LUNDS
UNIVERSITET

Institutionen för elektrovetenskap

# Studiehäfte MATLAB för E1

MATLAB är en förkortning av Matrix Laboratory. Det är ett programpaket som ursprungligen utvecklades för att användas inom undervisning. Med tiden har paketet byggts ut och blivit det idag främsta mjukvarupaketet inom matematik, teknik och fysik för universitet och företag. Som student kommer du att träffa på MATLAB i ett antal kurser och det är viktigt att du i ett tidigt skede får lära dig grunderna. Syftet med denna introduktion är att du skall komma över den första tröskeln och kunna utföra enkla beräkningar, rita kurvor och skriva enkla program.

Du skaffar dig mer information genom att använda MATLABs inbyggda hjälpfunktioner eller genom litteratur såsom *Användarhandledning för Matlab 6* av Pärt-Enander och Sjöberg eller *MATLAB Primer* av Kermit Sigmon.

## MATLAB på tre nivåer

MATLAB används normalt på följande tre nivåer:

1. Avancerad kalkylator
2. Programmering
3. Toolboxes

Vi ger grunderna för de två första nivåerna och tillämpar dem på exempel tagna från kretsteorin. Toolboxar är paket av MATLAB-program som satts samman för att lösa vissa klasser av problem. De har användargränssnitt som är mycket användarvänliga, men är inriktade mot specifika teknikområden och kräver att man har ganska stora kunskaper inom dessa område. Det gör att det inte är meningsfullt att ägna tid åt toolboxar i denna introduktion. Om du vill få information om vilka toolboxar som finns tillgängliga och hur man använder dem kan du gå in under MATLAB help i menyn Help. Längst ner i MATLAB Help ligger toolboxarna. Du kommer att stöta på en del av MATLABs toolboxar senare i din utbildning.

## Starta MATLAB

På Unix-datorer startar du MATLAB genom att skriva matlab. Om du använder PC eller Mac startar du som vanligt genom att klicka eller markera matlabikonen. När du startar MATLAB hamnar du i det som kallas Command window. Här kan du utnyttja MATLAB som en avancerad kalkylator.

## Hjälp

Det finns flera hjälpfunktioner i MATLAB. Dels kan du använda hjälpfunktionen i menyn och dels kan du be om hjälp i Command window.

- Markera MATLAB Help under menyn Help. Där finner du information om allt i MATLAB. Det inte meningsfullt att läsa igenom alla avsnitten direkt utan det är bättre att efterhand utnyttja hjälpen när du behöver den. I vissa fall är det enklare att läsa i en lärobok för att få information.

- Antag att du vill ha reda på hur man ritar kurvor i MATLAB. Skriv då
  $>>$ help plot
  MATLAB svarar med en utförlig beskrivning av plot-rutinen. Den ger dessutom sökord för andra rutiner som har att göra med grafer och plottar.

- Om du skriver
  $>>$ help :
  svarar MATLAB med en kort beskrivning av de vanligaste kommandona.

- Skriver du
  $>>$ help help
  får du mer information om hjälpfunktionen.

## MATLAB som räknedosa

Det som skrivits efter %-tecknet nedan är förklarande kommentarer.
**Exempel:** Vad är $234 \cdot 27$? Skriv in 234*27 och slå return. Då får du:
$>>$ `234*27`

```
ans = 6318                    % MATLABs svar
```

**Exempel:** Antag att du parallellkopplar resistansen $R_1 = 25\ \Omega$ med resistansen $R_2 = 52\ \Omega$. Bestäm totala resistansen.

Alternativ 1:

```
>> 25*52/(25+52)

ans = 16.8831
```

Alternativ 2:

```
>> R1=25

R1 = 25

>> R2=52

R2 = 52

>> R=R1*R2/(R1+R2)

R = 16.8831
```

Alternativ 3:

```
>> R1=25;                % raden avslutas med semikolon
>> R2=52;
>> R=R1*R2/(R1+R2)

R = 16.8831
```

Genom att avsluta kommandot med ett semikolon (;) som i alternativ 3, så skriver inte MATLAB ut resultatet. På det sättet undviker du att fylla skärmen med onödiga siffror.

