Løsning an lineære likninger med MATLAB

Løse n likninger med n ukjente

Vi tar utgangspunkt i følgende likningssett av tre likninger med tre ukjente:

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

Til informasjon er svaret på dette likningssettet at $x = 2$, $y = -2$ og $z = 3$, men det skal vi finne ut ved hjelp av MATLAB. Først og fremst, MATLAB er ikke forkortelse for «matematikklaboratorium», men «matriselaboratorium». Matriseregning er tema i MAT106, så denne beskrivelsen er kun en kokebokmetode for å løse denne typen likninger.

Likningene ovenfor kan på matriseform skrives på følgende måte:

$$\begin{bmatrix} 2 & 3 & -1 \\ 1 & 1 & -1 \\ 1 & -1 & -2 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -5 \\ -3 \\ -2 \end{bmatrix}$$

Altså at «koeffisientmatrise» · «variabelvektor» = «svarvektor». Tallene i koeffisientmatrisen er de samme som de respektive faktorene i likningssettet øverst. På generell form kan vi si at:

$$\mathbf{K} \cdot \mathbf{v} = \mathbf{s}$$

Man skulle tro at ved å dividere svarvektoren på koeffisientmatrisen så kan vi finne variabelmatrisen (altså x, y og z), men det er bare nesten sant. Vi må multiplisere svarmatrisen med den inverse koeffisientmatrisen.

$$\mathbf{v} = \mathbf{K}^{-1} \cdot \mathbf{s}$$

Metode 1, Invers matrise

Nedenfor vises koden for hvordan dette kan gjøres MATLAB. Det er brukt semikolon bak linjer som vi ikke ønsker at MATLAB skal gjenta. På den måten ser det litt bedre ut sånn rent stilistisk. Variablene er kalt «kmatr» og «svekt», men her står du fritt. Kall dem gjerne «pompel» og «pilt» om du vil… Den tredje variabelen blir kalt «kinvers».

Legg merke til at linjeskift markeres med et semikolon i en MATLAB matrise.

```
>> svekt = [-5 ; -3 ; -2];

>> kmatr = [2 3 -1 ; 1 1 -1 ; 1 -1 -2];

>> kinvers = inv(kmatr);

>> kinvers * svekt


ans =

    2.0000

   -2.0000

    3.0000
```

Svaret (ans) blir 2, -2 og 3, noe som var forventet for henholdsvis x, y og z.

Hvis vi har matrise og vektor på feil plass så nekter MATLAB å utføre operasjonen. Faktorenes orden er med andre ord IKKE likegyldig her.

```
>> svekt = [-5 ; -3 ; -2];

>> kmatr = [2 3 -1;1 1 -1; 1 -1 -2];

>> kinvers = inv(kmatr);

>> svekt * kinvers

Error using  *

Inner matrix dimensions must agree.
```

Metode 2, linsolve

En annen måte å løse dette på er å bruke «linsolve»-kommandoen, men denne metoden kan gi galt svar dersom vi skriver matrise og vektor i feil rekkefølge.

Først på rett måte, altså med kmatr som første operand og svekt som andre.

```
>> svekt = [-5 ; -3 ; -2];

>> kmatr = [2 3 -1;1 1 -1; 1 -1 -2];

>> linsolve(kmatr,svekt)


ans =

     2

    -2

     3
```

Og så på feil måte, altså med svekt som første operand og kmatr som andre.

Denne gir også et svar, og det kan være ganske skummelt! Eneste hint om at utregninga er utført på feil måte er at svarvektoren kommer i breddeformat i stedet for i høydeformat (i tillegg til at svarene selvfølgelig er feil). Derfor foretrekker i hvert fall undertegnede å bruke den første metoden, selv om denne metoden er den som krever minst inntasting.

```
>> svekt = [-5 ; -3 ; -2];

>> kmatr = [2 3 -1;1 1 -1; 1 -1 -2];

>> linsolve(svekt,kmatr)


ans =

   -0.3947   -0.4211    0.3158
```

Det bør nevnes at en backslash (skråstrek motsatt vei) gir samme resultat som linsolve.

