Miniprojekt 4

i Lineær Algebra

I denne opgave skal vi se på mindste kvadraters metode i forbindelse med regression. Læs først §6.4 i bogen (dette svarer til afsnit 7.4 i udgaven fra 2014).

Som et eksempel på, hvordan vi benytter mindste kvadraters metode, gennemregnes her Example 1 på side 404 i bogen (dette svarer til side 468 i udgaven fra 2014):

```
>> x = ones(5, 1)

x =

     1
     1
     1
     1
     1

>> w = [260 272 275 267 268]'/100

w =

    2.6000
    2.7200
    2.7500
    2.6700
    2.6800

>> C = [x w]

C =

    1.0000    2.6000
    1.0000    2.7200
    1.0000    2.7500
    1.0000    2.6700
    1.0000    2.6800

>> y = [200 210 210 203 204]'/100

y =

    2.0000
    2.1000
    2.1000
    2.0300
    2.0400
```

```
>> a = inv(C' * C)*C'*y

a =

    0.0555
    0.7446
```

En alternativ måde at beregne a, er med kommandoen `mldivide`:

```
>> a = mldivide(C, y)

a =

    0.0555
    0.7446
```

Hvis ligningssystemet $Ca = y$ er inkonsistent, får vi stadig et svar med `mldivide`; svaret er en løsning i den forstand, der er omtalt på side 405 i bogen (svarende til side 469 i udgaven fra 2014).

Udover `mldivide` kan vi benytte følgende kommandoer i forbindelse med mindste kvadraters metode.

`polyfit` direkte metode til at få *koefficienterne* til den rette linie (eller mere generelt, det polynomium), vi vil beskrive data med.

For eksemplet ovenover, får vi med `polyfit`:

```
>> p = polyfit(w, y, 1)

p =

    0.7446    0.0555
```

Bemærk, at de koefficienter, vi får fra `polyfit`, er sorteret efter den potents, de hører til – læs mere i dokumentationen.

`polyval` udregner funktionsværdierne af et polynomium.

Regn opgave 1, 11 og 15 på side 409 (svarende til side 473 i 2014 udgaven), samt opgave 8 på side 484 (svarende til side 548 i 2014 udgaven).

Som nævnt på side 406 (svarende til side 470 i 2014 udgaven) i bogen er det simpelt at udvide metoden fra at finde approksimerende rette linjer til at finde approksimerende polynomier.

I stedet for at indtaste kommandoerne til at udregne a hver gang vi får nye data, kan vi samle kommandoerne i en funktion.

I tidligere opgaver har vi samlet kommandoer i en m-fil. I MATLAB-terminologi kaldes sådan en samling kommandoer for et *script*.

Forskellen på en funktion og et script er, at vi i en funktion ikke gemmer mellemregninger til senere brug, men kun får de resultater, vi har bedt om – *output* fra funktionen.

Du kan finde eksempler på funktioner på hjemmesiden i form af filerne `linreg.m` og `linreg2.m`, der udregner koefficienterne til approksimerende første- og andengradspolynomier fra mindste kvadraters metode. Brugen af de to funktioner kan du se i filen `example.m` fra hjemmesiden og i screencast 7.

Her følger også en kort introduktion til funktioner:

- En funktion skal have samme navn som den fil, man gemmer funktionen i.
- Den første linie i filen med funktionen `etellerandet` skal være

  `function ud = etellerandet( ind )`

  `ud` er her resultaterne fra funktionen og `ind` er input – på samme måde som i `polyfit` ovenover.
- Afslut funktionen med `end`.
- Det er *altid* en god ide at inkludere en forklaring i toppen af funktionen, som beskriver hvordan funktionen virker – i næste semester har du sikkert glemt hvad funktionen gør, selvom det virker indlysende når du laver den.

I funktionen `linreg2` ses et eksempel på en *løkke*. Lad os her se, hvad funktionen gør og hvorfor det er smart at bruge en løkke.

Først defineres matricen C med kommandoen

```
C = ones( size(x, 1), 3 );
```

der laver en `size(x)` $\times 3$ matrix med 1-tallet i alle indgange. Vi skal nu erstatte anden søjle med $x = [x_1, \ldots, x_n]$ og tredje søjle med $x.\hat{}2 = [x_1^2, \ldots, x_n^2]$. Dette gøres ved hjælp af løkken

```
for j = 1:2
    C(:, j+1) = x.^j;
end
```

Ovenstående løkke løber fra $j = 1$ til $j = 2$. I første gennemløb er j altså lig 1 og kommandoen `C(:, 2) = x.^1 = x` udføres. Altså: Søjle 2 erstattes med x som ønsket.

I andet gennemløb er j lig med 2 og `C(:, 3) = x.^2` udføres. Alstå: Søjle 3 erstattes med `x.^2` som ønsket.

Ønskes højere potenser svarende til en matrix af eksempelvis bredde 4, da ændres

```
C = ones( size(x, 1), 3 );
```

til

```
C = ones( size(x, 1), 4 );
```

og `for j = 1:2` til `for j = 1:3`.

Lav selv en funktion, for eksempel med navnet `linregn`, der som input tager, x-værdier, y-værdier og graden på det approksimerende polynomium n og giver koefficienterne til det approksimerende n-gradspolynomium.

Test din funktion på de data, der ligger på hjemmesiden i filen `polynomial_data.txt`, ved at approkismere dem med et tredjegradspolynomium; brug for eksempel funktionen `dlmread` til at indlæse data.

Afslutningsvis regnes MATLAB-opgaverne 5, 6 og 7 på side 289. I forbindelse med opgaverne 6 og 7 er det en god ide at kigge på eksempel 3 på side 279-280 (ignorer kommentaren om opgave 98 i afsnit 4.5).