämnigt,

Apart[expr], utför partialbråksuppdelning,

Cancel[expr], förkortar bort gemensamma faktorer i täljare och nämnare,

Simplify[expr], strävar till en möjligast enkel framställning,

Numerator[expr], plockar ut täljaren,

Denominator[expr], plockar ut nämnaren.

Några exempel:

In[29]:= e = (x - 1)^ 2 (2 + x)/((1 + x) (x - 3) ^ 2)

Out[29]= $\frac{(-1+x)^2(2+x)}{(-3+x)^2(1+x)}$

In[30]:= Expand[e]

Out[30]= $\frac{2}{(-3+x)^2(1+x)} - \frac{3x}{(-3+x)^2(1+x)} + \frac{x^3}{(-3+x)^2(1+x)}$

In[31]:= ExpandAll[e]

Out[31]= $\frac{2}{9+3x-5x^2+x^3} - \frac{3x}{9+3x-5x^2+x^3} + \frac{x^3}{9+3x-5x^2+x^3}$

In[32]:= Factor[%]

Out[32]= $\frac{(-1+x)^2(2+x)}{(-3+x)^2(1+x)}$

In[33]:= Apart[%]

Out[33]= $1 + \frac{5}{(-3+x)^2} + \frac{19}{4(-3+x)} + \frac{1}{4(1+x)}$

In[34]:= Together[%]

Out[34]= $\frac{2-3x+x^3}{(-3+x)^2(1+x)}$

In[35]:= Simplify[%33]

Out[35]= $\frac{(-1+x)^2(2+x)}{(-3+x)^2(1+x)}$

In[36]:= Numerator[%]

Out[36]= $(-1+x)^2(2+x)$

In[37]:= Denominator[%%]

Out[37]= $(-3+x)^2(1+x)$

Om vi vill transformera uttryck innehållande trigonometriska funktioner kan vi använda några specialkommandon:

In[1]:= Expand[(Sin[x]– Cos[x])^ 2]

Out[1]= $Cos[x]^2 - 2Cos[x]Sin[x] + Sin[x]^2$

In[2]:= Expand[% , Trig – > True]

Out[2]= $1 - Sin[2x]$

In[3]:= Factor[% , Trig – > True]

Out[3]= $(Cos[x] - Sin[x])^2$

In[4]:= Factor[Sin[3 (x + y)], Trig – > True]

Out[4]= (1 + 2 Cos[2 x + 2 y]) Sin [x + y]

Som vi ser kan kommandona **Factor** och **Expand** operera på trigonometriska uttryck om vi i kommandot innefattar optionen **Trig – > True** . Nu kan vi plocka in ytterligare kommandon ur ett av Mathematicas s.k. paket. Vi läser in paketet **Algebra'Trigonometry'** och får access till kommandot **TrigReduce**.

In[5]:= << Algebra'Trigonometry'

In[6]:= TrigReduce[Sin[3 (x + y)]]

Out[6]= $3(Cos[y]Sin[x] + Cos[x]Sin[y] - \frac{4(Cos[y]Sin[x]+Cos[x]Sin[y])^3}{3})$

1.8. Listor

En **lista** i Mathematica är en ordnad mängd vars element är nya listor, eller enkla (atomära) element. En lista innesluts i klamrar, {}, och elementen åtskiljs med kommatecken.

In[1]:= lista1 $= \{1, 7, 5\}$

Out[1]= {1, 7, 5}

In[2]:= lista2 $= \{\{5, 9\}, Sin[1], \{a, 2\}\}$

Out[2]= {{5, 9}, $Sin[1]$, {a, 2}}

Man kan plocka ut element ur en lista med **Part** kommandot eller med dubbla hakparenteser, [[ ]].

In[3]:= lista1[[2]]

Out[3]= 7

In[4]:= Part[lista2, 3]

Out[4]= $\{a, 2\}$

Man kan också ändra värdet på element i en lista:

In[5]:= lista2[[1]]= 8

Out[5]= 8

In[6]:= lista2

Out[6]= $\{8, Sin[1], \{a, 2\}\}$

Och man kan operera matematiskt med listor. Då utförs operationerna elementvis:

In[7]:= lista3 = 5 lista1

Out[7]= $\{5, 35, 25\}$

In[8]:= lista1*lista1

Out[8]= $\{1, 49, 25\}$

Med kommandona **Sort** och **Reverse** kan man sortera listor och med kommandot **Length** kan man bestämma antalet element i en lista.

In[9]:= Sort[lista1]

Out[9]= $\{1, 5, 7\}$

In[10]:= Reverse[Sort[lista1]]

Out[10]= $\{7, 5, 1\}$

Vi kan bilda unionen och snittet av listor med kommandona **Union** och **Intersection**, samt bilda komplementet med avseende på en grundmängd med kommandot **Complement**. Med kommandot **Table** kan vi bilda en lista enligt en regel:

In[11]:= grundmengd = Table[i, $\{i, 1, 7\}$]

Out[11]= $\{1, 2, 3, 4, 5, 6, 7\}$

In[12]:= Complement[grundmengd, lista1]

Out[12]= $\{2, 3, 4, 6\}$

In[13]:= Union[lista1, lista3]

Out[13]= $\{1, 5, 7, 25, 35\}$

In[14]:= Intersection[lista1, lista3]

Out[14]= $\{5\}$

1.8.1. Parenteser, klamrar och dylikt

Som vi har sett använder man i Mathematica parenteser för att gruppera uttryck och termer vid uträkningar. Hakparenteser används för att instänga argumenten för en funktion eller ett kommando. Klamrar används för att bilda listor och dubbla hakparenteser för att plocka ut element ur listor.

Om man jobbar på en terminal som inte har dessa tecken så kan man använda skandinaviska tecken istället, nämligen: **Ä** = **[**, **Å** = **]**, **ä** = { samt **å** = }.

1.8.2. Vektorer och matriser

Mathematica kan också tolka en lista som en vektor. Ta som exempel vektorerna $z_1 = 2\bar{x}+3\bar{y}$ och $z_2 = -\bar{x} + 2\bar{y}$. I Mathematica kan vi representera dem som listor:

```
In[1]:= z1 = {2, 3}
Out[1]= {2, 3}
In[2]:= z2 = {-1, 2}
Out[2]= {-1, 2}
```

Vid Notebook-körning kan vektorer och matriser också skapas via paletten: **Creating Lists and Matrices** (finns under **File→Palettes→BasicCalulations→Lists and Matrices→**). Härvid kan antalet rader ökas genom att trycka **Ctrl Enter**, kolonnantalet ökas genom att trycka **Ctrl** +.

Med + och – tecknen kan vi addera respektive subtrahera vektorer.

```
In[3]:= z1 + z2
Out[3]= {1, 5}
In[4]:= z1 - z2
Out[4]= {3, 1}
```

Skalära produkten bildas med punkttecknet: . , t.ex.:

```
In[5]:= z1 . z2
Out[5]= 4
```

Om man har rymdvektorer med 3 komponenter så kan man göra motsvarande operationer med listor bestående av tre element. Om man har vektorer med n st. komponenter, så består motsvarande listor av n element.

En matris kan i Mathematica representeras som en lista med listor, där första raden i matrisen är en lista som är första element i listan som betecknar matrisen. Den i:te raden i matrisen är en lista som är element nummer i i den lista som betecknar matrisen. Vi tar som exempel en $2x2$ matris med första raden bestående av elementen 1 och 3, och andra raden bestående av elementen -4 och 9.

```
In[6]:= a = {{1, 3}, {-4, 9}}
Out[6]= {{1, 3}, {-4, 9}}
```

Med kommandot **MatrixForm** får vi ett mera "matrislikt" utseende på utskriften:

```
In[6]:= MatrixForm[a]
Out[6]= / / MatrixForm = 1  3
                        -4  9
```

Vi kan använda **Table** kommandot för att bilda matriser enligt en viss regel.

```
In[1]:= f[i_,j_]:= i + 2j
In[2]:= Table[f[i, j], {i,1,2}, {j,1,2}]
Out[2]= {{3, 5}, {4, 6}}
```

Matrismultiplikation utförs med punkttecknet. Som exempel definierar vi två matriser och multiplicerar dem:

```
In[1]:= m1 = {{1, 2}, {3, 4}} ; m2 = {{3, 5}, {-7, -2}};
In[2]:= m1 . m2
Out[2]= {{-11, 1}, {-19, 7}}
```

Transponatet av en matris bildas med kommandot **Transpose**. Med kommandot **Det** kan man beräkna determinanten av en matris, och med **Eigenvalues** kommandot kan man beräkna egenvärden för en matris. Kommandot **Inverse** beräknar inversen till en matris.

In[3]:= Transpose[m1]

Out[3]= {{1, 3}, {2, 4}}

In[4]:= Det[m1]

Out[4]= −2

In[5]:= Eigenvalues[m1]

Out[5]= $\{\frac{5+\sqrt{33}}{2}, \frac{5-\sqrt{33}}{2}\}$

In[6]:= Inverse[m1]

Out[3]= $\{\{-2, 1\}, \{\frac{3}{2}, -\frac{1}{2}\}\}$

1.9. Gränsvärden

För gränsvärdesbestämning utnyttjas kommandot **Limit** enligt följande syntax.

Limit[expr, x − > x0], gränsvärdet av expr då x − > x0,

Limit[expr, x − > x0, Direction − > −1], gränsvärdet av expr då x − > x0 från höger, dvs högergränsvärdet,

Limit[expr, x − > x0, Direction − > 1], gränsvärdet av expr då x − > x0 från vänster, dvs vänstergränsvärdet.

Några exempel:

In[1]:= Limit[Sin[x]/x , x − > 0]

Out[1]= 1

In[2]:= Limit[(1 + 1/x)^ x , x − > Infinity]

Out[2]= E

In[3]:= Limit[1/x , x − > 0]

Out[3]= Infinity

Det sista resultatet var överraskande, vi väntade oss att gränsvärdet skulle vara odefinierat, men fick istället högergränsvärdet! Det kan vara skäl att i Mathematica alltid kolla också höger- och vänstergränsvärdet:

In[4]:= Limit[1/x , x − > 0, Direction − > −1]

Out[4]= Infinity

In[5]:= Limit[1/x , x -> 0, Direction -> 1]

Out[5]= -Infinity

Om vi försöker bestämma gränsvärdet av en funktion som har varken högergränsvärde, vänstergränsvärde eller egentligt gränsvärde, så kan Mathematica ibland ange inom vilket intervall funktionen antar värden. Exempel:

In[6]:= Limit[Sin[x], x -> Infinity]

Out[6]= RealInterval[{-1, 1}]

1.10. Summor och produkter

Med kommandona **Sum** och **Product** kan vi beräkna summor och produkter.

Sum[f[i], {*i*, *imin*, *imax*}**]**, summan av f(i) med avseende på i från imin till imax,

Sum[f[i, j], {*i*, *imin*, *imax*}, {*j*, *jmin*, *jmax*}**]**, dubbelsumma, först över j sen över i,

Product[f[i], {*i*, *imin*, *imax*}**]**, Produkten av f(i) med avseende på i då i går från imin till imax.

Vår vana trogen tar vi några exempel:

In[1]:= Sum[x^ i/i , {*i*, 1, 4}**]**

Out[1]= $x + \frac{x^2}{2} + \frac{x^3}{3} + \frac{x^4}{4}$

In[2]:= Sum[i^ 2 + 2^ i , {*i*, 1, 100}**]**

Out[2]= 25353012004564588029934067491000

In[3]:= Sum[1/i^ 4, {*i*, 1, *Infinity*}**]**

Out[3]= $\frac{Pi^4}{90}$

In[4]:= N[%, 10]

Out[4]= 1.082323234

Avslutningsvis två exempel med produkt-kommandot:

In[7]:= Product[x + i, {*i*, 1, 4}**]**

Out[7]= (1 + x) (2 + x) (3 + x) (4 + x)

In[8]:= Product[E^i, {*i*, 1, *n*}**]**

Out[8]= $E^{(n(1+n))/2}$

Från paletten (under **File→Palettes→BasicCalculations→Calculus→Common Operations**) hittas symbolerna för summering och produkt:

Dessa används genom att fylla i de tomma fälten.

1.11. Derivering

Kommandot **D** används vid derivering. Här följer en syntaxbeskrivning:

D[f[x], x], partiella derivatan av f med avseende på x,

D[f[x1,...,xn], x1, x2, ... , xn], partiell derivering n gånger med avseende på x1, x2 , ... , xn,

D[f[x], $\{x,\ n\}$**]**, partiella derivatan av ordning n med avseende på x.

Några exempel:

In[1]:= D[x^ n, x]

Out[1]= nx^{-1+n}

In[2]:= D[x^ 2 y^ 3 + x y^ 2 + x y z , x , y]

Out[2]= $2y + 6xy^2 + z$

In[3]:= D[x^ 2 Sin[x], $\{x,\ 4\}$**]**

Out[3]= $-8xCos[x] - 12Sin[x] + x^2 Sin[x]$

Dessa funktioner fås även (vid Notebook-körning) i symbolform från motsvarande palett, under **File→Palettes→BasicCalculations→Calculus→Common Operations**.

1.12. Integrering

Kommandot **Integrate** används för att beräkna bestämda och obestämda integraler. Med kommandot **NIntegrate** kan man numeriskt approximera värdet av integraler som inte går att bestämma exakt.

Integrate[f[x], x], obestämda integralen av f(x) med avseende på x,

Integrate[f[x], {*x*, *xmin*, *xmax*}], integralen då x går från xmin till xmax,

NIntegrate[f[x], {*x*, *xmin*, *xmax*}], numerisk approximation av integral.

Så tar vi några exempel:

In[1]:= Integrate[x^ 2 Exp[x], x]

Out[1]= $2E^x - 2E^x x + E^x x^2$

In[2]:= Integrate[x^ 2 , {*x*, *a*, *b*}]

Out[2]= $\frac{-a^3}{3} + \frac{b^3}{3}$

In[3]:= Integrate[Sin[Cos[x]], {*x*, 0, 1}]

Out[3]= Integrate[Sin[Cos[x]], {*x*, 0, 1}]

In[4]:= NIntegrate[Sin[Cos[x]], {*x*, 0, 1}]

Out[4]= 0.738643

Integral-symboler finns också i samma palett som deriveringssymbolerna.

1.13. Nollställen till polynom

Med kommandona **Roots** och **NRoots** kan man hitta exakta respektive approximativa nollställen till ett polynom. **Reduce** ger även alla möjliga lösningar med beaktande av eventuella parametrar.

Roots[ekvation , x], ger exakta nollställen till polynom av gradtal mindre än eller lika med 4,

NRoots[ekvation , x], ger numeriska approximationer av nollställena till ett polynom.

Reduce[ekvation , x], ger alla tänkbara lösningar m.avs. på x resp. parameter till en ekvation.

Vi löser några polynomekvationer:

In[1]:= p[x_]:= x^ 4 - x^ 3 + 2 x^ 2 - 4 x - 8

In[2]:= Roots[p[x]== 0 , x]

Out[2]= x == -1 || x == 2 || x == 2 I || x == -2 I

In[3]:= p = x^ 6 - 7x^ 5 + x^ 4 - 4x^ 3 + x^ 2 - x + 5

Out[3]= $5 - x + x^2 - 4x^3 + x^4 - 7x^5 + x^6$

In[4]:= NRoots[p == 0, x]

Out[4]= x == -0.635494 - 0.638162 I || x == -0.635494 + 0.638162 I ||
x == 0.237584 - 0.988592 I || x == 0.237584 + 0.988592 I ||
x == 0.859725 || x == 6.93609

In[5]:= Reduce[ax + b == 0, x]

Out[5]= b==0 && a==0 || a≠0 && x==$-\frac{b}{a}$

1.14. Ekvationssystem

Med kommandona **Solve** och **NSolve** kan man lösa linjära ekvationssystem och också endel icke-linjära ekvationssystem exakt resp. approximativt.

Solve[ekvationer, {$x1,\ x2,\ \ldots\ ,\ xn$}**]**, löser ett ekvationssystem med avseende på x1,x2, ... , xn.

Ekvationssystem ges lämpligen in som listor:

In[1]:= e1 = {$3x + 3y - z == 5, x - 4y + 2z == 1, -3x + y + z == 2$}**;**

In[2]:= Solve[e1 , {$x,\ y,\ z$}**]**

Out[2]= {{$x- > \frac{15}{14}, y- > \frac{7}{4}, z- > \frac{97}{28}$}}

In[3]:= NSolve[e1 , {$x,\ y,\ z$}**]**

Out[3]= {{$x- > 1.07143, y- > 1.75, z- > 3.46429$}}

In[4]:= Solve[{ x^ 2 + y^ 2 == 5 , x + y == 1 } , { x, y}]

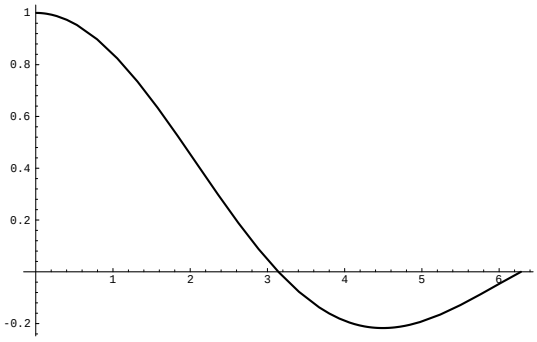
Out[4]= {{$x- > 2,\ y- > -1$}, {$x- > -1,\ y- > 2$}}

1.15. Grafik

1.15.1. Grafer av funktioner av en variabel

Plot är standardkommandot för grafisk framställning av en funktion $f(x)$ av en variabel. Exempelvis grafen av $f(x) = \frac{sin(x)}{x}$ i intervallet $[0, 2\pi]$.

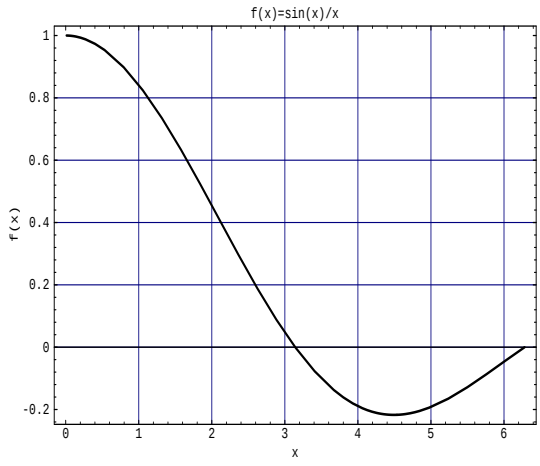
```
In[1]:= f[x_]:= Sin[x]/ x
In[2]:= Plot[f[x], {x, 0.01 , 2 Pi}]
```



Figur 1.1. Plottning av endimensionell funktion.