Operanden «\» kalles "left matrix divide".

```
>> kmatr\svekt

ans =

     2

    -2

     3

>> svekt\kmatr

ans =

   -0.3947   -0.4211    0.3158
```

Metode 3, solve

Dette er kanskje den metoden som gir best bruk av symbolikk, men metoden krever dessverre en del mer tasting enn matrisemetoden. Matrisemetoden er derfor å foretrekke da den er mer tidsbesparende. Likningssettet er gitt til høyre, og svarene blir som kjent henholdsvis *2*, *-2* og *3*.

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

MATLAB-koden blir:

```
>> syms x y z

>> likn1 = 2*x + 3*y - z == -5;

>> likn2 = x + y - z == -3;

>> likn3 = x - y - 2*z == -2;

>> svar = solve([likn1, likn2, likn3] , [x, y, z]);

>> svar.x

ans =

2


>> svar.y

ans =

-2


>> svar.z

ans =

3
```

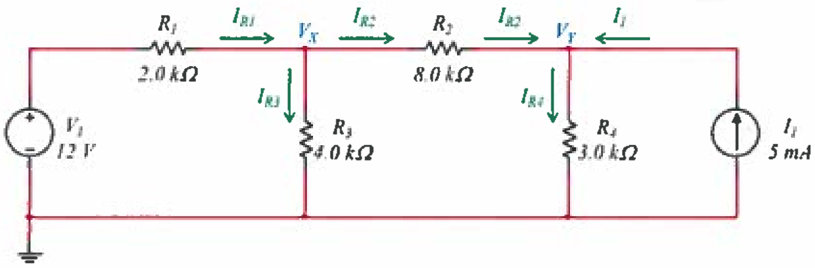
Legg merke til at vi først må definere (deklarere) tre symbolske variabler. Vi må også skille mellom enkle og doble likhetstegn. Tenk på et enkelt likhetstegn som «blir» og et dobbelt som «er lik». Svaret kommer i form av en såkalt «struktured array» eller «strukturert rekke» på norsk. Brukeren må be om enkeltelementer i denne rekka, og det gjøres med de tre siste kommandoene.

Lykke til!

Uordnede likninger med MATLAB

Kretsanalyse med MATLAB

Følgende krets skal analyseres. Det vil si, finn alle strømmer og spenninger.



Tre masker og to essensielle noder ut over referansenoden. Det betyr at nodespenningmetoden er å foretrekke.

Dersom vi er konsekvente med å ha alle spenninger i volt, alle strømmer i milliampere og alle resistanser i kiloohm, så blir likningene som følger (med V_X og V_Y som ukjente).

1. $$\frac{V_1 - V_X}{R_1} - \frac{V_X}{R_3} - \frac{V_X - V_Y}{R_2} = 0$$
   $$\frac{12 - V_X}{2} - \frac{V_X}{4} - \frac{V_X - V_Y}{8} = 0$$

2. $$\frac{V_X - V_Y}{R_2} + I_1 - \frac{V_Y}{R_4} = 0$$
   $$\frac{V_X - V_Y}{8} + 5 - \frac{V_Y}{3} = 0$$

Så her har vi altså to likninger med to ukjente. Dette kan løses med «Metode 3» fra dokumentet «Løsing av lineære likningssett».

Metode 3, solve

MATLAB-koden blir:

```
>> syms VX VY
>> likn1 = (12-VX)/2 - VX/4 - (VX-VY)/8 == 0;
>> likn2 = (VX-VY)/8 + 5 - VY/3 == 0;
>> svar = solve([likn1, likn2] , [VX,VY]);
>> svar.VX
ans =
324/37

>> double(svar.VX)
ans =
    8.7568

>> double(svar.VY)
ans =
   13.2973
```

Legg merke til at svaret opprinnelig (default) kommer i heltallsform, altså brøken 324/37. Vi kan be om å få desimalverdien med kommandoen «double». For den spesielt interesserte betyr double at tallet representeres med dobbelt presisjon, altså 64 bits i stedet for 32, eller litt mer folkelig «med en haug med siffer etter komma».

Alt.