**Exempel:** Vad är $\sqrt{\sin^2(\pi/4) - \tan^3(\pi/7)}e^{-3+\pi}$? Detta skriver man på följande sätt:

```
>> sqrt(sin(pi/4) ^ 2-tan(pi/7) ^ 3)*exp(-3+pi)

ans = 0.7179
```

### Uppgift 1

Bestäm $\log_{10}(2)$ och $\ln(3)$ (använd help log om du inte vet)

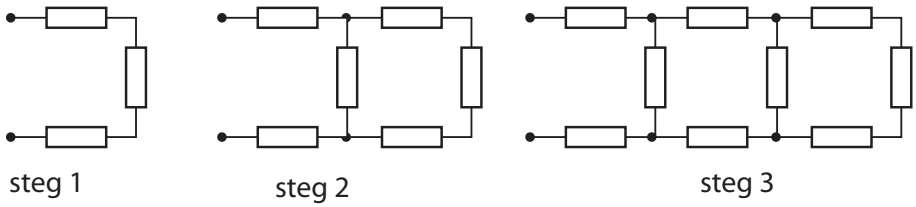
Svar: $\log_{10}(2) = 0.301$ och $\ln(3) = 1.0986$

## Uppgift 2

Lös uppgift 2.12 i exempelsamlingen på följande sätt: Låt varje resistans ha resistansen $R = 1\,\Omega$.

a) Bestäm med MATLAB ett numeriskt värde på totala resistansen mellan $A$ och $B$. Om du inte hinner lösa uppgiften kan du titta i facit.

b) Du skall här iterera fram lösningen. Antag först att stegen endast består av ett steg med tre seriekopplade resistanser. Bygg på stegen genom att koppla in nästa steg av tre resistanser (förslagsvis från vänster). Vad blir resistansen? Koppla in ytterligare ett steg på ett systematiskt sätt och kontrollera resistansen. Vid varje nytt steg kan du jämföra med det exakta värdet på stegens resistans. Hur många steg behövs för att få alla siffror rätt?




c) Ändra antalet siffror som MATLAB visar genom att skriva `format long`. Hur många steg krävs för att få alla siffror rätt? Notera att du samtidigt lyckats numeriskt bestämma $\sqrt{3}$ ganska snabbt genom att använda de fyra räknesätten.

## Praktiska tips

Om du har skrivit ett kommando och skickat iväg det med return så kan du alltid få fram samma kommande genom att trycka på tangenten med pil upp. Om du vet att kommandot började med t.ex. R så kan du skriv R och sedan pil upp tangenten. Då kan du gå tillbaks till det kommando du vill ha.

Om du varit inne i MATLAB och gjort ett antal saker och sedan stänger av MATLAB kan det vara värdefullt att återskapa det du gjort. Gå upp i menyn och klicka på View och Command history. Där finns dina kommandon sparade och du kan få fram dem genom att dubbelklicka.

Det går att radera alla variabler som man skapat i Command window genom att skriva `clear`. Vill du bara radera variabeln p så skriver du `clear p`. Om du vill ha reda på vilka variabler du skapat kan du skriva `who`.

## Vektorer

Vektorer används flitigt i all slags programmering. En vektor är en rad eller kolonn av tal eller ord. (i en rad är talen uppradade horisontellt, i en kolonn är de uppradade vertikalt)

**Exempel:** Skriv

```
>> f(1)=1;
>> f(2)=3;
>> f(3)=5;
>> f(4)=7;
```

f är då en radvektor med fyra element. Om du skriver

```
>> f
```

så svarar MATLAB med

```
f=1 3 5 7
```

**Exempel:** Man kan bilda f på ett enklare sätt genom att skriva

```
>> f=[1 3 5 7];
```

Eftersom elementen i f skiljer sig åt med 2 så kan f också bildas på följande sätt:

```
>> f=[1:2:7];
```

där i [n1:n2:n3] n1 står för startvärde, n2 för inkrement och n3 för slutvärde. Om inte n2 skrivs ut så blir inkrementet automatiskt 1.