Det finns ett stort antal olika optioner att modifiera grafiken med. **PlotLabel** – > **"Rubrik"** är en option som ger en rubrik åt bilden. Med **FrameLabel** – > {"$x - axeln$", "$y - axeln$"} ger man namn åt x-axeln och y-axeln. Kommandot **Frame** – > **True** rutar in grafen och **GridLines** – > **Automatic** utför en rutindelning. Om man inte överhuvudtaget önskar något koordinatsystem i bilden ger man optionen **Axes** – >**False**.

```
In[3]:= Plot[f[x], {x, 0.01 , 2 Pi}, Frame – >True, PlotLabel – >
"f(x)=sin(x)/x", FrameLabel – > { "x","f(x)"}, GridLines – >Automatic ]
```



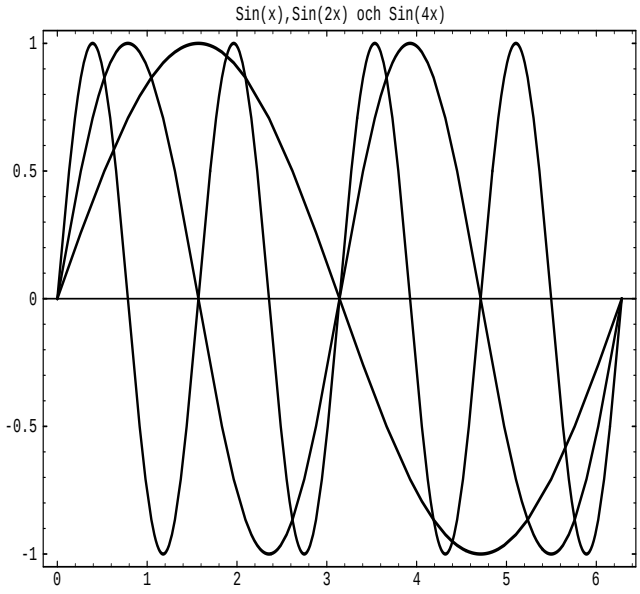
Figur 1.2.

Om grafen inte verkar tillräckligt jämn kan man öka på antalet funktionsvärden som används vi utritningen. Detta görs med kommandot **PlotPoints – > nr**, där **nr** är en siffra som anger minimiantal funktionsberäkningar som görs vid utritningen av grafen. Det normala värdet på **nr** är 25, genom att sätta detta till ett större värde får man en jämnare graf.

Om man ritar ut en funktion som varierar mycket i y-led så kan Mathematica automatiskt "klippa av" en del av grafen. Detta kan undvikas med optionen **PlotRange – > All**. Man kan också ange att endast de funktionsvärden som ligger i intervallet [ymin, ymax]ritas ut genom att ge optionen **PlotRange – > { ymin, ymax }**.

Om man vill rita ut flera kurvor i samma koordinatsystem så kan det göras enligt syntaxen **Plot[{ f1[x], f2[x], ... , fn[x]} , { x, a, b}]**. Exempel

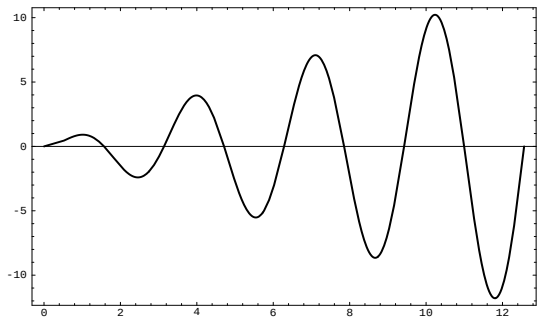
```
In[4]:= Plot[{ Sin[x], Sin[2x], Sin[4x]} , {x, 0 , 2 Pi}, Frame – >True,
PlotLabel – > "Sin(x), Sin(2x) och Sin(4x)"]
```



Figur 1.3.

Med kommandot **Show** möjliggörs en ny utritning av en tidigare definierad graf. Man kan då testa med nya optioner. Exempel:

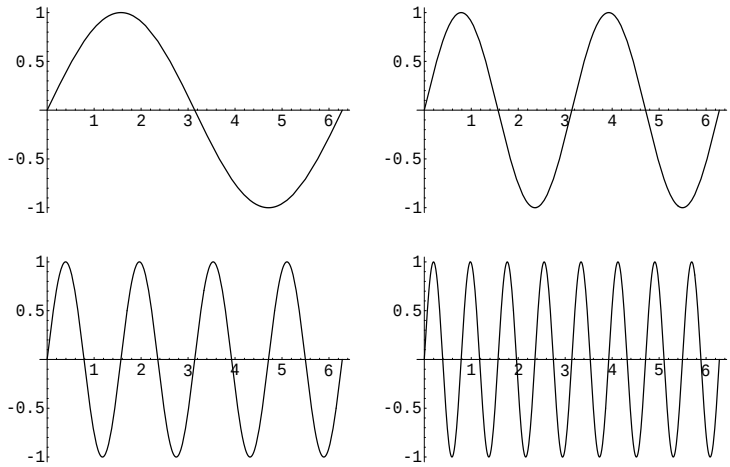
```
In[5]:= g = Plot[x Sin[2x], { x, 0, 4 Pi }];
In[6]:= Show[g, Frame– >True]
```



Figur 1.4.

Med Kommandot **GraphicsArray** kombinerat med kommandot **Show** kan vi gruppera och rita ut grafer i ett önskat antal koordinatsystem. Som exempel ritar vi ut 4 koordinatsystem.

```
In[7]:= g1 = Plot[Sin[x], {x, 0 , 2 Pi}];g2 = Plot[Sin[2x], {x, 0 , 2 Pi}];
g3 = Plot[Sin[4x], {x, 0 , 2 Pi}];g4 = Plot[Sin [8x], {x, 0 , 2 Pi}];
In[8]:= Show[GraphicsArray[{{ g1 , g2}, { g3, g4}}]]
```

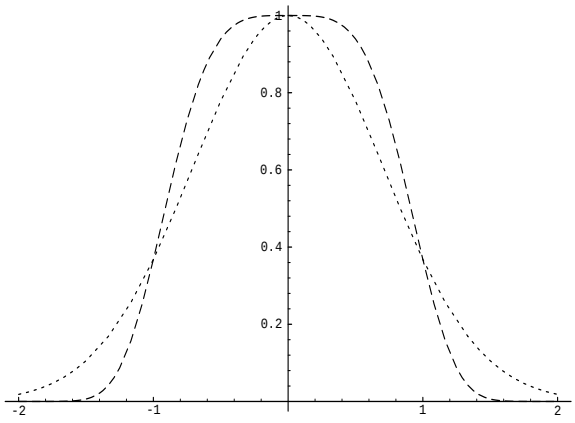


Figur 1.5.

Om man ritar ut flera grafer i samma koordinatsystem kan det vara bra att använda olika linjetyper. Detta kan åstadkommas med kommandot **Dashing**. Exempelvis **Dashing[{ 0.01 }]** anger att linjetypen består av streck och mellanrum med längderna 0.01. Där längden mäts som en bråkdel av bredden av bilden. Med **Dashing[{ 0.010, 0.01, 0.015, 0.01 }]** avses linjetyp av omväxlande längden 0.010 och 0.015, med mellanrum på 0.01. **Dashing[{}]** ger en heldragen linjetyp. Om man vill ändra på tjockleken på linjerna kan detta anges med

kommandot **Thickness**. Exempelvis **Thickness[0.01]** anger att tjockleken är 0.01 delar av bredden på bilden. Ändringen av linjetyp och tjocklek görs som optioner med kommandot **PlotStyle**. Exempel:

```
In[9]:= Plot[{ Exp[-x^ 2], Exp[-x^ 4]} , { x, -2, 2 }, PlotStyle-> {{
Dashing [{ 0.004, 0.01 }], Thickness[0.002]},
{ Dashing[{ 0.014, 0.01 }],Thickness[0.002]}}]
```

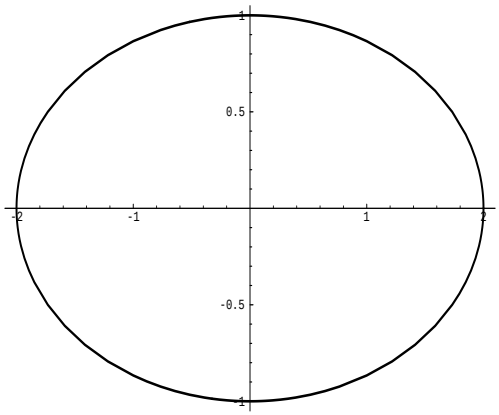


Figur 1.6.

1.15.2. Parameterframställda plana kurvor

En kurva vars (x,y) koordinater ges av funktionerna $x(t)$ och $y(t)$ då $t \in [a, b]$ kan framställas med kommandot **ParametricPlot**. Tag exempelvis en ellips:

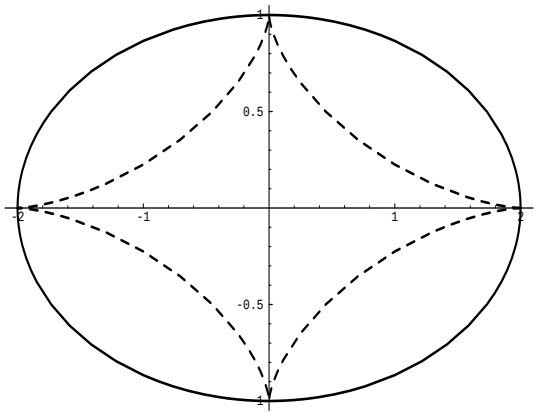
```
In[1]:= x[t_]:= 2 Cos[t];y[t_]:= Sin[t];
In[2]:= ParametricPlot[{ x[t], y[t]} , { t, 0, 2 Pi }]
```



Figur 1.7.

Man kan också rita ut flera parameterframställda kurvor i samma koordinatsystem. Exempelvis:

```
In[3]:= x1[t_]:= 2 Cos[t];y1[t_]:= Sin[t];
In[4]:= x2[t_]:= 2 Cos[t]^ 3 ;y2[t_]:= Sin[t]^ 3 ;
In[2]:= ParametricPlot[{{ x1[t], y1[t]},{ x2[t ], y2[t]}} , { t, 0, 2 Pi } ,
PlotStyle -> { Dashing[{}], Dashing[{ 0.015, 0.015 }]}]
```

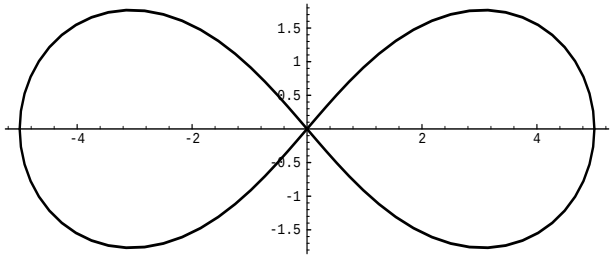


Figur 1.8.

1.15.3. Implicit givna kurvor

Grafer av implicit givna kurvor kan framställas grafiskt om man först läser in paketet **Graphics'ImplicitPlot'** genom att ge kommandot: **<<Graphics'ImplicitPlot'**. Vi tar en lemniskata som exempel:

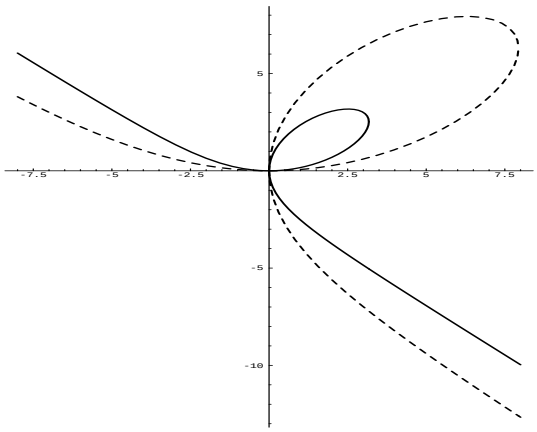
```
In[1]:= ekv = (x^ 2 + y^ 2)^ 2 == 25 (x^ 2 - y^ 2);
In[2]:= ImplicitPlot[ekv, { x, -5, 5 }]
```



Figur 1.9. Grafen av en lemniskata.

Även detta kommando kan användas för att rita ut flera implicit givna kurvor i samma koordinatsystem.

```
In[3]:= ekv1 = x^ 3 + y^ 3 == 6 x y ;
In[4]:= ekv2 = x^ 3 + y^ 3 == 15 x y ;
In[5]:= ImplicitPlot[{ ekv1, ekv2 }, { x, -8, 8 },
PlotStyle -> { Dashing[{} ],Dashing[{ 0.015, 0.015 }]} ]
```

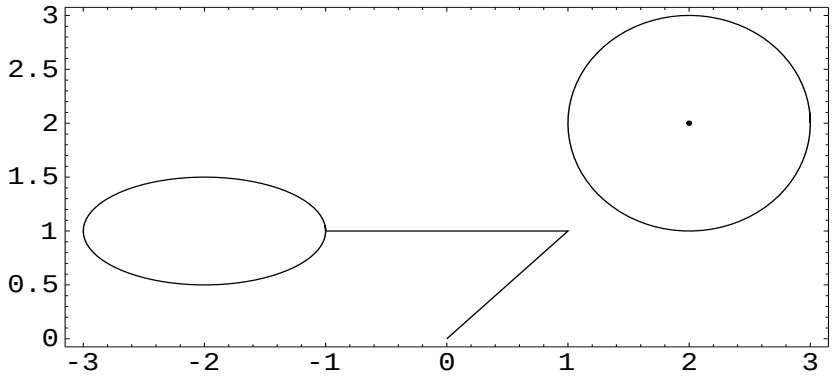


Figur 1.10.

1.15.4. Plana geometriska figurer

Vissa standardgrafer som punkter, linjer, cirklar och ellipser kan man framställa enklare:

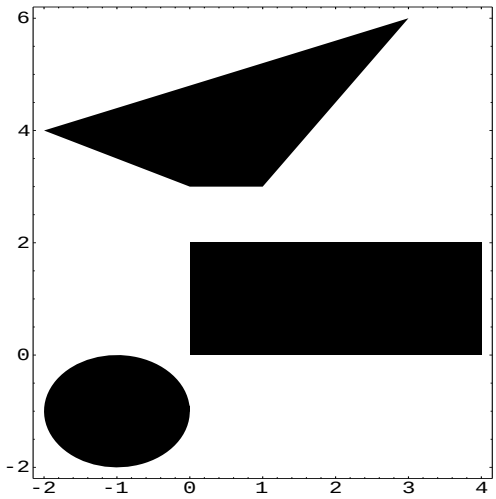
```
In[1]:= punkt = Point[{ 2, 2 }];linje = Line[{{ 0, 0 },{ 1, 1 },{ -1 , 1 }}];
In[2]:= cirkel = Circle[{ 2, 2 } , 1];ellips = Circle[{ -2, 1 },{ 1, 0.5 }];
In[3]:= Show[Graphics[{ punkt, linje, cirkel, ellips }], Frame -> True
AspectRatio ->Automatic]
```



Figur 1.11.

Man kan också på liknande sätt framställa fyllda cirklar, rektanglar och månghörningar:

```
In[4]:= rektangel = Rectangle[{ 0, 0 },{ 4, 2 }];skiva = Disk[{ -1, -1 }, 1];
In[5]:= polygon = Polygon[{{ 0, 3 } , { 1, 3 },{ 3, 6 },{ -2 , 4 }}];
In[6]:= Show[Graphics[{ rektangel, skiva, polygon }], Frame -> True
AspectRatio -> Automatic ]
```

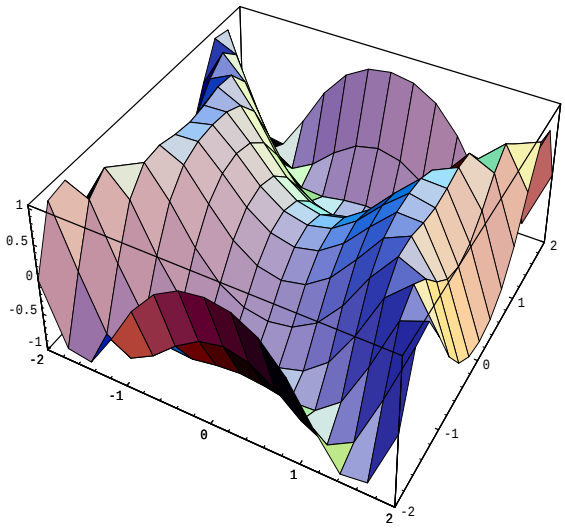


Figur 1.12.

1.15.5. Ytor och nivåkurvor av funktioner av två variabler

Om vi vill rita ut ytan $f(x, y)$ över en kvadrat i xy–planet kan detta göras med kommandot **Plot3D**. Exempel:

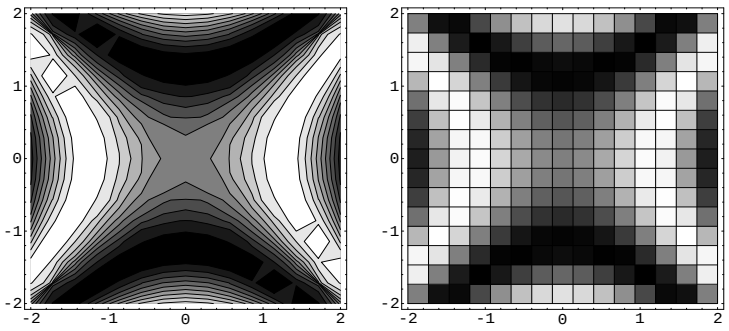
```
In[1]:= f[x_ , y_]:= Sin[x^ 2 - y^ 2];
In[2]:= Plot3D[f[x , y], {x, -2, 2} , {y, -2, 2}]
```



Figur 1.13.