I) $12 \cdot V_1 - 5V_2 = 50$

II) $6 \cdot V_1 - 9 \cdot V_2 = -36$

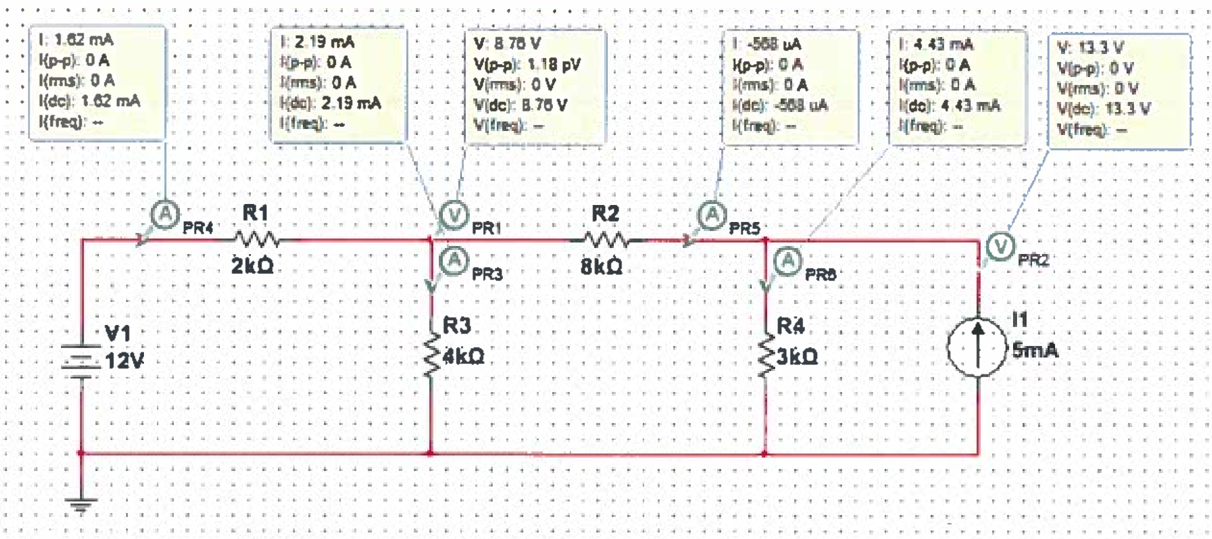
```
>> syms x y
>> likn1 = 12*x - 5*y == 50
>> likn2 = 6*x - 9*y == -36
[x, y] = solve(likn1, likn2)
```

svar direkte ut.

Når disse spenningene nå er funnet så kan alle strømmene finnes ved hjelp av loven til Georg Simon Ohm. Alle strømmene er gitt i milliampere.

```
>> IR1 = double((12-svar.VX)/2)

IR1 =

    1.6216


>> IR2 = double((svar.VX-svar.VY)/8)

IR2 =

   -0.5676


>> IR3 = double(svar.VX/4)

IR3 =

    2.1892


>> IR4 = double(svar.VY/3)

IR4 =

    4.4324
```

Nedenfor vises et faksimile fra Multisim. Dette bekrefter at utregningene er korrekte.



Lykke til!

ELE141.

Maud

Polynomer med MATLAB

Til hjelp ved kretsanalyse med MATLAB

Et polynom er i matematikk en sum av et endelig antall ledd der hvert ledd er en konstant multiplisert med en eller flere variabler opphøyd i positive heltalseksponenter. (Definisjonen er hentet fra Wikipedia).

Altså er følgende fire likninger eksempler på polynomer:

$$y = 5x^2 + 19x + 20$$
$$G = 5s^2 + 12s + 20$$
$$y = x^3 + 7x^2 + 14x + 8$$
$$y = x^3 + 3x^2 + 7x + 8$$

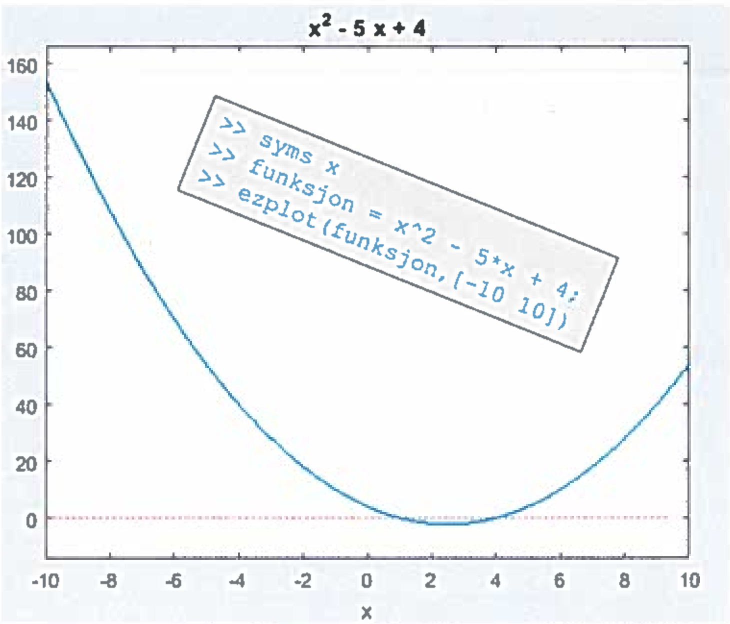
Innenfor elektrofag, slik som i de andre ingeniørfagene, beskrives ofte elementer ved hjelp av likninger, og polynomer er for eksempel brukt til å beskrive kretser med kondensatorer, spoler og resistanser. Mer om dette i ELE101.