**Exempel:** Skapa en radvektor g som innehåller $\sin(x)$ för $x = \pi/10, 2\pi/10, \ldots \pi$

```
>> x=(1:10)*pi/10;
```

Nu har vektorn $x$ skapats och det är enkelt att skapa g

```
>> g=sin(x);
```

Kommentar: Vi behövde egentligen inte skapa $x$ först utan det hade räckt med

```
>> g=sin((1:10)*pi/10);
```

**Exempel:** Skapa radvektorerna e= $x\sin(x)$, f=$\cos(x)/x$, g=$e^x \sin(x) + cos(x)$, h=$x^3 + x^2 + x$ och k=$g \cdot h$ för $x = \pi/10, 2\pi/10, \ldots \pi$. Vi skapar först vektorn $x$ på samma sätt som ovan:

```
>> x=(1:10)*pi/10;
```

För att bilda vektorn e måste vi multiplicera vektorn $x$ med vektorn $\sin(x)$ så att

varje element i $x$ multipliceras med motsvarande element i $\sin(x)$. Man gör detta genom att använda en punkt.

```
>> e=x.*sin(x);
```

Operationen .* betyder alltså att varje element i vektorn till vänster multipliceras med motsvarande element i den högra vektorn. De andra vektorerna skapas på liknande sätt

```
>> f=sin(x)./x;     % operationen ./ ger elementvis division
>> g=exp(x).*sin(x)+cos(x);     % man behöver inte använda . vid + eller -.
>> h=x.^ 3+x.^ 2+x;     % operationen .^ ger att varje element i x upphöjs till 3.
>> k=g.*h
```

**Exempel:** Skapa en kolonnvektor f med elementen 1, 3, 5, 7, 11.

```
>> f=[1; 3; 5; 7; 11];
```

Om du låter MATLAB skriva ut f så ser du att elementen är staplade vertikalt.

### Uppgift 3

En elvisp kopplas in på ett vanligt eluttag, dvs den matas med den sinusformade spänningen $v(t) = V_0 \sin \omega t$ där $V_0 = 325$ V (motsvarar 230 V effektivvärde), $\omega = 2\pi f$ och $f = 50$ Hz. När man mäter upp strömmen som går in i elvispen finner man att den ges av $i(t) = I_0 \sin(\omega t - 0.12\pi)$ där $I_0 = 3$ A. Bilda en vektor p som innehåller effekten $p(t) = v(t)i(t)$ vid varje millisekund i intervallet $t = 0$ till $t = 20$ ms. Låt MATLAB skriva ut värdena.

## Matriser

En $3 \times 2$ matris

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix}$$

skrivs i Matlab som

```
>> A=[1 2;3 4;4 5];
```

Multiplikation av en $m \times n$ matris $A$ med en $n \times p$ $B$ ger en matris $C = AB$. Detta skrivs

```
>> C=A*B;
```

Antag att vi vill lösa ekvationssystemet $Bx = y$ där $B$ är en $3 \times 3$ matris

$$B = \begin{pmatrix} 1 & 2 & 1 \\ 3 & 4 & 3 \\ -1 & 2 & 1 \end{pmatrix}$$

och $y$ är en kolonnvektor

$$y = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

Vi löser detta genom att skriva

```
>> B=[1 2 1;3 4 3; -1 2 1];
>> y=[1;2;3];
>> x=B^-1*y
```

vilket ger x=(-1.0,0.5,1.0). Det går att skriva detta på några fler sätt. T.ex.

```
>> x=inv(B)*y
>> x=B\y
```

Det sista sättet att skriva är lite mer allmänt eftersom backslash kan användas på ekvationssystem som inte kvadratiska. Läs mer om detta under help slash.

Transponatet av en matris A får man om man skriver

```
>> transp(A);
```

eller `>> A.';`
Skriver man `>> A';`
får man hermitkonjugatet av matrisen, dvs komplexkonjugatet av den transponerade matrisen. Är A en reell matris är alltså transp(A), A' och A.' samma sak.

Kommentar: Om man har två $m\times$ matriser A och B och skriver C=A*B får man vanlig matrismultiplikation vilket ger en $m \times m$-matris C. Om vi däremot skriver C=A.*B får vi en $m \times m$-matris där element $c_{pq}$ (elementet i rad $p$ och kolonn $q$ av C) ges av $a_{pq}b_{pq}$, dvs det ger elementvis multiplikation. Det finns lång rad andra matrismanipulationer man kan göra i Matlab men vi nöjer oss med detta.