Nivåkurvor kan ritas ut med kommandot **ContourPlot** och en "täthetskurva" i gråtoner med kommandot **DensityPlot**:

```
In[3]:= g1 = ContourPlot[f[x , y], {x, −2, 2} , {y, −2, 2}];
g2 = DensityPlot[f[x , y], {x, −2, 2} , {y, −2, 2}];
In[4]:= Show[GraphicsArray[{ g1 , g2 }]]
```

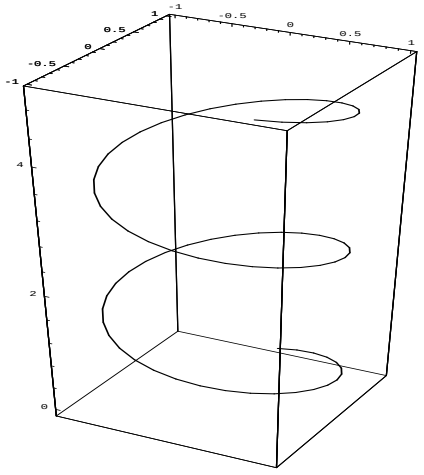


Figur 1.14.

1.15.6. Parameterframställda ytor och rymdkurvor

En rymdkurva kan beskrivas med tre funktioner $x(t), y(t)$ och $z(t)$, som beskriver koordinaternas förändringar då $t \in [a, b]$. Vi använder plotkommandot **ParametricPlot3D**:

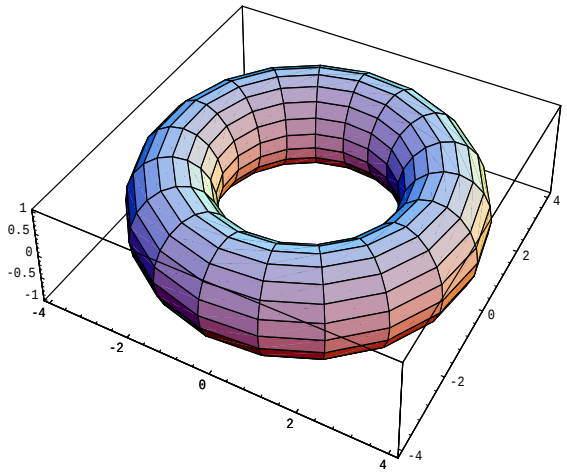
```
In[1]:= x[t_]:= Sin[t]; y[t_]:= Cos[t]; z[t_]:= t/ 3 ;
In[2]:= ParametricPlot3D[{ x[t], y[t], z[t]} , {t, 0, 15}]
```



Figur 1.15.

För en parameterframställd yta ges koordinaterna $x(u,v), y(u,v)$ och $z(u,v)$ som funktioner av två variabler u, v. Vi använder samma plottningskommando som för rymdkurvor.

```
In[3]:= x[u_ , v_]:= Cos[v](3 + Cos[u]) ; y[u_ , v_]:= Sin[v](3 + Cos[u]) ;
z[u_ , v_]:= Sin[u];

In[4]:= ParametricPlot3D[{ x[u, v], y[u, v], z[u, v]} ,{v, 0, 2 Pi},

{u, 0, 2 Pi } ]
```

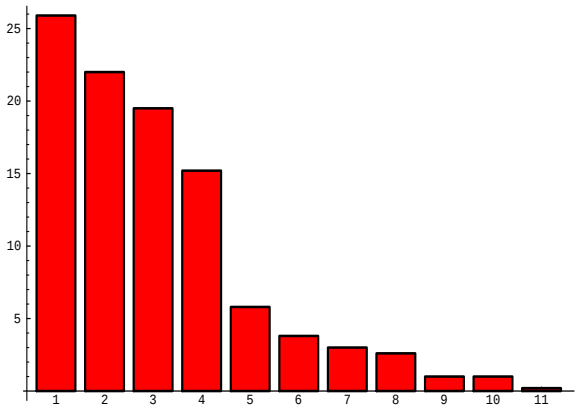


Figur 1.16.

1.15.7. Stapel- och cirkeldiagram

För att kunna använda stapel- och cirkeldiagram läser vi in paketet <<**Graphics‘Graphics‘**. Vi gör ett stapeldiagram över resultatet i första valomgången i presidentvalet 1994 med kommandot **BarChart**.

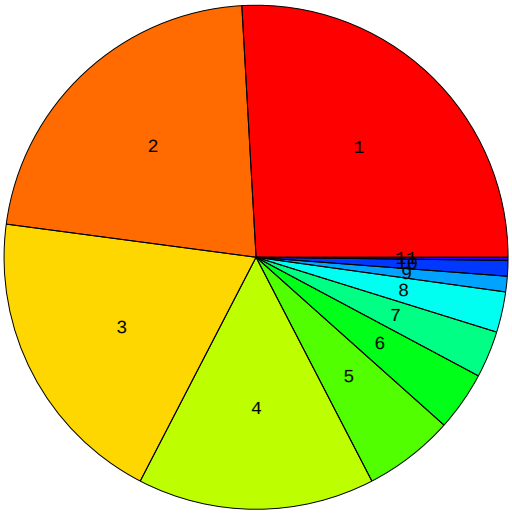
```
In[1]:= resultat = { 25.9, 22.0, 19.5, 15.2, 5.8, 3.8, 3.0, 2.6, 1.0, 1.0, 0.2 } ;

In[2]:= BarChart[resultat, PlotRange-> All ]
```



Figur 1.17. Presidentvalet 1994.

Med kommandot **PieChart** fås ett cirkeldiagram över resultatet.

In[3]:= PieChart[resultat]



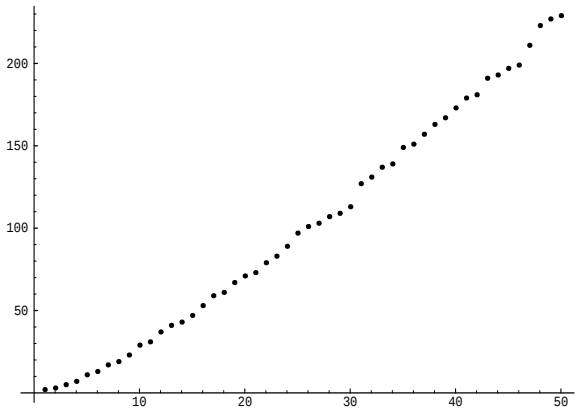
Figur 1.18.

1.15.8. Visualisering av data i listor

Om vi vill pricka ut diskret data sparat i en lista, så använder vi kommandot **ListPlot**. Listans index fås på x-axeln och motsvarande listelement i y-led. Som exempel bildar vi en lista med de 50 första primtalen och prickar ut dem i en graf.

In[1]:= lista = Table[Prime[i], { i, 1, 50 }];

In[2]:= ListPlot[lista]



Figur 1.19. De 50 första primtalen.

1.15.9. Utskrift av grafer

Ett sätt att få den i Mathematica skapade grafiken på papper är som följer. Innan man går in i Mathematica programmet bestämmer man på vilken laserskrivare man skall ha ut grafiken genom att ge kommandot % **setenv PRINTER namn**, där namn står för namnet på en laserskrivare, (t.ex. asa). Med kommandot **Display** kan grafiken sändas till printer el. fil. Ifall man vill skicka en "EPS"-graf (*Encapsulated PostScript*) till den valda printern, görs detta enligt:

In[1]:= Plot[Sin[x], { x, 0, 2 Pi }]

In[2]:= Display["!lpr",% , "EPS"]

Alternativt kan grafen skrivas ut till en fil *sin.ps* enligt:

In[3]:= Display["sin.ps",Plot[Sin[x], { x, 0, 2 Pi }], "EPS"]

notera att **Plot**-kommandon kan inkluderas i **Display**-kommandot. Filen *sin.ps* kan därefter skrivas ut (på Aton) med kommandot: **lpr -Pmate sin.ps**
(-Pmate styr utskriften till lasern mate).

Grafer kan bl.a. skrivas ut enligt följande format:

EPS - Encapsulated PostScript
GIF - GIF-format
Illustrator - Adobe Illustrator format
PDF - Adobe Acrobat dokument
TIFF - TIFF-format
XBitmap - X-Windows bitmap

Ifall man vill spara en graf i "gif"-form ges optionen "GIF", enligt:

In[4]:= Display["filnamn.gif",Plot[Sin[x], { x, 0, 2 Pi }], "GIF"]

Vid Notebook-körning av Mathematica kan grafer även skrivas ut genom att först aktivera (dubbelklicka) ifrågavarande graf och därefter välja **Print** från **File**-menyn.

Referenser

[1] Andersson, G. : Tillämpad matematik med Mathematica, 1994.

[2] Cameron, S. : The Mathematica graphics guidebook, 1995.

[] Kent, P., Ramsden, P., Wood, J.: Eperiments in Undergraduate Mathematics ;A Mathematica-Based Approach, 1996.

[3] Ruskeepää, H. : Mathematica-opas, Versio 2.2, 1993.

[4] Wolfram, S. : Mathematica. A System for Doing Mathematics by Computer, 1988.

[5] Wolfram, S. : The Mathematica book. Third ed. Mathematica version 3, 1996.

Övningsuppgifter

1. Bestäm konstanten **a** så att ytan som ligger mellan $f(x) = 2x(a - x)$ och x-axeln är **a** ytenheter.Använd t.e.x. **Integrate** och **Solve** kommandona.

2. Beräkna Fibonacci tal rekursivt

$$x_n = x_{n-1} + x_{n-2}$$
$$x_1 = x_2 = 1$$

3. Beräkna

$$\int x^3 e^{-a^2 x^2} dx$$
$$\int_0^\infty \frac{1}{\sqrt{x}(x+1)} dx$$
$$\int_0^\pi cos(cos(x)) dx.$$

4. Lös ekvationssystemet med avseende på x och y

$$3px - p^2 y = 5$$
$$-p^2 x + 5y = 10$$

5. Upprita $f(x) = sin(\frac{1}{x})$ i intervallen: a) $x \in [-10, 10]$, b) $x \in [-1, 1]$ och c) $x \in [-0.01, 0.01]$.

6. Trochoiden är den kurva som beskrivs av en punkt P på avståndet b från medelpunkten av en cirkel med radien a, när cirkeln rullar framåt på x-axeln. Kurvans ekvation kan ges i parameterform:

$$x(t) = at - bsin(t),$$
$$y(t) = a - bcos(t),$$

   Upprita trochoiden då $t \in [0, 6\pi]$, med a) a = b = 2 , b) a = 2, b = 2.5 , c) a = 2, b = 1 och d) Fallena a), b) och c) i samma koordinatsystem med olika linjetyp.

7. Implicit givna kurvor av typen $x^n + y^n = 1$, där n är ett positivt heltal, kallas stundom superellipser. Upprita dessa för $n = 2, 4, 6, 8$ och $x \in [-1, 1]$. Vad händer då n är jämnt och växer?

8. Upprita ytan och nivåkurvorna för funktionen $f(x, y) = sin(xy)$, då $x \in [0, 3]$ och $y \in [0, 3]$.

9. Upprita och ta ut på papper den rymdyta som ges av:

$$x(t, u) = sin(2t),$$
$$y(t, u) = sin(2t)sin(u),$$
$$z(t, u) = sin(2t)cos(u),$$
$$t \in [-\frac{\pi}{2}, \frac{\pi}{2}], u \in [0, 2\pi].$$

10. Pröva **Fit[data, fkt, var]** kommandot, bilda t.e.x. en lista **fp** bestående av de 50 första primtalen **fp=Table[Prime[x],**$\{x, 50\}$ **]**. Kommandot **gp=ListPlot[fp]** plottar punkterna.

f1=Fit[fp,$\{1, x\}$ **,x]** ger nu den bästa lineära kombinationen till primtalen av funktionerna **1** och **x**,plottas med: **f1p=Plot[f1,**$\{x, 0, 50\}$ **]**, **Show[f1p,gp]** plottar båda i samma kordinatsystem.

Pröva även med **f2=Fit[fp,**$\{1, x, x^2\}$ **,x]** kvadrattermen torde ge en bättre approximation.

11. Skriv in vektorerna $z_1 = 2\bar{x} + 5\bar{y}, z_2 = -3\bar{x} + \bar{y}$ och $z_3 = 4\bar{x} - 3\bar{y}$ som tre stycken listor. Beräkna a) $z_1 + z_2 + z_3$ b) $z_1 - 3z_2 + 8z_3$ c) $z_1 \cdot z_2$

12. Skriv in matriserna

$$A = \begin{pmatrix} 3 & 8 \\ -7 & 4 \end{pmatrix},$$

$$B = \begin{pmatrix} -1 & 6 \\ 2 & 5 \end{pmatrix}.$$

Beräkna a) $A + B$ b) $A * B$ c) $B * A$ d) Determinanten av A och B e) Inversen av A och B f) Egenvärdena till A och B.

13. Beräkna följande gränsvärden i Mathematica:

$$a) \lim_{x \to \infty} \frac{5x^3 - 7x}{2x^3 + 8x^2}$$

$$b) \lim_{x \to 0} sin(\frac{1}{x})$$

$$c) \lim_{x \to 4} \frac{\sqrt{x} - 2}{x^2 - 8x + 16}$$

14. Beräkna summan av följande ändliga respektive oändliga geometriska serier:

$$a) \sum_{j=0}^{n} aq^j$$

$$b) \sum_{j=0}^{\infty} aq^j$$

$$c) \sum_{j=0}^{\infty} 2(\frac{1}{3})^j$$

15. a) Beräkna $f'''(x)$, då $f(x) = ln(x^3)sin(x^2)$

b) Beräkna partiella derivatorna med avseende på x och y av funktionen:

$$f(x, y) = 3x^3 ycos(xy^4)$$

16. Sök nollställena till:

$$a)\ f(x) = x^4 + 2x^3 + 29x^2 + 128x - 2240$$

$$b)\ f(x) = 3x^5 - 7x^2 + 8x - 21$$

2. Matlab

Matlab är ett programpaket som ursprungligen kodades i FORTRAN av Cleve Moler och senare i C av Steve Bangert. Avsikten var att åstadkomma en användarvänlig programmeringsomgivning där man inte behövde använda traditionella programmeringsspråk utan kunde hålla sig till en problemformulering som var nära den matematiska. Man ville även visualisera data i form av grafer på ett snabbt och behändigt sätt.

Rutinerna i Matlab bygger på FORTRAN rutinerna i LINPACK och EISPACK biblioteken, som betraktas som de bästa för numerisk linjär algebra. Matlab erbjuder goda möjligheter för numerisk analys, matriskalkyl, signalanalys och grafik. För mera specifika ändamål finns toolboxar utvecklade.

På Aton finns sedan hösten 1997 installerad versionen Matlab 5.1.

I det följande beskrivs hur Matlab fungerar i UNIX miljö. Kommandona är dock i "allmänhet" likadana oberoende av version och operativsystem.

2.1. Start av Matlab

Matlab startas med kommandot **matlab** och avslutas med **exit** eller **quit**.

```
% matlab

                        < M A T L A B (R) >
              (c) Copyright 1984-97 The MathWorks, Inc.
                        All Rights Reserved
                           Version 5.1.0.421
                             May 25 1997

  To get started, type one of these commands:  helpwin, helpdesk, or demo.
  For information on all of the MathWorks products, type tour.

>>
```

Vill man ha information om kommandon så skriver man **help**, varefter följande litania dyker upp.

```
>> help

HELP topics:

matlab/general       -  General purpose commands.
matlab/ops           -  Operators and special characters.
matlab/lang          -  Programming language constructs.
matlab/elmat         -  Elementary matrices and matrix manipulation.
matlab/elfun         -  Elementary math functions.
matlab/specfun       -  Specialized math functions.
matlab/matfun        -  Matrix functions - numerical linear algebra.
matlab/datafun       -  Data analysis and Fourier transforms.
matlab/polyfun       -  Interpolation and polynomials.
matlab/funfun        -  Function functions and ODE solvers.
matlab/sparfun       -  Sparse matrices.
matlab/graph2d       -  Two dimensional graphs.
matlab/graph3d       -  Three dimensional graphs.
matlab/specgraph     -  Specialized graphs.
matlab/graphics      -  Handle Graphics.
matlab/uitools       -  Graphical user interface tools.
matlab/strfun        -  Character strings.
matlab/iofun         -  File input/output.
matlab/timefun       -  Time and dates.
matlab/datatypes     -  Data types and structures.
matlab/demos         -  Examples and demonstrations.
simulink/simulink    -  Simulink
simulink/blocks      -  Simulink block library.
simulink/simdemos    -  Simulink demonstrations and samples.
simulink/dee         -  Differential Equation Editor
toolbox/ident        -  System Identification Toolbox.
toolbox/local        -  Preferences.
toolbox/optim        -  Optimization Toolbox.
toolbox/signal       -  Signal Processing Toolbox.
stateflow/stateflow  -  Stateflow
stateflow/sfdemos    -  (No table of contents file)
toolbox/tour         -  MATLAB Tour

For more help on directory/topic, type "help topic".
```

Information om något specifikt ämne, t.ex. demos fås med kommandot » help demos .

2.1.1. Input

Matlabs prompt består av tecknet >>. Vid inmatning av kommandon fås en utskrift av resultatet ifall man ej avslutar med ; .

```
>> 1+1 % utskrift fås av resultatet

ans =

     2

>> 1+1;    % ; tecknet utför operationen men ger ingen utskrift
```

Kommentarer på kommandorader eller i funktionsfiler bör föregås av % -tecknet. Variabel- och funktionsnamn börjar med bokstav och efterföljs av bokstäver eller siffror. Namn kan bestå av högst 19 signifikanta tecken.

Matlab är "case-sensitive" d.v.s. man skiljer på stora och små bokstäver i variabelnamn. Inbyggda funktionsnamn skrivs med små bokstäver.

Kommandot **who** listar använda variabler, medan kommandot **whos** ger en mera detaljerad beskrivning av variablerna. **what** kommandot används för att lista .m och .mat filerna i den aktuella directoryn

Tal kan ges in i formerna; 3, -99, 0.0001, 3.1000, 9.631E-10, 5.3e20. Man bör observera att inget blanktecken föregår E,e. Default display är 4 decimaler, **format long e** kommandot, ger en display med 16 decimaler. För att komma tillbaka till default display ges kommanadot **format**. Matlab räknar med 16 decimalers noggrannhet (dubbelprecision). Representerbara positiva flyttal ligger i intervallet (1e-308,1e308). Några färdigt definierade "konstanter" är

```
inf      % positiva oändligheten
eps      % 2^(-52)
pi       % 3.141....
i        % imaginära enheten
```

Man kan skriva komplexa tal i rektangulär eller polär form, z = 3+4∗i eller w = r∗exp(i∗θ). Vi kan även bilda komplexa matriser, t.ex. **a=[1 2;3 4] + i∗[5 6;7 8]** eller **a=[1+5∗i 2+6∗i;3+7∗i 4+8∗i]**. Observera att blanktecken saknas mellan operatorerna och imaginär- respektive realdelen.