Å finne røtter i et polynom

Røttene til et polynom med én variabel er de verdiene av variabelen som fører til at polynomet blir null. Ta for eksempel følgende polynom:

$$y(x) = x^2 - 5x + 4$$

Dersom vi lager et plot av funksjonen så får vi følgende resultat (koden er limt inn i plottet i ettertid). Den røde stiplede linja (også lagt inn i ettertid) avslører at grafen krysser linja $y = 0$ to ganger en plass når x er mellom 0 og 5.



Røttene kan vi regne oss fram til på den tradisjonelle måten:

$$y(x) = x^2 - 5x + 4 = ax^2 + bx + c \rightarrow \begin{cases} a = 1 \\ b = -5 \\ c = 4 \end{cases}$$

$$r_1 = \frac{-b - \sqrt{b^2 - 4ac}}{2a} = \frac{-(-5) - \sqrt{(-5)^2 - 4 \cdot 1 \cdot 4}}{2 \cdot 1} = \frac{5 - \sqrt{9}}{2} = 1$$

$$r_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} = \frac{-(-5) + \sqrt{(-5)^2 - 4 \cdot 1 \cdot 4}}{2 \cdot 1} = \frac{5 + \sqrt{9}}{2} = 4$$

En mer elegant måte å finne nullpunktene på er ved hjelp av funksjonen «`roots`». Her slipper vi å definere en variabel, og vi slipper å skrive så mye. Kun koeffisientene til polynomleddene.

```
>> funksjon = [1 -5 4];
>> roots(funksjon)

ans =
      4
      1
```

Vi får altså som svar at røttene (de verdiene av x som fører til at funksjonen blir null) er henholdsvis 1 og 4. Alternativt kan vi skrive det hele i en én-gang:

```
>> roots([1 -5 4])

ans =
      4
      1
```

Å faktorisere dette polynomet kan gjøres med funksjonen «`factor`».

```
>> syms x
>> funksjon = x^2 - 5*x + 4;
>> factor(funksjon)

ans =
[ x - 1, x - 4]
```

Svaret blir altså at $y(x) = x^2 - 5x + 4 = (x-1)(x-4)$

Dette kunne vi tatt rett fra «`roots`»-funksjonen. Når svaret fra «`roots`»-funksjonen er «1» og «4», altså at to positive reelle tall, så er det negative tall som må stå som konstanter i funksjonen på faktorisert form.

Når funksjonen ikke har reelle røtter:

Ta utgangspunkt i en ny funksjon:

$$y(x) = x^2 + 2x + 5$$

Hvis vi prøver «`factor`»-funksjonen på denne:

```
>> syms x
>> funksjon = x^2 + 2*x + 5;
>> factor(funksjon)

ans =
x^2 + 2*x + 5
```

Det er tydelig at «`factor`»-funksjonen nekter å faktorisere denne. Det er fordi funksjonen ikke lar seg faktorisere med reelle røtter. Men hva gir «`roots`»-funksjonen?

```
>> roots([1 2 5])

ans =
  -1.0000 + 2.0000i
  -1.0000 - 2.0000i
```

Her får vi et svar, riktignok på kompleks form. Vi som tilhører elektro-menigheten bruker den imaginære enheten *«j»* i stedet for *«i»*. Dette for å unngå forvirring med symbolet for elektrisk strøm. I likhet med eksempelet med reelle røtter, kan vi nå faktorisere polynomet, men vi må også her huske å bruke motsatt fortegn på konstantene i de faktoriserte leddene:

$$y(x) = x^2 + 2x + 5 = (x + 1 - j2)(x + 1 + j2)$$

Røttene er nå et såkalt komplekst konjugert par. Det vil si at de har lik reell del og motsatt imaginær del. Det gjelder for alle 2-ordens polynomer som ikke har reelle røtter.