## Kurvor

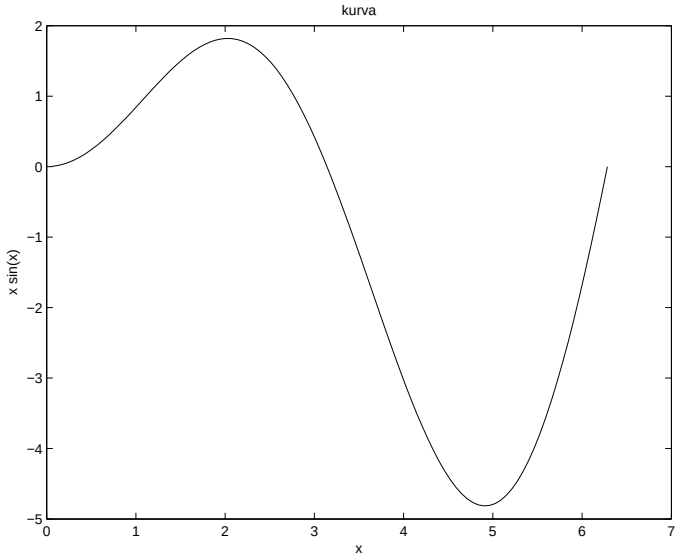
MATLAB erbjuder fantastiska möjligheter att grafiskt visa resultat. Vi går igenom hur man ritar en kurva av en funktion f(x) som funktion av x. Man låter då $x$ anta diskreta värden, dvs $x$ blir en vektor, och f(x) en vektor med motsvarande funktionsvärden. Gör `help plot` så får du den information du behöver om plotfuktionen. Längst ner i `help plot` står det `See also SEMILOGX, SEMILOGY, LOGLOG, PLOTYY, GRID, CLF, CLC, TITLE, XLABEL, YLABEL, AXIS, AXES, HOLD, COLORDEF, LEGEND, SUBPLOT, STEM`. Du kan då göra t.ex. `help xlabel` (använd små bokstäver) för att få information om hur man sätter text på x-axeln.

**Exempel:** Rita upp funktionen $x\sin(x)$ i intervallet $0 \leq x \leq 2\pi$. Låt axlarna ha texten $x\sin(x)$ respektive $x$ och figuren ha rubriken **kurva**.

```
>> x=(0:100)*2*pi/100;

>> plot(x,x.*sin(x))
>> xlabel('x')
>> ylabel('xsin(x)')
>> title('kurva')
```

I figurfönstret visas då följande kurva:




### Praktiskt tips

Som synes är siffrorna små. Det kan man ändra genom att klicka på `Edit` i figurfönstret. Genom att gå in i `Figure properties` eller `Axes properties` kan man fixa till figuren. Det går också att skriva in titel och text på axlarna genom att klicka i figurfönstret.

### Uppgift 4

Låt $v(t)$, $i(t)$ och $p(t)$ vara givna av uppgift 3. Rita in de tre kurvorna för $v(t)/V_0$, $i(t)/I_0$ och $p(t)/(I_0V_0)$ i samma figur. Den första kurvan skall vara heldragen, nästa streckad och den sista prickad. (Gör help plot, help hold så får du hjälp).

### Uppgift 5

Använd subplot (gör help subplot) och rita upp de tre kurvorna för $v(t)$, $i(t)$ och $p(t)$ i tre figurer som ligger parallellt med varandra. Skriv ut lämpliga texter på axlarna och ge figurerna lämpliga titlar.

## M-filer

Man kan spara kommandon i MATLAB och även skriva program. För dessa ändamål använder man M-filer. Du skapar en M-fil genom att trycka på `New M-file` under menyn `File`. Om du vill spara kommandon så skriver du in dem i M-filen och gör save. Om du trycker på F5 så körs scriptet. Du kan också köra det från i Command window genom att skriva filnamnet.

### Exempel

En M-fil som består av raderna

```
a=2;
b=4;
c=a*b
```

ger resultatet

```
>> c=8
```

Om man i första raden skriver clear så kan inte gamla värden på variabler användas. Om man glömt ge värdet till en variabel i programmet så finns en risk att man ändå kan köra det genom att variabeln tidigare fått ett värde. Det undviker man om man börjar med clear.

### Uppgift 6

Låt MATLAB rita upp en julgran. Du får fria händer hur du skall gå tillväga. Spara kommandona i en M-fil.