Om ett uttryck är så långt att det inte ryms på en rad avslutas raden med ... varefter man trycker på return och fortsätter uttrycket på följande rad.

Matlab är användarvänligt, med pilarna på tangentbordet kan man röra sig åt alla fyra håll, alternativt till vänster med Ctrl-L, till höger med Ctrl-R, uppåt med Ctrl-P och nedåt med Ctrl-N. Det är således lätt att ändra på ett tidigare kommando. Dessutom söker Matlab upp det senaste kommandot av något slag, t.ex. **plot(x,y)** om man skriver **plo** och trycker på pil uppåt.

2.1.2. Output

Om man vill spara delar av sin session på en logfil kan detta göras med kommandot **diary**. Kommandot **diary fff.txt** öppnar en logfil **fff.txt** i vilken allt som skrivs ut på skärmen sparas. Kommandot **diary off** stänger logfilen.

Kommandot **save** sparar variablerna i arbetsarean i den externa filen **matlab.mat**. Filen kan läsas in med **load** kommandot. Kommandot **save temp x** sparar endast variabeln **x** i **temp.mat**, medan **save temp x y z** sparar variablerna **x, y, z**. Med kommandot **load temp** läser man in variabeldata från filen **temp.mat**.

Vidare kan man med **load fil.dat** läsa in innehållet i filen **fil.dat** på underområdet till Matlab. Antag t.ex. att filen xx.dat innehåller följande;

```
3 34 17
```

Då kunde ett anrop se ut så här:

```
>> load xx.dat
>> xx

   xx =

       3 34 17
```

Vid inläsning av .dat -filer bör dock varje rad (i filen) ha lika många värden.

I en extern fil med filförlängningen **.m** på subdirectoryn Matlab kan man läsa in variabler eller en följd av uttryck i Matlab-syntax. I Matlab kan denna fil läsas in genom att skriva ut filnamnet (utan förlängning). Mera om dessa **.m** filer i kapitel 2.5.

Vidare erbjuder Matlab goda möjligheter till visualisering av data. Plottning och utskrift beskrivs senare (kapitel 2.4.).

2.2. Matriser och vektorer

Matlab är en interaktiv programmeringsomgivning som erbjuder kraftfulla kommandon för matrismanipulationer. Grundobjekt är matriser, skalärer tolkas som 1×1-matriser. Matriser kan läsas in explicit, element för element, genereras med hjälp av uttryck och funktioner eller läsas in från externa filer. Inga typdeklarationer på variablerna och ingen matrisdimensionering krävs.

2.2.1. Inläsning och accessering av matriser

En 3×3 matris kan läsas in enl:

```
>> P = [1 2 3;4,5,6;7 8 9]      eller      >> P = [1 2 3
                                                  4 5 6
   P =                                            7 8 9]
        1 2 3
        4 5 6                                P =
        7 8 9                                     1 2 3
                                                  4 5 6
                                                  7 8 9
```

En vektor läses in analogt:

```
>> V = [1 2 3 4 5 6 7 8 9]      eller      >> V =[1,2,3,4,5,6,7,8,9]


   V =                                        V =
        1 2 3 4 5 6 7 8 9                         1 2 3 4 5 6 7 8 9
```

denna definieras dock enklast enligt

```
>> V = 1:9          eller      >> V = 1:1:9


   V =
         1 2 3 4 5 6 7 8 9
```

Vid tilldelningen ovan gäller att V går från 1 till 9 med steglängden 1. En ekvidistant tilldelning med steglängden $\frac{1}{2}$ utförs enligt

```
>> V = 1:1/2:4          eller    >> V = 1:.5:4

   V =
        1.0000    1.5000    2.0000    2.5000    3.0000    3.5000    4.0000
```

Matriselementen behöver inte nödvändigtvis vara reella eller komplexa tal, vilka som helst syntax-riktiga Matlab uttryck duger bra. Vektorkomponenter refereras till med ett index, **V(i)** där i≥1. Matriselement refereras till med rad- och kolonnindex, **P(i,j)** där i≥1 och j≥1. Noteras bör att indexeringen av matriser och vektorer alltid börjar från (1,1) resp. (1).

För vektorn ovan lyckas ej följande:

```
>> V(0)              eller      >> V(10)

???    Index exceeds matrix dimensions.
```

För att plocka ut delmatriser finns följande kommandon:

```
P(i:j,k:l)       delmatris av P som ges av raderna i till j och
                                  kolonnerna k till l.
P(i:j,:)                 delmatris av P som ges av raderna i till j och
                                  alla kolonner.
P(:,k:l)         delmatris av P som ges av kolonnerna k till l och
                                  alla rader.
P(:,:)                     hela P.
P(:)                         vektorn där  "kolonnerna i P är satta efter varandra".
```

```
>> P(2:3,:)

   ans =
          4 5 6
          7 8 9

>> P(2:3,2:3)

   ans =
         5 6
         8 9

>> x = [-1 2 3];        % ; utför tilldelning men inte utskrift
```

```
>> x(6) = -(x(1))

   x =
      -1 2 3 0 0 1      % x(4),x(5) fylldes i med nollor

>> P = [1 2 3;4 5 6];

>> P = [P;10 11 12]        % sätter till en rad
   P =
       1  2  3
       4  5  6
       7  8  9
      10 11 12
>> P = [P,[1;1;1]]      % sätter till en kolonn
   P =
       1 2 3 1
       4 5 6 1
       7 8 9 1
```

2.2.2. Funktioner för generering av matriser

Matlab har sex inbyggda funktioner som genererar specifika matriser (vektorer):

eye - diagonalelementen är ettor övriga är nollor.

ones - varje element är ett

zeros - varje element är noll

rand - likformigt fördelade slumptal på intervallet [0,1]

randn - normalfördelade slumptal (medelvärde 0, variansen 1)

randperm(n) - en vektor bestående av slumpmässigt permuterade heltal mellan 1 och n

Funktionerna används genom att ange dimension på matris †, t.ex. ones(2,4) ger en 2×4 matris bestående av ettor. Ifall ingen dimension anges returneras en skalär.

```
>> eye(1)          ekvivalent med       >> eye
   ans =
          1

>> eye(2)          >> eye(2,3)
   ans =              ans =
          1 0                1 0 0
          0 1                0 1 0
```

† randperm-funktionen returnerar endast vektorer

```
>> P=ones(3)          ekvivalent med       >> P=ones(3,3)

   P =
        1 1 1
        1 1 1
        1 1 1


>> P=zeros(3)              eller      >> P=zeros(1,3)

   P =                                   P =
        0 0 0                                 0 0 0
        0 0 0
        0 0 0

>> s=rand                  eller      >> s=rand(2)

   s =                                   s =
        0.2191                                0.0470    0.6793
                                              0.6789    0.9347

>> s=randn                 eller      >> s=randn(2,1)

   s =                                    s =
        -0.0751                               1.1650
                                              0.6268
```

Avslutningsvis ett exempel på användning av permutations-kommandot **randperm** som är användbart vid simulering av t.ex. en kömodell.

```
>> queue=randperm(20)

queue =

Columns 1 through 10

5    18    12    19    13    16     7     4     1     8

Columns 11 through 20

3    11     9    20     6     2    17    15    10    14
```

2.3. Operatorer och funktioner i Matlab

De vanliga algebraiska operationerna finns tillgängliga:

```
+    % addition
-    % subtraktion
*    % matrismultiplikation
/    % höger division, dvs X=B/A är lösning till X*A=B
     % där A är kvadratisk
\    % vänsterdivision, dvs X=A\B är lösning till A*X=B
     % där A är kvadratisk
^    % potensering
```

En operator som föregås med punkt (.) betecknar elementvis operation. Operatorerna + och - är redan elementvisa så ingen punkt behövs. Med .* fås elementvis multiplikation.

```
>> a=[1 2 3;1 2 2];
>> a.*a
  ans =
       1 4 9
       1 4 4
```

Elementvisa kvoten ges av ./ och .\ . Ingående matriser bör vara av samma dimension.

```
>> x=[1 3 4];y=[5 8 9];     >> x./2
>> x./y                        ans =
  ans =                            0.5000 1.5000 2.0000
        0.2000 0.3750 0.4444

>>x.\y                      >> x.\2
  ans =                       ans =
        5.0000 2.6667 2.2500         2.0000 0.6667 0.5000
```

Nedan demonstreras elementvis exponentiering.

```
>> x.^y                    >> y.^2
  ans =                     ans =
        1 6561 262144             25 64 81

>> 2.^x                    >> 2.^[x y]
  ans =                     ans =
        2 8 16                    2 8 16 32 256 512
```

Några av de i Matlab ingående funktionerna för elementvis operation på matriser är

```
abs      % absolutbelopp            round   % avrundning till närmaste heltal
real     % realdel                  fix     % avrundar mot noll
imag     % imaginärdel              ceil    % avrundar mot inf
conj     % komplexa konjugatet      floor   % avrundar mot -inf
sqrt     % kvadratrot               sign    % teckenfunktionen
angle    % argumentet               rem     % rest vid division
log      % naturliga logaritmen     log10   % log.  med basen 10
exp      % exponentialfunktionen
```

Några av de trigonometriska funktionerna är

```
sin         asinh         sinh          asinh
cot         acoth         coth          acoth
cos         acosh         cosh          acosh
tan         atanh         tanh          atanh
```

Därtill finns en större uppsättning speciella funktioner typ gamma, bessel etc samtliga beskrivna genom kommandot **help specfunc**. Gammafunktionen kan användas till fakultetsberäkning, ty Matlab har ingen annan funktion för detta ändamål.

```
>> help gamma

GAMMA  The gamma function.
       Y = GAMMA(X) evaluates the gamma function at all the elements of X.
       X must be real.
       gamma(x) = integral from 0 to inf of t^(x-1) exp(-t) dt.
       gamma(n+1) = n!  = n factorial = prod(1:n).

       See also GAMMALN, GAMMAINC.

>> gamma(5)          % gamma(5) = 4!  = 4*3*2*1 = 24

ans =
      24
```

2.3.1. Relationsoperatorer

Tillgängliga relationsoperatorer är

```
< , <= , > , >= , == , ~=
```

betecknande $<$, $\leq$, $>$, $\geq$, $=$ respektive $\neq$.

Logiska operatorer är

& , | , xor

betecknande AND, OR respektive XOR.

Matriser kan jämföras elementvis, resultatet av en jämförelse av två matriser, med lika dimensioner, är en boolesk matris bestående av ettor och nollor betecknande sant respektive falskt.

```
>> 5+7~=12          >> 3+2~=4
   ans =               ans =
        0                   1

>> a=[2 1; 3 4]; b=[2 0; 2 3];
>> a==b
ans =
      1 0
      0 0
```

Om a och b är likadimensionerade matriser så är **a&b** en matris med nollor där antingen a(i,j)$=0$ eller b(i,j)$=0$, ettor i övrigt.

```
>> a&b
ans =
     1 0
     1 1
```

a | **b** är en matris med ettor där a(i,j)$\neq 0$ eller b(i,j)$\neq 0$, nollor i övrigt.

```
>> a|b
ans =
     1 1
     1 1
```

xor(**a**, **b**) är en matris med ettor där a(i,j)$\neq 0$ eller b(i,j)$\neq 0$, dock ej båda, nollor i övrigt.

```
>> xor(a,b)
ans =
     0 1
     0 0
```

2.3.2. Matrisfunktioner och manipulering av matriser

Några av de mera avancerade matrisfunktioenerna följer nedan

```
'         % transponat
size      % storleken av en matris
cond      % konditionstalet
norm      % matrisnorm, finns flere olika typer
rank      % rangen, dvs antalet linjärt oberoende rader eller kolonner
det       % determinanten
trace     % diagonalelementens summa
lu        % LU-faktorisering
inv       % matrisinvers
qr        % Orthogonal-triangulär uppdelning
eig       % egenvektorer och egenvärden
poly      % ger det karakteristiska polynomets koefficienter.
svd       % singulärvärdesuppdelning
```

Om z är komplex så är z' el. **conj(z)** det konjugerade transponatet.

```
>> x = [-1 0 2+i]'
   x =
      -1
       0
       2-i

>> x = conj([-1 0 2+i])
   x =
      -1
       0
       2+i

>> a = [1 2; 3 5];

>> trace(a)
   ans =
           6

>> det(a)
   ans =
          -1

>> size(a)
   ans =
           2  2
```

Nedan följer en demonstration hur en matris kan uppdelas i Matlab.

```
>> A = [1 2 3; 4 5 6; 7 8 0];

>> [L,U] = lu(A)              % LU-faktorisering

L =                           % om raderna permuteras så är L nedåt triangulär

    0.1429    1.0000         0
    0.5714    0.5000    1.0000
    1.0000         0         0


U =                           % U är uppåt triangulär

    7.0000    8.0000         0
         0    0.8571    3.0000
         0         0    4.5000

>> L*U                        % kontroll

ans =

     1     2     3
     4     5     6
     7     8     0

>> [Q,R] = qr(A)              % ortogonal-triangulär uppdelning

Q =                           % här är Q ortogonal

   -0.1231    0.9045    0.4082
   -0.4924    0.3015   -0.8165
   -0.8616   -0.3015    0.4082


R =                           % R är uppåt triangulär

   -8.1240   -9.6011   -3.3235
         0    0.9045    4.5227
         0         0   -3.6742
```

```
>> Q*R                          % kontroll

ans =

    1.0000    2.0000    3.0000
    4.0000    5.0000    6.0000
    7.0000    8.0000    0.0000


>> [U,S,V] = svd(A)             % singulärvärdesuppdelning

U =

    0.2304    0.3961   -0.8889
    0.6073    0.6552    0.4493
    0.7604   -0.6433   -0.0896


S =

   13.2015         0         0
         0    5.4388         0
         0         0    0.3760


V =

    0.6046   -0.2733    0.7482
    0.7257   -0.1982   -0.6589
    0.3284    0.9413    0.0784

>> [X,D] = eig(A)               % egenvektorer resp.  egenvärden

X =                             % kolonnerna är egenvektorer

    0.7471   -0.2998   -0.2763
   -0.6582   -0.7075   -0.3884
    0.0931   -0.6400    0.8791
```

```
D =                          % på diagonalen ligger egenvärdena

   -0.3884         0         0
         0   12.1229         0
         0         0   -5.7345

>> inv(X)                    % inversberäkning

ans =

    0.9187   -0.4648    0.0834
   -0.5725   -0.7202   -0.4981
   -0.5140   -0.4751    0.7661

>> X*D*inv(X)                % kontroll

ans =

    1.0000    2.0000    3.0000
    4.0000    5.0000    6.0000
    7.0000    8.0000    0.0000


>> p = poly(A)     % karakteristiska polynomets koefficienter i fallande potens


p =

    1.0000   -6.0000  -72.0000  -27.0000

>> roots(p)         % polynomets nollställen

ans =                % de är ju egenvärdena till A

   12.1229
   -5.7345
   -0.3884
```

2.3.3. Lösning av ekvationssystem

Ekvationssystem löses behändigt med division operatorerna / och \ . Betraktar ekvationssystemet:

$$\begin{pmatrix} c_{11}X_1 + \cdots + c_{1n}X_n \\ c_{21}X_1 + \cdots + c_{2n}X_n \\ \vdots \\ c_{n1}X_1 + \cdots + c_{nn}X_n \end{pmatrix} = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix}$$

vilket är ekvivalent med

$$\begin{matrix} & \mathbf{C} & & & \mathbf{X} & & \mathbf{A} \end{matrix}$$
$$\begin{pmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & \ddots & \vdots \\ c_{n1} & \cdots & c_{nn} \end{pmatrix} \begin{pmatrix} X_1 \\ \vdots \\ X_n \end{pmatrix} = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix}$$

Ovan antas $n \times n$-matrisen **C** vara given (koefficienterna), vektorn **X** är obekant. Huruvida ekvationen $\mathbf{CX} = \mathbf{A}$ har någon lösning kan kontrolleras m.hj.a. determinanten till **C**,

```
>> det(C)
```

Ifall determinanten till **C** är olika noll har systemet en lösning. Obekanta vektorn **X** fås nu enligt:

```
>> X = C\A                    Alternativt      >> X= inv(C)*A
```

Löser t.ex. följande ekvationssystem:

$$\begin{aligned} 2x + 3y + \ z &= \ 7 \\ 2x - 3y + 2z &= -6 \\ x + \tfrac{1}{2}y - \ z &= \ 3 \end{aligned}$$

vilket kan skrivas enl. beteckningen ovan

$$\begin{matrix} & \mathbf{C} & & & \mathbf{X} & & \mathbf{A} \end{matrix}$$
$$\begin{pmatrix} 2 & 3 & 1 \\ 2 & -3 & 2 \\ 1 & \frac{1}{2} & -1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 7 \\ -6 \\ 3 \end{pmatrix}$$

varav lösningen fås enl.

```
>> X=C\A
   X =
       1
       2
      -1
```

```
>> X=inv(C)*A
   X =
       1
       2
      -1
```

2.3.4. Polynom i Matlab

Polynom representeras som vektorer med koefficienterna i fallande potens . Kommandon för polynomhantering:

poly - om A är en kvadratisk matris, returnerar poly(A) en radvektor bestående av koefficienterna till A:s karakteristiska polynom.

om V är en vektor, returnerar poly(V) en vektor bestående av koefficienterna till ett polynom vars nollställen finns i V.

polyder - returnerar koefficienterna för derivatan.

polyfit - söker upp det polynom av en given grad som i minsta kvadrat mening bäst anpassas till en mängd datapunkter.

polyval - evaluerar ett polynom i en given punkt.

roots - returnerar nollställen till ett polynom.