Brøker med polynomer i nevner

La oss nå analysere følgende brøk:

$$y(x) = \frac{5x^2 + 19x + 20}{x^3 + 7x^2 + 14x + 8}$$

Når vi ser på nevneren til denne brøken så finner vi følgende røtter:

```
>> roots([1 7 14 8])

ans =
    -4.0000
    -2.0000
    -1.0000
```

Altså kan vi skrive om denne brøken til:

$$y(x) = \frac{5x^2 + 19x + 20}{x^3 + 7x^2 + 14x + 8} = \frac{5x^2 + 19x + 20}{(x+4)(x+2)(x+1)}$$

Ved hjelp av teknikkene for delbrøkoppspalting, så kan vi splitte denne brøken opp i tre seksjoner, hver med sitt ledd fra nevneren. Manuell delbrøkoppspalting er et tema for MAT100. Fremgangsmåten blir derfor ikke vist her, men resultatet blir:

$$y(x) = \frac{5x^2 + 19x + 20}{(x+4)(x+2)(x+1)} = \frac{4}{x+4} - \frac{1}{x+2} + \frac{2}{x+1}$$

MATLAB har en veldig hendig funksjon for dette, nemlig «`partfrac`». Navnet kommer av det engelske ordet for delbrøkoppspalting, «Partial Fractions».

```
>> syms x
>> funksjon=((5*x^2 + 19*x + 20)/(x^3 + 7*x^2 + 14*x + 8));
>> partfrac(funksjon)

ans =
2/(x + 1) - 1/(x + 2) + 4/(x + 4)
```

Dette svaret er kanskje ikke så veldig lettleselig, men da har MATLAB en vakker (pretty) funksjon på lager for oss, nemlig funksjonen «`pretty`».

```
>> syms x
>> funksjon=((5*x^2 + 19*x + 20)/(x^3 + 7*x^2 + 14*x + 8));
>> svar = partfrac(funksjon);
>> pretty(svar)
  2       1       4
----- - ----- + -----
x + 1   x + 2   x + 4
```

For all del, her var det ikke nødvendig å skrive de to første linjene om igjen, men alle gule bokser er skrevet slik at man kan «hoppe inn» hvor som helst i dokumentet. De fordrer med andre ord ikke at du har skrevet alle koder fra dokumentets første side.

I og for seg kan funksjonen «`pretty`» brukes for å lettere se om du har tastet inn riktig:

```
>> syms x
>> funksjon=((5*x^2 + 19*x + 20)/(x^3 + 7*x^2 + 14*x + 8));
>> pretty(funksjon)
      2
   5 x  + 19 x + 20
--------------------
  3      2
 x  + 7 x  + 14 x + 8
```

På grunn av den tekstbaserte oppstillinga, kan potensene være litt lett å gå glipp av.

Ofte er det nevnerens røtter som er mest interessante. Når nevneren blir null, så blir jo brøken som sådan uendelig. Innenfor reguleringsteknikk for eksempel, så er dette et tegn på ustabilitet. Dette kalles for systemets poler. På engelsk blir dette «poles».

Fra før har vi sett at nevneren i gitte eksempel er null når x = -4, x = -2 og når x = 1. Altså er systemets poler -4, -2 og -1. MATLAB har også en funksjon for dette:

```
>> syms x
>> funksjon=((5*x^2 + 19*x + 20)/(x^3 + 7*x^2 + 14*x + 8));
>> poles(funksjon)

ans =
 -2
 -4
 -1
```

På denne måten slipper vi å taste inn nevneren separat dersom vi allerede har hele funksjonen «liggende» i minnet som en variabel.

Oppgave:

a. Bruk MATLAB til å vise at:

$$y(x)=6\frac{x^2-9}{x^3+7x^2+14x+8}=6\frac{(x+3)(x-3)}{(x+4)(x+2)(x+1)}=\frac{7}{x+4}+\frac{15}{x+2}-\frac{16}{x+1}$$

b. Finn nullpunktene til den delbrøkoppspaltede funksjonen.

Lykke til!