## Att skriva program i MATLAB

Hittills har vi arbetat i Command window och använt MATLAB som en avancerad räknedosa. Vi skall nu se hur man skriver program i MATLAB. För att göra detta öppnar vi ett nytt fönster genom att trycka på `New M-file` under menyn `File`. Vi börjar med att skriva ett enkelt program som räknar ut resistansen för två parallellkopplade motstånd. Programmet ser ut på följande sätt:

```
function rp=rparallell(r1,r2)
rp=r1*r2/(r1+r2);
```

function talar om att det är ett program, rp är utdata och r1,r2 är indata. Du kan spara programmet genom att göra Save. MATLAB föreslår då att det sparas som filen rparallell.m, vilket är lämpligt att acceptera. Låt oss säga att du vill ha resistansen r för två parallellkopplade resistanser r1=2.1 Ω och r2=3.1 Ω. I command window skriver du då

>> `r=rparallell(2.1,3.1)`

MATLAB svarar då med

`r=1.2519`.

*Kommentar:* Om du fick felmeddelandet ??? Undefined function or variable 'rparallell' så betyder det att MATLAB inte hittat filen rparallell.m. Du måste då tala om i vilken katalog (eller mapp) MATLAB skall leta. Detta gör du genom att gå in i `Set path` under `File`. Lägg till katalogen och glöm inte göra `Save path`.

**Uppgift 7**

Skriv ett pedagogiskt program för enkel kretsteori. Programmet skall ställa frågor och användaren skall svara genom att skriva in värden. Värdena skall läsas in av programmet som sedan meddelar om det angivna svaret var rätt eller fel. Använd dig av en snurra så att programmet forsätter att ställa frågor tills dess användaren säger ifrån. Du får själv välja hur avancerat programmet skall vara.

Ledning: Gör `help input` så får du information om hur programmet blir interaktivt. För att göra en snurra använder du dig antingen av en for slinga eller av en while sats. Gör `help for`, `help if` och `help while` så finner du tillräckligt med information för att göra ett enklare program. Om du vill att Matlab själv skall välja värden kan du använda dig av slumptalsgenerator. Gör `help rand` och `help fix` för information.

## Matriser och lösning av ekvationssystem

Vid nodanalys av en krets blir resultatet ett ekvationssystem med $N-1$ ekvationer, där $N$ =antalet väsentliga noder i kretsen. Vi har i kursen sett att man brukar skriva detta ekvationssystem på matrisform. Anledningen är att man då förbereder för en numerisk lösning av nodpotentialerna. Denna numeriska lösning görs med fördel i MATLAB.

**Exempel:** Lös tal 4.7 i exempelsamlingen, eller studera lösningen till detta tal. I lösningen ser vi att nodpotentialerna ges av ett ekvationssystem för de tre obekanta nodpotentialern $v_1$, $v_2$ och $v_3$.

**Exempel:** Följande korta MATLAB-program löser ekvationssystemet:

```
function v=nodanalys(R1,R2,R3,R4,R5,h,is,vs)
% Detta program löser ekvationssystemet för tal 4.7
% i exempelsamlingen
A=[1/R1+1/R4  -(h/R2+1/R4)  0;-1/R4  1/R2+1/R4+1/R5...
-1/R5;0  -1/R5  1/R3+1/R5];
hl=[0;0;-is+vs/R3];
v=A\hl;
```

Matrisen A skapas alltså genom att sätta ett semikolon mellan varje rad. Elementen i varje rad separeras med ett blanktecken. Högerledet är en kolonnvektor vilket motsvarar en matris med tre rader och en kolonn. Man kan se operationen som en division mellan vektorn i högerledet och matrisen A. Det går även att skriva detta som $A^{-1} * hl$ där $A^{-1}$ kallas för inversen av A. I kursen Linjär Algebra, som går i lp 1 på vårterminen, kommer du att lära dig mer om vektorer och matriser. Du får där se hur man matematiskt utför "divisionen" med A.

### Praktiska tips

De tre punkterna ... markerar att man fortsätter på nästa rad. Det är lämpligt att använda om uttrycken är långa. En kort dokumentation är inlagd i programmet. Du skriver den genom att börja med ett procenttecken. MATLAB bryr sig inte om vad som kommer efter procentteckenet.