Antag att vi har ett polynom $x^7 + 3x^6 + 2x^5 + 5x^4 + 5x^2 + 7x + 8$.

```
>> p = [1 3 2 5 0 5 7 8]    % definierar polynomet
>> roots(p)                 % beräknar nollställen
ans =
      -2.9446
       0.8813 + 0.8131i
       0.8813 - 0.8131i
      -0.2130 + 1.4397i
      -0.2130 - 1.4397i
      -0.6959 + 0.6386i
      -0.6959 - 0.6386i

>> polyval(p,2)             % beräknar värdet då x=2

ans =

      506

>> y = polyder(p)           % deriverar polynomet

y =

    7    18    10    20     0    10     7
```

Med **polyfit** sökes det polynom av given grad som i minsta kvadrat mening bäst anpassas till en mängd datapunkter.

```
>> format long
>> x = 0:9;
>> y = sin(x);             % definierar 10 datapunkter
>> z = polyfit(x,y,9)'     % bestämmer bästa approximerande polynomet av grad 9

z =

   -0.00000039797827
    0.00000091590432
    0.00029667775794
   -0.00557313510195
    0.03845569568488
   -0.09258029863236
   -0.00370254524941
   -0.15042632647134
    1.05500039889359
    0.00000000000056

>> sin(4.5)-polyval(z,4.5)   % testar om det approximerar bra i punkten 4.5

   ans =

         -1.717375543810462e-04


>> x=0:0.01:2*pi;
>> fel=sin(x)-polyval(z,x);   % felen mellan 0 och 2*pi
>> norm(fel,inf)              % beräknar max( abs(fel) )

   ans =
          0.00668112394493
```

Med kommandot **polyfit** har vi ena foten inne på området optimering. Matlab har en toolbox *optim* med en massa effektiva optimeringsrutiner. Vi beskriver hur man kan lösa följande problem i linjär programmering med Matlab.

Bestäm det $x = (x_1, x_2, x_3)$ som minimerar $-5x_1 - 4x_2 - 6x_3$ under bivillkoren

$$x_1 - x_2 + x_3 \leq 20, 3x_1 + 2x_2 + 4x_3 \leq 42, 3x_1 + 2x_2 \leq 30, x_1, x_2, x_3 \geq 0.$$

Syntaxen är;

```
x = lp(f,A,b,VLB,VUB) löser linjär programmerings problemet:

            min f'x    under bivillkoren   Ax <= b,  VLB < X < VUB
             x
```

```
>> f = [-5 -4 -6];
>> A = [1 -1 1
        3 2 4
        3 2 0];
>> b = [20 42 30]';
>> x=lp(f,a,b,zeros(3,1))  % zeros(3,1) är en kolonnvektor av nollor svarande mot VLB
                           % VUB fattas => oändligheten är övre gräns

x =

         0
   15.0000
    3.0000
```

Mera information om Matlabs optimerings toolbox fås med kommandot

```
>> help optim
```

Med kommandot **optdemo** startas en demonstration av ovannämnda toolbox.

2.4. Grafik

Matlab erbjuder goda möjligheter till grafisk visualisering av data. Vid plottning öppnas ett fönster där visualiseringen sker.

På arbetsstationer (*SPARC*-maskiner) ställer man före start av Matlab in displayen med kommandot: setenv DISPLAY xxx:0 där xxx är namnet på station (t.ex. qed, tor etc.). Vid användning av Matlab på PC:n från X-Win programmet bör också displayen ställas in enligt kommandot ovan. PC-versioner av Matlab (Matlab for Windows) kräver inga dylika inställningar.

På Televideo terminaler finns ingen möjlighet att rita bilder på skärmen. På Falco terminaler däremot kan man få grafiken på skärm. I Matlab ges kommandot **terminal** varefter nummer 10 väljs (emulering av tektronix). Man byter till grafikskärm med Ctrl tangenten nedtryckt och trycker två gånger på F6.

2.4.1. Plottning i 2-dimensioner

Några kommandon för plottning:

```
plot       % plottning av datavektorer
fplot      % plottning av funktioner
subplot    % indelning av fönster
title      % titel ovanför figuren
xlabel     % titel på x-axeln
ylabel     % titel på y-axeln
print      % utskrift av graf till printer el.  ps-fil
figure     % skapar nytt fönster
grid       % skapar rutmönster i en figur
axes       % skapar axlar i en figur
axis       % ställer axlarnas min- och max-värden
clf        % tömmer figuren
```

Kommandot för plottning av funktioner och data är **plot**. Givet två vektorer $x = x_1, ..., x_n$ och $y = y_1, ..., y_n$. Grafen av (x_i, y_i) i ett xy-koordinatsystem med automatisk skalinställning och solid linjetyp ritas då med kommandot

```
>> plot(x,y);
```

Detta kommando var ett specialfall av kommandot **plot(x,y,S)**, där S är en sträng innanför ' ' tecken av maximalt tre tecken bildade av

	Färgerna	markörer
y	gul	.
m	magenta	o
c	cyan	x
r	röd	+
g	grön	*
b	blå	-
w	vit	:
k	svart	-.
		--

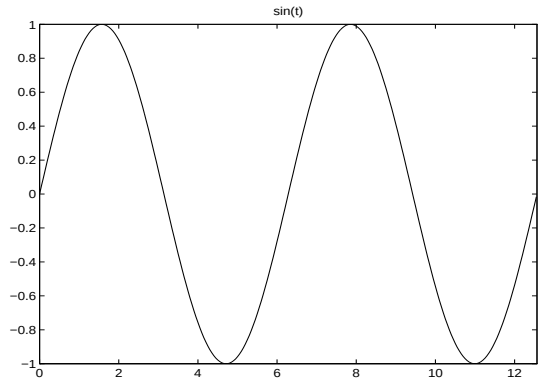
T.ex. ritar **plot(x,y,'b.')** ut den första grafen med små blå punkter.

Om manuell skalinställning önskas görs detta med kommandot

```
>>axis([xmin,xmax,ymin,ymax]);
```

Exempel

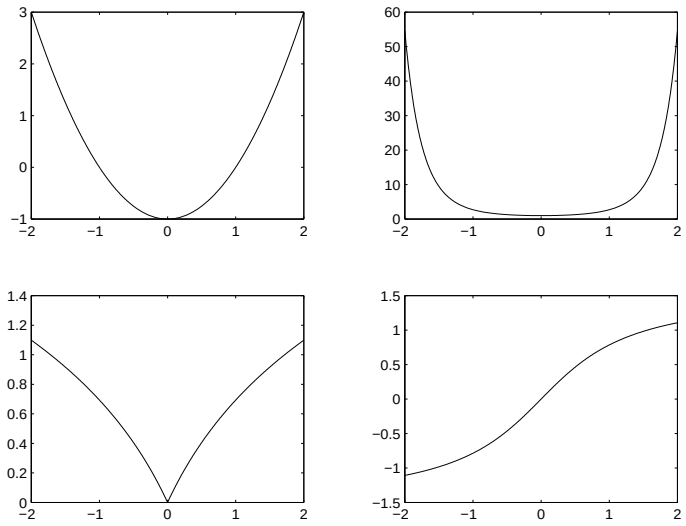
```
>> t=0:.05:4*pi;
>> plot(t,sin(t)), title('sin(t)'), axis([0 4*pi -1 1]);
```



Figur 2.1. $sin(t),\ t \in [0, 4\pi]$

Om z är en komplex vektor så är **plot(z)** liktydigt med **plot(real(z),imag(z))**. Om man önskar plotta flera kurvor i komplexa talplanet i samma graf måste man använda det senare alternativet. Om man tar ut en graf på papper så täcker den övre halvan av ett A4-ark. Man kan uppdela detta område i fönster och rita ut flera grafer på samma papper med **subplot** kommandot. **subplot(m,n,p)** definierar en uppdelning i m*n fönster i m rader och n kolonner. p avser det p:te fönstret räknat radvis börjande i nordvästra hörnet. Exempelvis kan man rita ut fyra grafer på samma papper på följande sätt.

```
>> x=-2:.01:2; y1=x.^2-1; y2=exp(x.^2); y3=log(abs(x)+1); y4=atan(x);
>> subplot(2,2,1),plot(x,y1),subplot(2,2,2),plot(x,y2);
>> subplot(2,2,3),plot(x,y3),subplot(2,2,4),plot(x,y4);
```



Figur 2.2. Användning av subplot-kommandot.

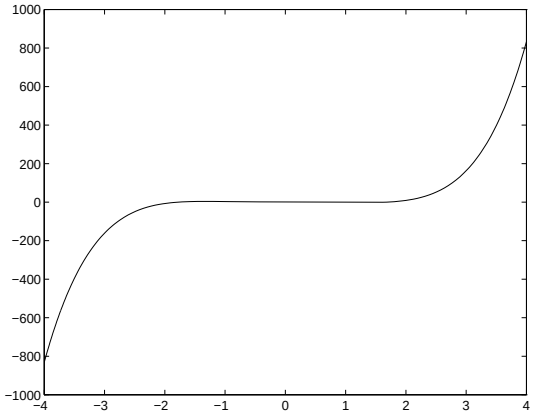
2.4.2. Plottning av funktioner

Ett altenativt sätt att rita $sin(t)$ i intervallet $[0, 4\pi]$ är med kommandot **fplot**:

```
>> fplot('sin',[0,4*pi])
```

fplot(fname,limits) används genom att ange funktion resp. gränser. Funktioner anges inom tecknen ' ' t.ex. '2+x'. Gränser kan anges enl. [XMIN XMAX] el. [XMIN XMAX YMIN YMAX]. Polynom plottas behändigt m.hj.a. detta kommando.

```
>> fplot('x^5-3*x^3+1',[-4,4]);
```

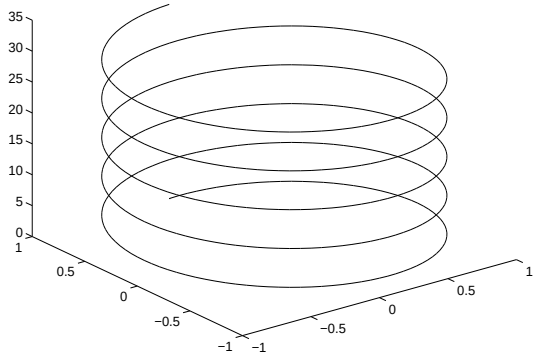


Figur 2.3. $x^5 - 3x^3 + 1,\ x \in [-4, 4]$

2.4.3. Plottning i 3-dimensioner

Parameterkurvor i rymden åstadkoms med **plot3**.

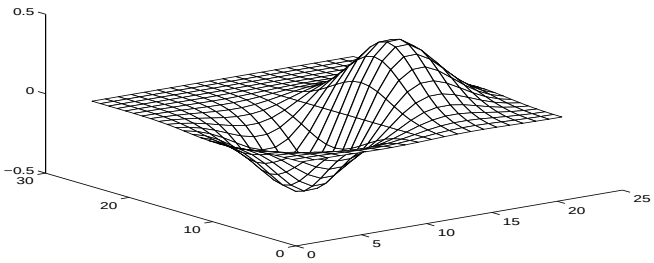
```
>> t=0:pi/50:10*pi;
>> plot3(sin(t),cos(t),t)
>> print bil.ps
```



Figur 2.4. $(sin(t), cos(t)),\ t \in [0, 10\pi]$

Antag att vi vill grafiskt framställa en yta, $f(x,y) = x * exp(-x^2 - y^2)$, över området $(-2,2) \times (-2,3)$. Enklast görs detta med **meshgrid** och **mesh** kommandona.

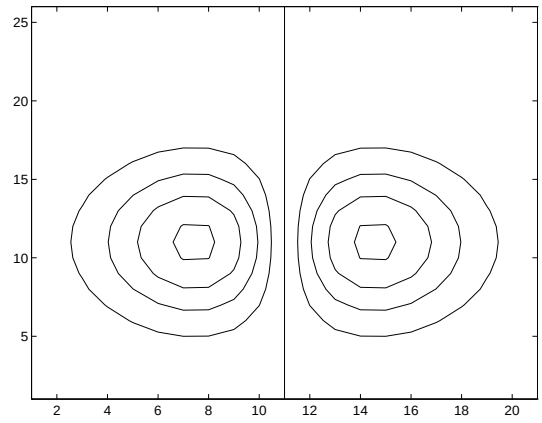
```
>>[x,y]=meshgrid(-2:0.2:2,-2:0.2:3);
>>z=x.*exp(-x.^2-y.^2);
>>mesh(z);
>>print fil.ps   % skapar filen fil.ps som lagras i dierctoryn
```



Figur 2.5. ytan $x * exp(-x^2 - y^2)$, $x \in (-2,2)$, $y \in (-2,3)$.

Om z bildats som ovan fås en framställning av nivåkurvorna med

```
>>contour(z);
```



Figur 2.6. Nivåkurvor till ytan i figur 5.

2.4.4. Utskrift av figurer

Figurer skrivs ut från Matlab med **print**-kommandot. Figurer kan skrivas ut till en fil (figur.ps) enligt

```
>> print figur.ps
```

Alternativt kan en figur skrivas ut direkt till default-printern enligt

```
>> plot(x,y)          % plottning
>> print              % print-kommandot
```

Observera att vid **print**-kommando skrivs den senast skapade figuren ut, därav är det behändigast att skriva ut figurerna efter respektive plottning.

Exempel

```
>> t=0:0.05:4*pi;
>> y=sin(t);
>> plot(t,y);
>> print graf.ps
```

Kommandona ovan ritar ut $sin(t)$ i intervallet $[0, 4\pi]$ med steglängd 0.05 på grafisk terminal och sparar grafen i (PostScript-)filen graf.ps. PS-filen kan ritas ut på lasern (mate) med kommandona:

```
>> !lpr -Pmate graf.ps       % inne i Matlab

 % lpr -Pmate graf.ps        % utanför Matlab
```

Figuren i exemplet ovan kan även skapas och skrivas ut, på default-printern, enligt följande

```
>> fplot('sin',[0,4*pi]);
>> print
```

2.5. Programmering i Matlab

Matlab består av många användbara funktioner vilket medger funktionell programmering. Egna funktioner och kommandon kan skrivas i .m-filer, en stor del av de inbyggda funktionerna i Matlab finns belägna i .m-filer.

Matlab tillhandahåller bl.a. följande styrstrukturer

```
FOR - loop:              WHILE - loop:        IF och BREAK:
>>for i=1:m               >>n=1;              >>if n<10
    for j=1:n             >>while n<100         statements;
      a(i,j)=1/(i+j-1);       statements;     elseif n=10,break,end;
    end;                      n=n+1;          else
  end;                      end;                statements;
                                                end;
```

Matlab program kan skrivas i disk-filer med .m förlängning. Man bör (på Aton) starta Matlab från samma underområde där ifrågavarande filer finns. Dessa program kan sedan aktiveras under körning av Matlab programmet genom att skriva filnamnet. Man kan med fördel ha en fil med namnet **startup.m** lagrad i underområdet. Varje gång Matlab startas utförs de instruktioner som finns i denna fil. Man kan t.ex. ha definitioner på konstanter i filen.

Följande *fibo.m*-fil beräknar Fibonacci-tal.

```
function
% .m - fil som beräknar Fibonacci-tal.
k=input(' Ge talet fram till vilket fibonacci talen beräknas ');
f=[1 1];i=1;
while f(i) + f(i+1) < k
  f(i+2)=f(i)+f(i+1);
  i=i+1;
end;
disp('fibonaccitalen under det inlästa talet');
disp(f');
```

Som *fibo.m*-filen nu fungerar så definieras vektorn om i varje loop, vilket gör det hela långsamt. Processen kan snabbas upp genom att skapa en stor vektor av f, t.ex skriva $f(10000) = 0$ efter input satsen i *fibo.m*-filen.

Programmet körs från Matlab genom att skriva *fibo*

```
>> fibo
   Ge talet fram till vilket fibonacci talen beräknas?  500
   Fibonacci talen under det inlästa talet
       1
       1
       2
       .
       .
     233
     377
```

Man kan definiera egna funktioner (funktionsprocedurer) i Matlab. Dessa är vanliga .m-filer där första raden definierar parameteröverföringen. I nedanstående exempel har vi en funktion **mean** som beräknar medelvärdet av elementen i varje kolonn i matrisen x och återsänder resultatet i vektorn y. Om x är en vektor beräknas medeltalet av elementen i x. Följande finns i filen mean.m :

```
function y=mean(x)
[m,n]=size(x);
if m == 1
   m=n;    % x radvektor
end;
y=sum(x)/m;
```

I Matlab förefaller det som om vi skulle kalla på en inbyggd funktion.

```
>> z=1:10;
>> mean(z)
   ans =
         5.5000
```

Allmänt om man har inparametrarna x1, ... ,xn och utparametrarna y1, ... ,ym så antar funktionsdeklarationen utseendet

```
function [y1,y2,...,ym]=fname(x1,x2,...,xn)
```

och anropet från Matlab blir då

```
>>[a1,a2,...,am]=fname(b1,b2,...,bn);
```

Vi tar ett avslutande exempel. Antag att vi har $n+1$ stycken datapunkter $(x_i, f(x_i)),\ i = 0,\ \ldots\ ,n$ givna. Vi vill bestämma det entydiga interpolationspolynom p av grad n som går igenom dessa datapunkter. Detta kan göras t.ex. med Lagranges interpolationsformel. Då ges p med hjälp av basfunktionerna l_k enligt

$$l_k(x) = \prod_{\substack{j=0 \\ j\neq k}}^{n} \frac{x - x_j}{x_k - x_j},\ k = 0,\ \ldots\ ,n$$

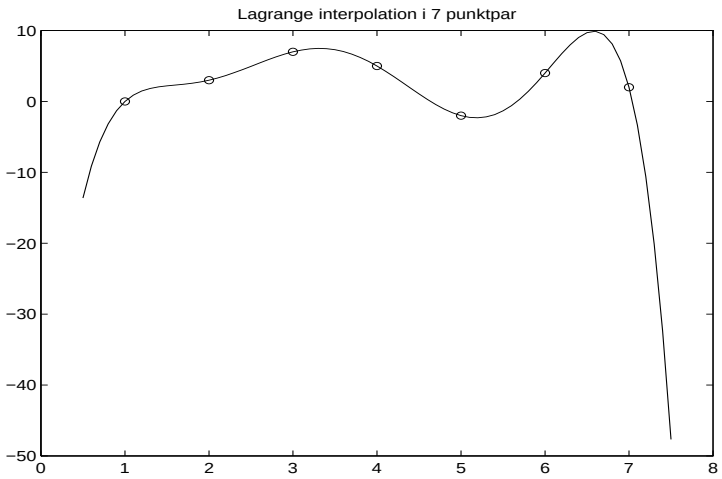
$$p(x) = \sum_{j=0}^{n} f(x_j) l_j(x).$$

I följande två funktionsprocedurer beräknar vi l_k och p.

```
function [x]=lk(k,xi,a,b,step);
 % The function lk(k,xi,a,b,step) calculates the k:th basis function
 % of the Lagrange interpolation formula in the interval (a,b) with
 % step being the increment, and xi containing the x-coordinates of
 % the interpolation points.

 xx=a:step:b;
 x=1;
 for i=1:length(xi)
   if i ~= k
     x=x.*((xx-xi(i))./(xi(k)-xi(i)));
   end
 end


 function [x]=lip(xi,fxi,a,b,step);
 % Calculates the Lagrange interpolation polynomial in the interval
 % (a,b) with increment step to datapoints (xi,fxi) stored in the
 % vectors xi and fxi.
 x=0;
 for i=1:length(xi)
   x=x+fxi(i)*lk(i,xi,a,b,step);
 end;
```

I Matlab tillämpar vi funktionerna på 7 punktpar.

```
>> x=[1,2,3,4,5,6,7];
>> fxi=[0,3,7,5,-2,4,2];
>> p=lip(x,fxi,0.5,7.5,0.1);
>> plot(x,fxi,'o',0.5:0.1:7.5,p),title('Lagrange interpolation i 7 punktpar');
>> print lag.ps
>> !lpr -Pmate  lag.ps
```



Figur 2.7. Lagrange interpolation.

2.6. Matlab version 5

Version 5 av Matlab innehar många nya funktioner både för beräkning och visualisering. Hjälpfilerna kan här med kommandot **helpdesk** läsas med en Web-browser. Från web-sidan *http://www.mathworks.com*, hittas även de nyaste nyheterna om Matlab.

2.6.1. Nya datastrukturer och funktioner

I Matlab 5 kan vektorer och matriser definieras flerdimensionellt t.ex. enligt:

```
>> A=rand(2,2,3);          % A är en 2*2*3 matris, dvs.  3 st.  2*2-matriser

>> B=ones(1,2,3,4);        % B består av 3*4 st.  1*2 matriser
```

Funktionerna: **ones**, **zeros**, **rand** och **randn** kan användas enligt ovan. Vidare finns funktioner för generering av glesa matriser; **sparse**, **sprand** och **sprandn**.

2.6.2. Nya grafiska finesser

TEX symboler och text kan här användas vid skapandet av figurer. De vanligaste grekiska bokstäverna och matematiska symboler kan användas som text i Matlab. Från tabellen nedan ses en del av de TEX kommandon som fungerar i Matlab 5 †.

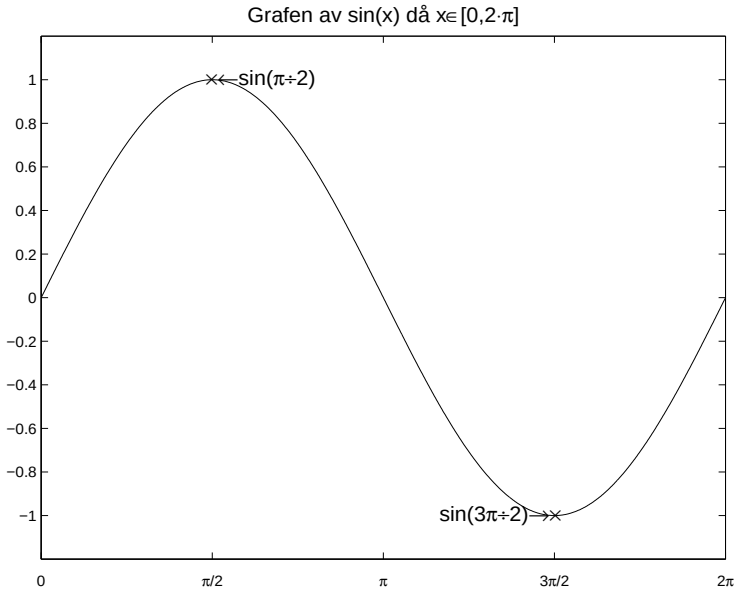
Tabell 1. Endel tillgängliga TEX kommandon i Matlab.

Kommando	Bokstav	Kommando	Tecken
\alpha	α	\int	$\int$
\beta	β	\sum	$\sum$
\gamma	γ	\in	$\in$
\delta	δ	\infty	∞
\epsilon	ϵ	\leq	$\leq$
\zeta	ζ	\geq	$\geq$
\eta	η	\leftarrow	$\leftarrow$
\theta	θ	\uparrow	$\uparrow$
\kappa	κ	\rightarrow	$\rightarrow$
\lambda	λ	\downarrow	$\downarrow$
\pi	π	\forall	$\forall$
\omega	ω	\exists	$\exists$
\mu	μ	\approx	$\approx$
\xi	ξ	\subset	$\subset$
\sigma	σ	\pm	$\pm$
\rho	ρ	\neq	$\neq$
\tau	τ	\div	$\div$

† En fullständig förteckning över tillgängliga TEX symboler hittas från *Help Desk* under **text**

TEX symbolerna kan användas i funktionerna: **text**, **title**, **xlabel** samt **ylabel**. Vid plottningen i figur 2.8 av $sin(x),\ x \in [0, 2\pi]$, användes även funktionerna † **set** och **XTickLabel** för att justera markeringen på X-axeln.

```
>> x=0:.01:2*pi; y=sin(x);
>> plot(x,y),axis([0 2*pi -1.2 1.2]);
>> title('Grafen av sin(x) då x\in[0,2\cdot\pi]','FontSize',14);
>> text(pi/2,sin(pi/2),'\times\leftarrowsin(\pi\div2)','FontSize',14);
>> text(3*pi/2,sin(3*pi/2),'sin(3\pi\div2)\rightarrow\times'...
   'HorizontalAlignment','right','FontSize',14);
>> set(gca,'XTick',[0 pi/2 pi 3*pi/2 2*pi],'XTickLabel','     ');
>> text(0,-1.3,'0','HorizontalAlignment','Center');
>> text(pi/2,-1.3,'\pi/2','HorizontalAlignment','Center');
>> text(pi,-1.3,'\pi','HorizontalAlignment','Center');
>> text(3*pi/2,-1.3,'3\pi/2','HorizontalAlignment','Center');
>> text(2*pi,-1.3,'2\pi','HorizontalAlignment','Center');
```



Figur 2.8. Användning av TEX vid plottning.

† Se hjälpfunktionerna för respektive funktion.

2.6.3. Programmering

Matlab 5 har en egen editor för editerng av .m-filer, vilken innehar behändiga redskap för editering av program.

Likt C-kod kan även här användas **switch**-satser

```
switch input
     case -1
          disp('Input var -1');
     case 0
          disp('Input var 0');
     case 1
          disp('Input var 1');
     otherwise
          disp('Input var nåt annat.');
end
```

Matlab 5 har även effektivare arbetsprincip vid kontroll av *if*-satser, ty satserna kontrolleras nu från vänster till höger, med stop genast ett villkor kan avgöras.

I exemplet nedan kollas ej andra villkoret $(b < 1)$ om $a \geq 1$.

```
>> if ( a<1 & b<1 )
     c=0;
   else
     c=-1;
   end;
```

Matlab erbjuder även möjlighet att kalla på annan kod t.ex. ett C-program från .m-filer samt vice versa.

Referenser

[1] Hill D.R.: Experiments in Computational Matrix Algebra.

[2] The MathWorks Inc.: Using Matlab, 1993.

[3] The MathWorks Inc.: Matlab The Language of Technical Computing: Matlab 5 New Features, 1996.

[4] Van Loan C.: Introduction to Scientific Computing: A Matrix-Vector Approach Using MATLAB, 1997.

Övningsuppgifter

1. Definiera en ekvidistant vektor $\mathbf{z} \in [0, 10]$, pröva sen på t.ex.: $2 + z$, $z.\hat{}2$, $max(z)$, $min(z)$, $sum(z)$ samt $plot(z, z, 'o')$ resp. $plot(z, z, '*')$.

2. Sätt $zz = cos(z)$ och pröva $plot(z, zz)$. Genom att öka antalet punkter i vektorerna z, zz fås en finare graf.

3. Läs in en (5×5) *permutationsmatris* A:

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

4. Undersök A^n:s beteenden för $n = 2, 3, 4, 5, 6, 15, 20....$

5. Knacka sen in

$$D = \begin{pmatrix} 0.2 & 0.5 & 0.3 & 0 \\ 0.3 & 0 & 0.4 & 0.3 \\ 0.3 & 0.2 & 0.2 & 0.3 \\ 0.1 & 0.3 & 0.4 & 0.2 \end{pmatrix}$$

Vad händer med D^n för stora värden på n ? Samma fråga för:

$$D = \begin{pmatrix} 0.2 & 0.5 & 0.3 & 0 \\ 0.3 & 0 & 0.4 & 0.3 \\ 0.3 & 0.2 & 0.2 & 0.3 \\ 0.2 & 0.3 & 0.1 & 0.4 \end{pmatrix}$$

6. Lös ekvationssystemet

$$2x - 5y = 13$$
$$-3x + 10y = 8$$

Läs in koefficienterna på vänstra sidan i en koefficientmatris **A** och högra sidan i en kolonnvektor **b**. Lösningen till Ax=b erhålls med kommandot $x = A\backslash b$. Bilda vektorn $z = 33 : 0.1 : 35$ svarande mot en diskretisering av intervallet (33,35). Bilda vektorerna

$$y_1 = \frac{2}{5} * z - \frac{13}{5}$$
$$y_2 = \frac{3}{10} * z + \frac{4}{5}$$

Checka lösningen grafiskt med kommandot

$$plot(z, y_1, z, y_2)$$

7. Skriv in matriserna

$$A = \begin{pmatrix} 3 & 8 \\ -7 & 4 \end{pmatrix}, \; B = \begin{pmatrix} -1 & 6 \\ 2 & 5 \end{pmatrix}$$

Beräkna a) $A+B$ b) $A*B$ c) $B*A$ d) Determinanten av A och B e) Inversen av A och B f) Egenvärdena till A och B.

8. Upprita $sin(\frac{1}{x})$ i intervallen: a) $x \in [-10,10]$, b) $x \in [-1,1]$ och c) $x \in [-0.01,0.01]$.

9. Upprita trochoiden:

$$x(t) = at - bsin(t)$$
$$y(t) = a - bcos(t)$$

för $t \in [0,6\pi]$ med a) $a = b = 2$, b) $a = 2, b = 2.5$ samt c) $a = 2, b = 1$.

Sätt först $t = 0 : .1 : 6 * pi$; sen $xa = 2t - 2sin(t)$; $ya = 2 - 2cos(t)$; och plotta enl.: $plot(xa, ya)$ o.s.v.. Försök plotta fallena a)-c) i tre olika fönster m.hj.a. *subplot*-koomandot, samt även i samma koordinatsystem.

10. Sök nollställena till:

$$a)\ f(x) = x^4 + 2x^3 + 29x^2 + 128x - 2240$$
$$b)\ f(x) = 3x^5 - 7x^2 + 8x - 21$$

11. Bilda 15 punktpar plockade från f(x)=xsin(x) d.v.s.:

$$x2 = 1 : 15;$$
$$y2 = x2. * sin(x2);$$

Plotta punkterna med plot(x2,y2,'*'); Diskretisera intervallet (1,15) i x med x=1:0.01:15 . Beräkna nu den i minsta kvadratmening bästa polynom anpassningen till punkterna med kommandot **polyfit**. Bilda 4 vektorer,pn , n=1, 4, 8, 10, svarande mot koefficienterna i fallande potens för den bästa anpassningen av grad n med kommandona

$$pn = polyfit(x2, y2, n)$$

Beräkna funktionsvärden för polynomen pn i intervallet (1,15) med kommandot

$$yn = polyval(pn, x);$$

Plotta sedan polynomen med kommandot

$$plot(x2, y2,' *', x, y1, x, y4, x, y8, x, y10)$$

för att grafiskt se hur goda anpassningar som åstadkommits.

12. Punkter i planet tillhörande *Ikedas attraktor* kan genereras med formeln

$$x_{n+1} = 0.97 + 0.9(x_n cos(t) - y_n sin(t))$$
$$y_{n+1} = 0.9(x_n sin(t) + y_n cos(t))$$
$$t = 0.4 - \frac{6.0}{1.0 + {x_n}^2 + {x_n}^2}$$
$$x_1 = y_1 = 0$$

Med följande programsnutt kan du generera vektorena x, y bestående av 5000 punkter i *Ikedas attraktor*.

$$
\begin{aligned}
&x(1) = 0; y(1) = 0;\\
&for\ \ i = 1 : 4999\\
&\ \ t = 0.4 - 6.0/(1.0 + x(i)^2 + y(i)^2);\\
&\ \ x(i+1) = 0.97 + 0.9 * (x(i) * cos(t) - y(i) * sin(t));\\
&\ \ y(i+1) = 0.9 * (x(i) * sin(t) + y(i) * cos(t));\\
&end;
\end{aligned}
$$

Punkterna kan plottas med $plot(x, y, '.')$;

13. Prova att editera t.ex. följande **grad.m** -fil

$$
\begin{aligned}
function\ nn = &grad(xx, yy)\\
&[x, y] = meshgrid(-xx : .2 : yy);\\
&z = x. * exp(-x.^2 - y.^2);\\
&[dx, dy] = gradient(z, .2, .2);\\
&contour(x, y, z)\\
&hold\ on\\
&quiver(x, y, dx, dy)\\
&hold\ off
\end{aligned}
$$

I matlab anropas funktionen **grad** med t.ex. parametrarna (2,2) med kommandot **grad(2,2)**. **grad**-funktionen ritar alltså ut gradient fältet för funktionen $z = xe^{-x^2-y^2}$ för $x \in [xx, yy]$ samt $y \in [xx, yy]$.

14. Matlab löser även ordinära differentialekvationer, kolla demoprogrammet för detta. Ge in kommandot **odedemo**.

15. Prova att editera och använda följande **.m**-filer i matlab:

Fibonaccitalen: $fib(n) = fib(n-1) + fib(n-2)$, $fib(1) = fib(2) = 1$;.

```
function y=fib(n);
if (n==1 | n==2)
  y=1;
else
  y=fib(n-1)+fib(n-2);
end;
```

Denna anropas från Matlab med kommandot: $fib(tal)$.

Fakulteten: $n! = n \cdot (n-1) \cdots 2 \cdot 1$, m.a.o. $fak(n) = n \cdot fak(n-1)$.

```
function y=fak(n);
if (n==0 | n==1)
  y=1;
else
  y=n*fak(n-1);
end;
```

Anropas med: $fak(tal)$

3. Macsyma

MACSYMA är ett expertsystem för symbolisk manipulation, "bokstavsräkning", bestående av mer än 300.000 rader LISP-kod. Man kan med MACSYMA erhålla en exakt symbolisk lösning till ett stort antal olika matematiska problem. MACSYMA är ett fullständigt interaktivt programpaket med över 1000 olika kommandon för manipulation av uttryck. Därtill erbjuder MACSYMA även en ALGOL-liknande programmeringsomgivning med kommandon för manipulation av externa filer.

Fördelar med MACSYMA:

1. Man kan använda MACSYMA som en "symbolisk kalkylator" för att kontrollera räkningar gjorda för hand.
2. Symbolisk lösning av uppgifter som i princip kan lösas för hand, men är onödigt tidskrävande.
3. Exakt lösning av delproblem i en numerisk algoritm, varvid effekten av avrundningsfel kan minskas.

3.1. Start av Macsyma

Macsyma startas (på Aton) med kommandot: **macsyma**

Nedan följer en exempelsession med inskrivna kommentarer. Notera att exemplena är körda på Åbo Akademis Sun dator **Aton**. Här presenteras hjälpkommandona **describe, apropos** och **example**, samt filhanteringskommandona **writefile, closefile, save, loadfile** och **batchload.**

```
% macsyma
Starting Macsyma math engine ......
This is ........
```

MACSYMA tillhandahåller tre slags numrerade lägen. Lägen med inledande C - bokstav är kommando lägen där användaren interaktivt specificerar önskad åtgärd. Kommandon avslutas med semikolon. Resultatet av ett komando skrivs ut på en D - rad med samma numrering som kommandoraden. Vid krävande beräkningar kan också mellanresultat skrivas ut på rader med inledande E - bokstav. Man kan under en session använda tidigare lägen genom att referera till dessa.

```
(c1) writefile("mac.txt");       / Sparar sessionen i filen mac.txt.  /
(d1)                      /home/aton/cglader/mac.txt

(c2) DESCRIBE(SUM);   / Kommandot describe(xxx) ger en kort beskrivning
                        av macsyma kommandot xxx.  /
SUM(exp, ind, lo, hi) performs a summation of the values of exp as
the index ind varies from lo to hi.    If the upper and lower limits
differ by an integer then each term in the sum is evaluated and added
together.    Otherwise, if the SIMPSUM [FALSE] is TRUE the result is
simplified.    This simplification may sometimes be able to produce a
closed form.    If SIMPSUM is FALSE or if 'SUM is used, the value is a
SUM noun form which is a representation of the sigma notation used in
mathematics.

If hi is one less than lo, we have an "empty sum" and SUM returns 0
rather than erring out.

Sums may be differentiated, added, subtracted, or multiplied with some
automatic simplification being performed.

SUM evaluates its first argument only once like almost every other
Macsyma command.    This implies that things like F(X):=SUM(X,I,1,3)$
F(-X); (Answer:  -3*X) and SUM(T[I],I,1,3); (Answer:  retrieve and sum
up T[1],T[2],T[3] without reevaluating the elements) will work
correctly whereas I^2; SUM(%,I,1,3); (Answer:  3*I^2) will not (if the
desired answer is 14).    To do the latter one will have to type
SUM(EV(%),I,1,3); or SUM(''%,I,1,3); .    (PRODUCT and ROMBERG behave
similarly.)

CAUCHYSUM[FALSE] when TRUE causes the Cauchy product to be used when
multiplying sums together rather than the usual product.    In the
Cauchy product the index of the inner summation is a function of the
index of the outer one rather than varying independently.

GENINDEX[I] is the alphabetic prefix used to generate the next
variable of summation.

GENSUMNUM[0] is the numeric suffix used to generate the next variable
of summation.    If it is set to FALSE then the index will consist only
of GENINDEX with no numeric suffix.

Do EXAMPLE(SUM); for examples.

See also NUSUM, SIMPSUM, SUMEXPAND, SUMHACK, INTOSUM, SUMCONTRACT,
BASHINDICES, NICEINDICES.
```

```
(c3) APROPOS(SUM);
(d3) [ALGEBRA_SUMMARY, ARRAY_SUMMARY, ASSUME, ASSUMESCALAR, ASSUMESCALARP,
ASSUME_POS, ASSUME_POS_PRED, CALCULUS_SUMMARY, CAUCHYSUM,CHANGE_SUM_LIMITS
DIVSUM, FACSUM, FACSUM_COMBINE, FACTORFACSUM, FACTORSUM, GENSUMNUM, GFAC-
TORSUM,INDEFSUM, INTEGRATION_SUMMARY, INTOSUM, MATRIX_ELEMENTS_ASSUMED_-
SCALAR, MATRIX_SUMMARY, MAT_CUMSUM, MAT_SUM, NEGSUMDISPFLAG, NUSUM, NUSUM1,
PLOTTING_SUMMARY, RULES_SUMMARY, SIMPSUM, SUM, SUMCONTRACT, SUMEXPAND,
SUMHACK,SUMSPLITFACT, TRIG_SUMMARY, UNSUM]
(c5) CLOSEFILE();    / Stänger logfilen.  /
```