### Uppgift 8

Skriv ett program som löser tal 4.8 numeriskt. Sätt in $v_s = 1$ V, $i_s = 1$ mA, $R_1 = 1$ kΩ, $R_2 = 2$ kΩ, $R_3 = 3$ kΩ, $R_4 = 4$ kΩ, $R_5 = 5$ kΩ, $R_6 = 6$ kΩ, $R_7 = 7$ kΩ och $R_8 = 8$ kΩ. Dokumentera programmet genom att skriva att det löser tal 4.8.

### Uppgift 9

Skriv ett program som ritar upp kurvorna $v_{R_1}(t)$, $v_{R_1}(t)$, $v_L(t)$ och $v_C(t)$ för uppgift 6.6 i samma graf. Som indata ger du $v_0$, $R_1$, $R_2$, $L\,C$. Alla fyra kurvorna skall ha olika typer av linjer (heldragen, streckad osv). På axlarna skall du ange tid (s) respektive spänning (V). Använd ditt program för $v_0 = 10$ V och välj värden på $R_1$, $R_2$, $L$ och $C$ så att det inte går att skilja kurvan för $v_{R_1}(t)$ från den för $v_C(t)$. Programmet skall själv välja tidsintervallet till 2 gånger den maximala tidskonstanten. Dokumentera programmet så att du någorlunda väl beskriver vad det gör.

# Komplexa tal

MATLAB har inga problem med att använda komplexa tal. Den imaginära enheten kan vara antingen $i$, som i matematiken, eller $j$, som i elektroniken.

**Exempel:** Bestäm (1+j)/(1-j) i MATLAB

```
>> (1+j)/(1-j)

ans = 0 + 1.0000i
```

**Exempel:** Bestäm realdelen, imaginärdelen och absolutbeloppet av $(1 + e^{j\pi/8})(3 - j\cos(\pi/4))$.

```
>> g=(1+exp(j*pi/8))*(3-j*cos(pi/4));
>> real(g)

ans= 6.0422

>> imag(g)

ans= -0.2123

>> abs(g)

ans = 6.0460
```

### Uppgift 10

Låt MATLAB räkna ut $j^j - e^{-\pi/2}$. Försök förklara varför du får det svar du får.

### Uppgift 11

I många av de uppgifter som finns i exempelsamlingen ges svaret som en komplex spänning eller ström. Skriv ett MATLAB-program som ritar upp motsvarande tidsharmoniska spänning, eller ström för intervallet $[0, T]$ där $T = 1/f$ är periodtiden. Programmet skall använda realdelskonvention. Du skall ange Ström/A på den vertikala axeln om det är en ström som ritas upp och Spänning/V om det är en spänning. Som indata till programmet skall du ge frekvensen och det komplexa talet som motsvarar spänning eller ström. Grafen skall ha titeln Tidsplan. Använd ditt program för att rita upp kurvan för den tidsberoende strömmen eller spänningen då

a) $I = I_0(1 + e^{j \arctan(1/3)})$ där $I_0 = 2$ A och $f = 50$ Hz

b) $V = V_0(1 + 0.3j)/(1 - 2j)$ där $V_0 = 10$ V och $f = 50$ Hz

### Uppgift 12

Skriv ett program som ritar ut Bode-diagram för lågpass-, högpass-, bandpass- och bandstoppfilter. Programmet skall anropas på följande sätt:
>> myfilter(R,L,C,'typ')
där typ står för lågpass, högpass, bandstopp eller bandpass. Programmet skall generera ett Bode-diagram för amplituden (ej asymptotiskt) för det filter som an-

ropas. Frekvensintervallet skall vara $[\omega_b/100,\ 100\omega_b]$ där $\omega_b$ är brytvinkelfrekvensen. Filtren skall vara såpass enkla att de kan konstrueras av två eller tre komponenter (R,L,C). Du får själv välja hur hur skall konstrueras. Programmet får inte använda sig av den inbyggda rutinen bode. Axlarna i Bodeplotten skall ha texten Vinkelfrekvens (rad/s) respektive Amplitud (dB) och figuren skall ha titelnamn som talar om vilket filter det är. Programmet skall vara väldokumenterat.

*Ledning:* Här behöver du använda if-satser. Gör help if. Du behöver också använda textsträngar. En användbar konstruktion är `if a=='bandpass'`.