Man kan skriva in MACSYMA kommandon i en editor utanför MACSYMA och läsa in filen med **batchload**. I exemplet har funktionen $f(x) := cos(x)$; definierats i den externa filen trigfil.txt.

```
(c2) BATCHLOAD("trigfil.txt"); / Läser in definitionerna i trigfil.txt.  /

Batching the file trigfil.txt
Batchload done.

(d2)                                  DONE
(c3) F(X);


(d3)                                 COS(X)
```

Med kommandot **save** kan man spara definitioner och resultat i en extern fil för att senare, under en ny session, läsa in dem med **loadfile** kommandot.

```
(c2) F(X):=X-SIN(X);
(d2)                           F(X) := X - SIN(X)


(C3) G(X):=COS(X);
(d3)                             G(X) := COS(X)

(c5) SAVE("sparfil.txt",F,G);

(d5)                        [/home/aton/cglader/sparfil.txt, f, g]
```

I en ny MACSYMA session kan vi läsa in de sparade definitionerna.

```
(c2) LOADFILE("sparfil.txt");
/home/aton/cglader/sparfil.txt  being loaded.
(d2)                     /home/aton/cglader/sparfil.txt
(d3) F(X);


(d3)                                X - SIN(X)
(c4) G(X);

(d4)                                  COS(X)
```

Quit kommandot avslutar en MACSYMA session.

```
(c70) QUIT();         / Går ut ur MACSYMA./
```

3.2. Manipulering av algebraiska uttryck

I detta avsnitt presenteras en del användbara kommandon för omformning av uttryck. I form av en exempelsession presenteras kommandona **num**, **denom**, **realpart**, **imagpart**, **ev**, **fpprec**, **bfloat**, **expand**, **ratsimp**, **factor**, **partfrac**, **trigsimp**, **part**, **lhs** och **rhs**.

```
(c2) (5*%E-%PI+2*%I)/20;   / Bildar en kvot.  Reserverade konstanter /
                           / föregås av %-tecken.  /

                               - %PI + 2 %I + 5 %E
(d2)                           -------------------
                                       20
(c3) NUM(%);               / Plockar ut täljaren, %-tecknet refererar /
                          / till föregående läge.  /


(d3)                           - %PI + 2 %I + 5 %E
(c4) DENOM(D2);           / Plockar ut nämnaren i uttryck (D2).  /

(d4)                                   20
(c5) REALPART(D2);               / Realdelen.  /


                                   5 %E - %PI
(d5)                               ----------
                                       20
```

```
(c6) IMAGPART(D2);                / Imaginära delen.  /


                                      1
(d6)                                  --
                                      10
(c7) EV(D5,NUMER);                / Erhåller ett flyttalsnärmevärde /
                                  / för (D5), antagen precision 6 dec.  /

(d7)                                0.522491
(c8) FPPREC:35;                   / Ställer om precisionen.  /

(d8)                                  35

(c9) BFLOAT(D5);

(d9)              5.22490824435271646916939698674190 48B-1

(c10) FPPREC:6;                   / Återställer precisionen /

(d10)                                 6

(c15) A:3*X^2-5*X+9;              / Definierar ett polynom som /
                                  / knyts till läget A. /

                                    2
(d15)                             3 X  - 5 X + 9
(c16) EV(A,X = 6/7);              / Evaluerar uttrycket för läget A /
                                  / med parametern X = 6/7.  /

                                     339
(d16)                                ---
                                     49
(c17) F(X):=X^3-7*X+4;            / Definierar en funktion F(X). /

                                      3
(d17)                        F(X) := X  - 7 X + 4

(c18) F(2);
(d18)                                    -2
(c20) F(F(X));

                       3           3       3
(d20)                 (X  - 7 X + 4)  - 7 (X  - 7 X + 4) + 4
(c21) G(X):=F(X)/F(X-1)+F(2)/F(X);
```

```
                                        F(X)     F(2)
(d21)                     G(X) := -------- + ----
                                  F(X - 1)   F(X)

(c28) (X+5)*(X-8)*(X-1)/((X+1)*(X+2)*(X+3));

                          (X - 8) (X - 1) (X + 5)
(d28)                     -----------------------
                          (X + 1) (X + 2) (X + 3)
(c29) EXPAND(%);          / Multiplicerar ut faktorerna och  /
                            och delar upp bråket i en summa.  /

              3                      2
             X                    4 X
(d29) -------------------- - -------------------- -
       3      2              3      2
      X  + 6 X  + 11 X + 6   X  + 6 X  + 11 X + 6

            37 X                   40
      - -------------------- + --------------------
         3      2               3      2
        X   +  X  + 11 X + 6    X  + 6 X  + 11 X + 6

(c30) RATSIMP(%);             / Gör ett uttryck liknämnigt.  /

                               3      2
                              X  - 4 X  - 37 X + 40
(d30)                         ---------------------
                               3      2
                              X  + 6 X  + 11 X + 6

(c31) FACTOR(%);              / Faktorer uttrycket.  /


                          (X - 8) (X - 1) (X + 5)
(d31)                     -----------------------
                          (X + 1) (X + 2) (X + 3)

(c32) PARTFRAC(%,X);          / Utför partialbråksuppdelning.  /


                           44      90      36
(d32)                     ----- - ----- + ----- + 1
                          X + 3   X + 2   X + 1

(c33) SIN(X)/COS(X)+COS(X)/SIN(X);
```

```
                                SIN(X)   COS(X)
(d33)                           ------ + ------
                                COS(X)   SIN(X)

(c34) RATSIMP(%);     / Ratsimp gör uttrycket liknämnigt men       /
                      / kan inte förenkla trigonometriska uttryck.  /

                                  2         2
                               SIN (X) + COS (X)
(d34)                          -----------------
                                 COS(X) SIN(X)

(c35) TRIGSIMP(D33);            / Använd istället trigsimp.  /


                                      1
(d35)                           -------------
                                COS(X) SIN(X)

(c36) PART(D33,1);              / Plockar ut första termen ur d33.  /


                                    SIN(X)
(d36)                               ------
                                    COS(X)
(c37) PART(D33,2);


                                    COS(X)
(d37)                               ------
                                    SIN(X)
(c39) X+4 = Y^2+5;


                                         2
(d39)                           X + 4 = Y  + 5
(c40) LHS(%);                   / Plockar ut vänstra membrum.  /


(d40)                               X + 4

(c41) RHS(D39);                 / Plockar ut högra membrum
```

3.3. Gränsvärden

MACSYMA tillhandahåller kommandot **limit** för beräkning av gränsvärden.

```
(c51) 'LIMIT((COS(X)-1)/X^2,X,0);  / Beräknar ett gränsvärde, apostrofen
                                     upphäver uträkningen./

                                       COS(X) - 1
(d51)                          limit  ----------
                               X -> 0      2
                                          X
(c52) EV(%,LIMIT);                 / Beräknar gränsvärdet./


                                        1
(d52)                                   - -
                                        2
(c53) LIMIT((1+1/X)^X,X,INF);

(d53)                                  %E

(c54) LIMIT(1/X,X,0);              / Gränsvärdet odefinierat.  /
(d54)                                  UND

(c55) LIMIT(1/X,X,0,PLUS);         / Högergränsvärde.  /
(d55)                                  INF

(c56) LIMIT(1/X,X,0,MINUS);        / Vänstergränsvärde.  /
(d56)                                  MINF
```

3.4. Summor och Serier

Med kommandot **sum** kan man försöka summera ett ändligt eller ett oändligt antal termer i en serie.

```
(c8) SUM(I^2+2^I,I,0,N);           / Ger utskrift av summan.  /

                                    N
                                   ====
                                   \       I    2
(d8)                                >    (2  + I )
                                   /
                                   ====
                                   I = 0

(c9) EV(SUM(I^2+2^I,I,0,N),SIMPSUM);      / Beräknar summan.  /
```

```
                                      3      2
                              N + 1   2 N  + 3 N  + N
(d9)                         2      + --------------- - 1
                                            6

(c10) EV(SUM((1/3)^I,I,1,INF),SIMPSUM);   / En oändlig summa.  /

                                          1
(d10)                                     -
                                          2
```

3.5. Derivering och integrering

Kommandona **diff** och **integrate** utför symbolisk derivering respektive integrering. Numerisk integrering kan utföras med **romberg** kommandot.

```
(c29) F(X,Y):=X^2*Y^2+B*X*Y^2-5; / Definierar ett polynom med 2 variabler.

                                      2  2        2
(d29)                    F(X, Y) := X  Y  + B X Y  - 5
(c30) DIFF(F(X,Y),X);             / Deriverar med avseende på X. /

                                        2      2
(d30)                               2 X Y  + B Y
(c31) DIFF(F(X,Y),X,1,Y,2);       / Deriverar 1 gång m.a.p.  X, /
                                  / 2 gånger m.a.p.  Y. /

(d31)                               4 X + 2 B
(c32) INTEGRATE(X*SIN(2*X),X);    / En obestämd integral.  /

                             SIN(2 X) - 2 X COS(2 X)
(d32)                        -----------------------
                                        4

(c35) INTEGRATE(EXP(-X^2),X,0,INF); / Integrerar en exponentialfunktion /
                                    / från 0 till oändligt.  /

                                    SQRT(%PI)
(d35)                               ---------
                                        2
```

```
(c38) F(X,Y):=X*Y*EXP(-(X^2+Y^2));
                                                 2    2
(d38)                   F(X, Y) := X Y EXP(- (X  + Y ))
(c40) 'INTEGRATE('INTEGRATE(F(X,Y),X,0,1),Y,0,1);
       / En dubbelintegral över kvadraten (0,1)x(0,1).  /
       / Apostroferna framför integrate-kommandot upphäver ut- /
       / räkningen och skriver ut dubbelintegralen.  /

                          1    1
                         /    /          2    2
                         [    [      - Y  - X
(d40)                    I  Y I  X %E           dX dY
                         ]    ]
                         /    /
                        0    0

(c41) EV(%,INTEGRATE);    / Evaluering av dubbelintegralen.  /

                                    - 2
                           %E - 1   %E    (%E - 1)
(d41)                      ------ - --------------
                             4 %E          4

(c44) INTEGRATE(SIN(X^2),X,0,1);  / Denna integral klarar inte macsyma av.


                                              SQRT(2) %I + SQRT(2)
(d44) SQRT(%PI) ((SQRT(2) %I + SQRT(2)) ERF(---------------------)
                                                       2

                                   SQRT(2) %I - SQRT(2)
      + (SQRT(2) %I - SQRT(2)) ERF(--------------------))\8
                                              2

(c50) ROMBERG(SIN(X^2),X,0,1);/ Numeriskt närmevärde med Rombergs metod./
(d50)                              0.310269
```

3.6. Serieutveckling

Taylorutveckling möjliggörs med kommandot **taylor**.

```
(c57) TAYLOR(SIN(COS(X)),X,0,5);  / Utvecklar en funktion i Taylorserie i
                                    punkten 0, med avseende på variabeln x.
                                    Trunkering av serien sker vid
                                    gradtalet 5.  /

                              2                          4
                         COS(1) X    (3 SIN(1) - COS(1)) X
(d57)\T\      SIN(1) - --------- - ---------------------- + .  .  .
                             2                 24

(c58) EV(%,NUMER);
                                  4              2
(d58)                - 0.082671 X  - 0.270151 X  + 0.841471
```

3.7. Matriser

För matrismanipulering presenteras kommandona **matrix**, **transpose**, **invert**, **determinant**, **entermatrix**, **ident**, **diagmatrix**, **zeromatrix**, **row**, **addrow**, **addcol**, **rank**, **submatrix**, **eigenvalues** och **eigenvectors**.

```
(c59) A:MATRIX([1,2,3],[2,0,5],[9,8,7]);    / Bildar matrisen A. /

                                  [ 1  2  3 ]
                                  [         ]
(d59)                             [ 2  0  5 ]
                                  [         ]
                                  [ 9  8  7 ]

(c60) B:TRANSPOSE(%);                       / Matristransponat.  /

                                  [ 1  2  9 ]
                                  [         ]
(d60)                             [ 2  0  8 ]
                                  [         ]
                                  [ 3  5  7 ]

(c61) A+B;                                  / Addition av matriser.  /

                                 [ 2   4   12 ]
                                 [            ]
(d61)                            [ 4   0   13 ]
                                 [            ]
                                 [ 12  13  14 ]
```

```
(c62) A*B;                                  / Elementvis multiplikation.  /
                                  [ 1   4   27 ]
                                  [            ]
(d62)                             [ 4   0   40 ]
                                  [            ]
                                  [ 27  40  49 ]

(c63) A . B;                              / "Vanlig" matrismultiplikation.  /
                                  [ 14  17  46  ]
                                  [             ]
(d63)                             [ 17  29  53  ]
                                  [             ]
                                  [ 46  53  194 ]

(c64) INVERT(%);                            / Bestämning av inversen.  /
                          [  2817     43     433  ]
                          [  ----   - ---  - ---- ]
                          [  4900     245    4900 ]
                          [                       ]
                          [   43     6       2    ]
(d64)                     [ - ---    --      ---  ]
                          [   245    49      245  ]
                          [                       ]
                          [   433     2      117  ]
                          [ - ----   ---     ---- ]
                          [   4900   245     4900 ]

(c65) DETERMINANT(%);                       / Bestämmer determinanten.  /
                                    1
(d65)                               ----
                                    4900
(c66) A^^(-1);                            / Matrisexponentiering ^^.  /

                                [   4   1    1   ]
                                [ - -   -    -   ]
                                [   7   7    7   ]
                                [                ]
                                [ 31     2   1   ]
(d66)                           [ --   - -   --  ]
                                [ 70     7   70  ]
                                [                ]
                                [ 8      1    2  ]
                                [ --     -  - -- ]
                                [ 35     7    35 ]
```

```
(c67) B^^(-1) .  B;
                                [ 1  0  0 ]
                                [         ]
(d67)                           [ 0  1  0 ]
                                [         ]
                                [ 0  0  1 ]

(c68) C:ENTERMATRIX(2,2);       / Annat sätt att läsa in en matris.  /

What kind of matrix?
   1.  Diagonal        2.  Symmetric        3.  Antisymmetric    4.  General
Selection:  1;
Row 1 Column 1:    4;
Row 2 Column 2:    8;
Matrix entered.

                                  [ 4  0 ]
(d68)                             [      ]
                                  [ 0  8 ]

(c69) IDENT(3);                 / Bildar en 3x3 enhetsmatris.  /
                                [ 1  0  0 ]
                                [         ]
(d69)                           [ 0  1  0 ]
                                [         ]
                                [ 0  0  1 ]

(c70) DIAGMATRIX(2,7);          / Bildar en diagonalmatris.  /
                                  [ 7  0 ]
(d70)                             [      ]
                                  [ 0  7 ]

(c71) ZEROMATRIX(3,2);          / Bildar en 3x2 nollmatris.  /
                                  [ 0  0 ]
                                  [      ]
(d71)                             [ 0  0 ]
                                  [      ]
                                  [ 0  0 ]
(c72) A;
                                [ 1  2  3 ]
                                [         ]
(d72)                           [ 2  0  5 ]
                                [         ]
                                [ 9  8  7 ]

(c73) ROW(A,2);                  / Plockar ut rad 2 ur A. /
(d73)                           [ 2  0  5 ]
```

```
(c74) ADDROW(A,D73);              / Lägger till en rad till A, (sist).  /
                                   [ 1  2  3 ]
                                   [         ]
                                   [ 2  0  5 ]
(d74)                              [         ]
                                   [ 9  8  7 ]
                                   [         ]
                                   [ 2  0  5 ]

(c75) ADDCOL(D74,[1,1,1,1]);      / Lägger till en kolonn till A. /
                                  [ 1  2  3  1 ]
                                  [            ]
                                  [ 2  0  5  1 ]
(d75)                             [            ]
                                  [ 9  8  7  1 ]
                                  [            ]
                                  [ 2  0  5  1 ]

(c76) SUBMATRIX(1,2,D75,3,4);     / Plockar ut en delmatris, stryker raderna
                                     1 och 2 samt kolonnerna 3 och 4.  /
                                     [ 9  8 ]
(d76)                                [      ]
                                     [ 2  0 ]

(c77) RANK(D75);                  / Bestämmer rangen av en matris.  /
(d77)                                  3

(c82) A:MATRIX([1,2],[3,4]);
                                     [ 1  2 ]
(d82)                                [      ]
                                     [ 3  4 ]

(c83) EIGENVALUES(%);   / Bestämmer egenvärden och multiplicitet.  /

                        SQRT(33) - 5  SQRT(33) + 5
(d83)                [[- ------------, ------------], [1, 1]]
                               2             2

(c84) EIGENVECTORS(A);  / Bestämmer egenvärden och deras multipli-
                          citet och korresponderande egenvektorer.  /

           SQRT(33) - 5  SQRT(33) + 5                    SQRT(33) - 3
(d84) [[[- ------------, ------------], [1, 1],  [1, - ------------],
                2             2                             4
                                                     SQRT(33) + 3
                                                 [1, ------------]]
                                                          4
```

3.8. Polynom och ekvationssystem

Här presenteras kommandon för exakt eller approximativ lösning av rötter till polynom, samt lösning av linjära och icke-linjära ekvationssystem. Kommandona är **linsolve**, **solve** och **allroots**.

```
(c95) P:X+2*Y-3*Z = 5$                  / Bildar en ekvation, $ Upphäver
                                          utskriften.  /
(c96) Q:3*X-2*Y+Z = 6;

(d96)                          Z - 2 Y + 3 X = 6
(c97) R:-X+Y+Z = T;

(d97)                            Z + Y - X = T

(c98) LINSOLVE([P,Q,R],[X,Y,Z]);  / Löser ett linjärt ekvationssystem.  /

                        4 T + 45      5 T + 16      8 T + 13
(d98)              [X = --------, Y = --------, Z = --------]
                          14            7             14
(c99) P(X):=(X-1)*(X-2)*(X^2+1);
                                                    2
(d99)                   P(X) := (X - 1) (X - 2) (X  + 1)
(c100) P(X);
                                              2
(d100)                      (X - 2) (X - 1) (X  + 1)
(c101) EXPAND(%);

                            4      3      2
(d101)                     X  - 3 X  + 3 X  - 3 X + 2

(c102) SOLVE(%,X);  / Ger nollställena för polynom av grad <= 4.  /
(d102)                     [X = 1, X = 2, X = - %I, X = %I]

(c114) X^5+7*X^4-3*X^3-X^2+5*X-10;

                          5      4      3    2
(d114)                   X  + 7 X  - 3 X  - X  + 5 X - 10
(c115) SOLVE(%);         / Solve kan inte lösa alla 5:e gradsekvationer.  /

                            5      4      3    2
(d115)                [0 = X  + 7 X  - 3 X  - X  + 5 X - 10]

(c116) ALLROOTS(D114);/ Allroots ger numeriska närmevärden för rötterna.  /
(d116) [X = 1.006785 %I + 0.271404, X = 0.271404 - 1.006785 %I,
        X = 2.938736e-39 %I - 1.205225, X = 2.938736e-39 %I + 1.035067,
        X = 2.938736e-39 %I - 7.372651]
```

```
(c117) X^2+2*Y = 2;
                                        2
(d117)                             2 Y + X  = 2
(c118) 2*X+4*Y = 3;
                                  4 Y + 2 X = 3
(c119) SOLVE([D117,D118],[X,Y]); / Solve kan lösa vissa icke-linjära ekvations-
                                   system.  /
                SQRT(3) - 1      SQRT(3) + 2
(d119) [[X = - -----------, Y = -----------],
                     2                4

              SQRT(3) + 1        SQRT(3) - 2
         [X = -----------, Y = - -----------]]
                    2                  4
```

3.9. Differentialekvationer

En mängd rutiner för exakt eller numerisk lösning av differentialekvationer och ekvationssystem av differentialekvationer finns implementerade i MACSYMA. Här visas **ode2**, som löser andra ordningens differentialekvationer.

```
(c54) DIFF(Y(X),X,2)-3*DIFF(Y(X),X)+2*Y(X) = SIN(X)+COS(X);
    / Bildar en 2:a ordningens linjär diff.  ekvation med konstanta koeff.  /
              2
             d                d
(d54)        --- (Y(X)) - 3 (-- (Y(X))) + 2 Y(X) = SIN(X) + COS(X)
               2              dX
             dX

(c55) ODE2(%,Y(X),X);    / Löser ekvationen.  /
                         SIN(X) - 2 COS(X)         2 X         X
(d55)           Y(X) = - ----------------- + %K1 %E    + %K2 %E
                                  5

(c56) YP;     / Ger den partikulära lösningen som ingår i svaret.  /
                                SIN(X) - 2 COS(X)
(d56)                         - -----------------
                                        5

(c57) IC2(D55,X = 0,Y(X) = 1,DIFF(Y(X),X) = 2);
      / Löser begynnelsevärdesproblemet (y(0)=1,y'(0)=2).  /

                                                    2 X
               SIN(X) - 2 COS(X)   (5 Y(0) - 13) %E                      X
(d57) Y(X) = - ----------------- - ------------------- + (2 Y(0) - 3) %E
                     5                    5
```

3.10. Programmering

Med **if then else** konstruktioner och relationsoperatorerna $<, <=, =, >, >=, \sharp$, samt logiska operatorena **NOT**, **AND**, **OR**, kan man definiera funktioner styckevis.

```
    / Vi definierar signum funktionen.  /

(c2) SIGNUM(X):=IF X > 0 THEN 1 ELSE (IF X < 0 THEN -1 ELSE 0);
(d2)     ''SIGNUM(X) := IF X > 0 THEN 1 ELSE (IF X < 0 THEN - 1 ELSE 0)
(c3) SIGNUM(3.8);
(d3)                                  1
(c4) SIGNUM(-2.89);
(d4)                                 - 1
(c5) SIGNUM(0);
(d5)                                  0
```

Man kan definiera funktioner rekursivt.

```
(c6) FAKULTET(N):= IF N=0 THEN 1 ELSE N*FAKULTET(N-1);
(d6)         FAKULTET(N) := IF N = 0 THEN 1 ELSE N FAKULTET(N - 1)
(c7) FAKULTET(0);
(d7)                                  1
```

Vi använder loop konstruktionen **for** i:x0 **thru** x1 **do** (satser).

```
    / Utskrift av fakulteterna från 1 till n med ldisplay.  /

(c8) FAKULTETER(N):=FOR I:1 THRU N DO LDISPLAY(FAKULTET(I))$
(c9) FAKULTETER(4);
(e9)                           FAKULTET(1) = 1
(e10)                          FAKULTET(2) = 2
(e11)                          FAKULTET(3) = 6
(e12)                          FAKULTET(4) = 24
(d12)                                DONE
```

Block kommandot används för att definiera ett block med valbart antal satser. Den allmänna formen för ett block är **block([var1,var2,...], sats1,sats2,...)**. Variabler som förekommer i början av block definitionen gäller lokalt i blocket, och kan tilldelas värden.

```
   / Definierar ett program för fakultetberäkning.  /

(c13) FAKULT():=BLOCK([S:1],

                      N:READ("ge in ett tal, avsluta med return"),
                      FOR I THRU N DO S:S*I,
                      PRINT("fakulteten av talet är ",S))$

(c14) FAKULT()$
ge in ett tal, avsluta med return
7;
fakulteten av talet är  5040
```

3.11. Körning på en SPARC-arbetsstation

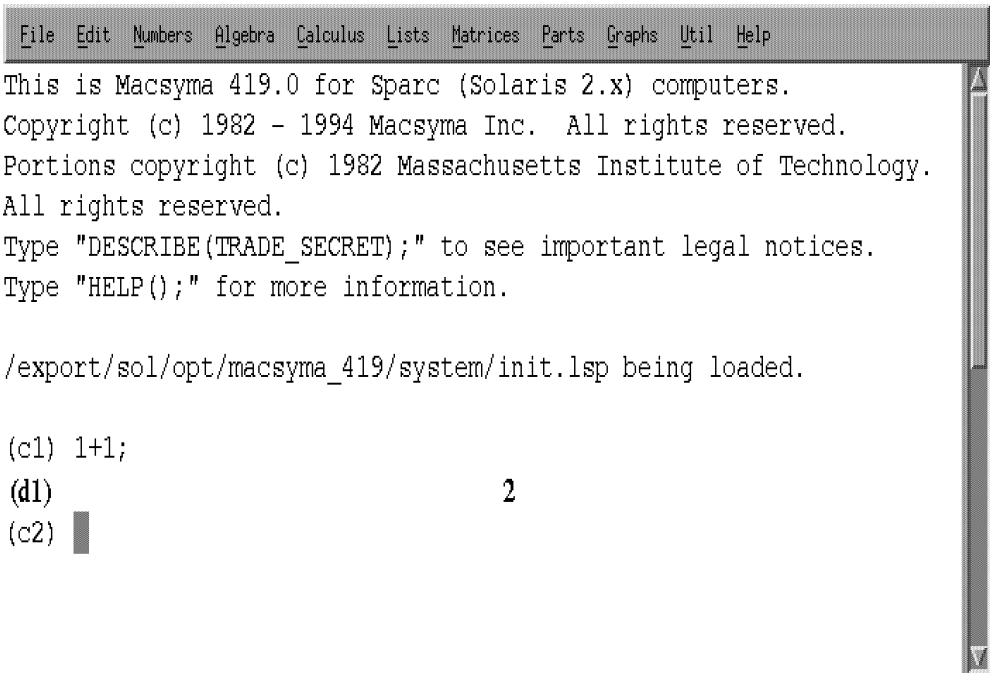
Efter inloggning på arbetsstationen startar man OpenWindows med kommandot: **openwin**. Från det öppnade arbetsbordet kan man ge in kommandon från **cmdtool** (CONSOLE) -fönstret. Alternativt kan man välja att arbeta från ett **shelltool** -fönster, detta öppnas genom att hålla högra musknappen nedtryckt (på en fri area) och gå in på **programs** i menyn och därifrån välja **shelltool**.

Därefter loggar man in på Aton med kommandot: **rlogin aton** , varefter utskriften styrs till ifrågavarande arbetsstation med kommandot:

setenv DISPLAY *name*:**0**

Notera att i kommandot ovan sätts det riktiga namnet på arbetsstationen in (istället för *name*). Namnet framgår ofta från skärmen el. datorn, t.ex. i Asa finns arbetsstationerna **tor** och **qed** emedan i DataCity finns bl.a. kompositörerna BACH, CHOPIN, MOZART och WAGNER. Slutligen startas Macsyman med kommandot: **macsyma**

Ett "notebook-likt" fönster öppnas efter kommandona ovan, se figuren nedan. Detta innehåller menyer varifrån funktioner bekvämt kan användas eller studeras, kommandon ges alltså in här. Vid val av information från **Help**-menyn öppnas ett sk. *front-end* fönster varifrån Macsyma visar sina hjälptexter.



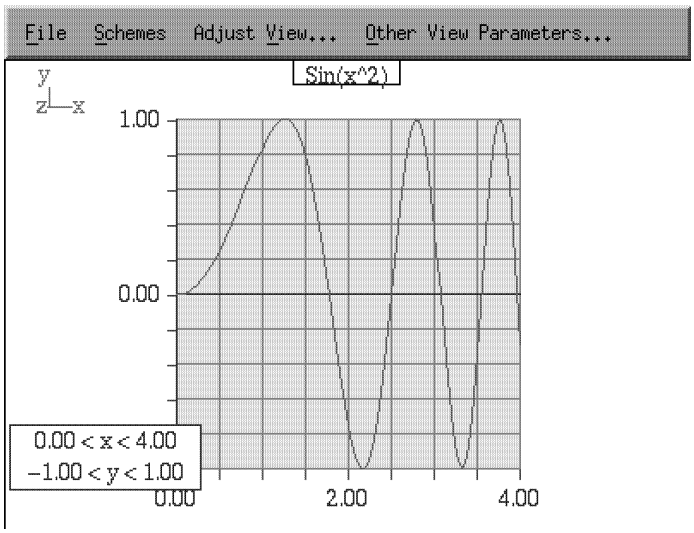
Figur 3.1. Kommandofönstret i X-Win omgivning.

3.12. Grafik

Macsyma har (nuförtiden) även ett stort antal funktioner för grafisk visualisering, både i 2 och 3 dimensioner. På Aton kan dessa egenskaper utnyttjas ifall man arbetar vid en *SPARC*-station (se kap. 3.11.) el. från en PC med *X-WIN* program. Kommandon för plottning i 2 dimensioner:

plot(f(x),x,xmin,xmax); – ritar ut grafen av $f(x)$ då $x \in [xmin, xmax]$.
contourplot(f(x,y),x,xmin,xmax,y,ymin,ymax); – ritar ut nivåkurorna av funktionen $f(x,y)$ då $x \in [xmin, xmax]$ och $y \in [ymin, ymax]$.
implicitplot(ekv,x,xmin,xmax,y,ymin,ymax); – plottar implicit givna funktioner.
paramplot([x(t),y(t)],t,tmin,tmax); – plottning av en parameterkurva i 2-dimensioner.
graph(x_list,y_list); – plottar datapunkterna i y_list som funktion av x_list.

```
(c1) plot(sin(x^2),x,0,5,false,false,''sin(x^2)'');
(d1)                          done
```

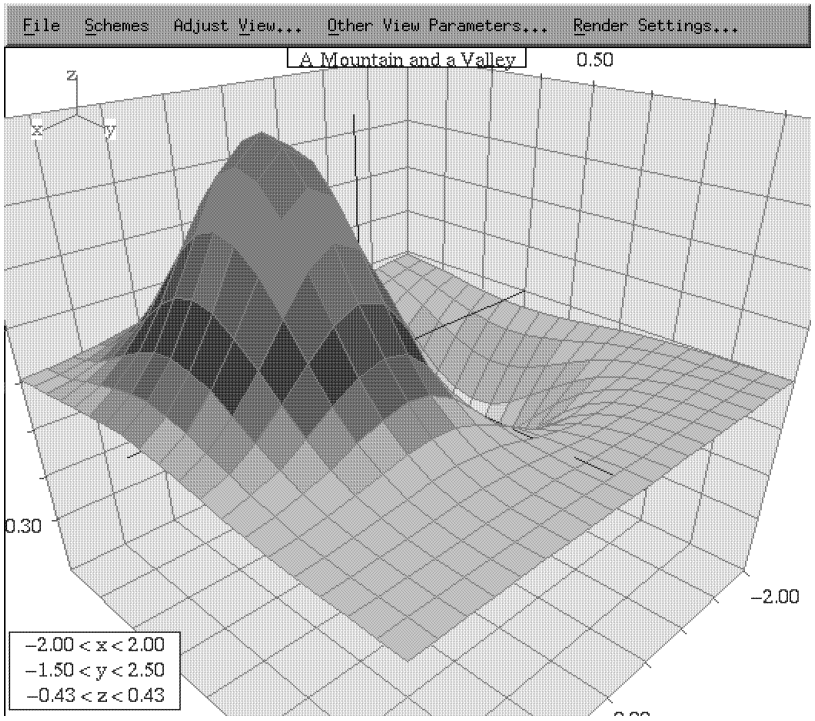


Figur 3.2. Plottning av $sin(x^2)$, $x \in [0,5]$.

Kommandon för 3-dim. plottning:

plot3d(f(x,y),x,xmin,xmax,y,ymin,ymax); – plottning av en 2-dim. funktion.
plot_data(M); – plottning av elementena i matrisen M som funktion av matrisindexen.
plotsurf([x(s,t),y(s,t),z(s,t)],s,smin,smax,tmin,tmax); – plottning av 2-dim. yta i 3 dimensioner.
contourplot3d(f(x,y),x,xmin,xmax,y,ymin,ymax); – nivåkurvan av $f(x,y)$.
paramplot3d([x(t),y(t),z(t)],t,tmin,tmax); – parameterkurva i 3 dim.
graph3d(x_list,y_list,z_list); – plottar listan z:s datapunkter gentemot x_list och y_list.

```
(c2) plot3d(exp(-x^2-y^2)*x,x,-2,2,y,-1.5,2.5,false,false,
                          ''A Mountain and a Valley'');
(d2)    done
```



Figur 3.3. Plottning av $e^{-x^2-y^2}x$, för $(x,y) \in ([-2,2],[-1.5,2.5])$.

3.13. Utskrift

Vid plottning av figurer kan dessa skrivas ut till default printern genom att välja alternativet **Print** från **File**-menyn. Macsyma sänder då en "dump" av ifrågavarande figur och sänder detta i PostScript-form till printern samt till en temporär fil, vars namn skrivs ut i kommandofönstret, till /**tmp** området på Aton. Macsyma skriver ut i kommandofönstret (**cmdtool** el. **shelltool**) namnet på den temporära filen, varefter denna kan kopieras/printas från /**tmp** området. Observera att vid printning skrivs det aktiva fönstret ut, m.a.o. man bör aktivera ifrågavarande plott-fönster genom att klicka på detta (annars kan fel fönster skrivas ut).

Referenser:

[1] SYMBOLICS Inc.: MACSYMA reference manual, 1984.
[2] Hohti, A.: An Introduction to Vaxima, 1987.
[3] Ruskeepää, H.: Johdatusta MACSYMA-ohjelmiston käyttöön, 1987.

Övningsuppgifter

Endel nya kommandon som inte presenterats tidigare inlärs via övningarna.

1. Använd kommandona **trigreduce** och **trigexpand** för att omforma trigonometriska uttryck med enkla argument till uttryck med multipla argument, respektive uttryck med multipla argument till uttryck bestående av enkla argument. Testa t. ex. **trigreduce** på uttrycket $1+3\sin(x)-2\sin^2(x)-4\sin^3(x)$, och använd det tidigare behandlade **trigsimp** för att förenkla mellanresultat. Tillämpa **trigexpand** på uttrycket $\tan(3x)$. Märk att MACSYMA inte ersätter $\tan(x)$ med $\frac{\sin(x)}{\cos(x)}$. Med kommandot **tellsimp** kan vi ge in en egen förenklingsregel i MACSYMA. Syntaxen är **tellsimp(tan(x),sin(x) /cos(x));**. Efter detta kan vi vidare förenkla kvoten av tangensfunktioner med kommandot **ev**.
2. Definiera t. ex. en kvot av två polynom. Träna på att plocka ut delar av ett uttryck med hjälp av kommandona **first, last** och **part**. Använd **describe** kommandot för att få en beskrivning av hur kommandona fungerar. Med kommandona **subst** och **substpart** kan man ersätta delar av ett uttryck med nya deluttryck. Kolla detta med hjälp av **describe** kommandot.
3. Med kommandot **interpolate** kan man numeriskt söka ett nollställe för en funktion $f(x)$ i intervallet [lo, hi]. Funktionen bör ha olika tecken i intervalländpunkterna. Syntaxen är **interpolate(f(x),x,lo,hi);**. Testa med några funktioner.
4. Definiera en rekursiv funktion $f(n)$ som beräknar det n:te Fibonacci talet. Fibonacci talen definieras genom differensekvationen $a_0 = 1,\ a_1 = 1,\ a_n = a_{n-1} + a_{n-2},\ n \geq 2$. Använd kommandot **makelist(f(i),i,0,9)** för att skriva ut de 10 första Fibonacci talen.
5. Testa kommandona **diff, integrate** och **romberg** på några funktioner.