# Løsningsforslag øving 5, ST1301

**Oppgave 1** *Newton's metode.*

Programmer en funksjon som finner løsningen på ligningen

$$e^x - 5 + x = 0, \tag{1}$$

ved hjelp av Newton's metode og som returner løsningen som funksjonsverdi. Stopp iterasjonene når den absolutte endringen i $x_i$ i løpet av en iterasjon er mindre enn $10^{-6}$. Skriv gjerne ut verdien av $x_i$ ved hver iterasjon ved å legge ett kall til `cat` eller `print` inne i while-løkken.

La $x_0$ være funksjonsargument og undersøk om løsningen påvirkes av valg av $x_0$.

Kontroller at løsningen er riktig ved innsetting.

```
newton <- function(x0=1,tol=1e-6) {
  x <- x0
  repeat {
    forrigex <- x
    x <- x - (exp(x)-5+x)/(exp(x)+1)
    if(abs(x-forrigex)<tol)
      break()
  }
  x
}
> losning <- newton()
[1] 1.344707
[1] 1.307129
[1] 1.306559
[1] 1.306559
```

Innsetting gir et svært lite tall. At det ikke er eksakt lik null skyldes at løsningen løsningen er beregnet numerisk med en nøyaktighet lik $10^{-6}$:

```
> exp(losning)-5+losning
[1] 3.019807e-14
```

Algoritmen ser ut til å fungere godt uavhengig av $x_0$. Det kan imidlertid se ut til algoritmen konvergerer svært langsomt for store $x_0$:

```
> newton(x0=5)
[1] 4.006693
[1] 3.04231
.
.
[1] 1.306559
[1] 1.306559
> newton(x0=3)
[1] 2.142278
[1] 1.547551
.
.
.
[1] 1.306559
[1] 1.306559
[1] 1.306559
> newton(x0=100)
[1] 99
[1] 98
[1] 97
.
.
.
[1] 1.306559
[1] 1.306559
[1] 1.306559
```

**Oppgave 2** *Finne dekningsgrad til forbedret konfidensintervall for parameteren $p$ i binomisk fordeling ved å simulere.*

Konfidensintervaller kan konstrueres på mange andre måter enn i Løvås (1999). Nedenfor er et alternativt konfidensintervall for parameteren $p$ i binomisk modell utledet (det alternative intervallet og utledning er ikke pensum men er tatt med til orientering).

Programmer en funksjon `konfint2` som beregner dette alternative intervallet gitt ved (12), (9), (7), og (2). Bruk lokale variabler konstruktivt for å ta vare på mellomregningsresultater. La funksjonen håndtere inn- og utdata på samme måte som funksjonen `konfint` i øving 4. Test om funksjonen fungerer for passende valg av $X$ og $n$ (f.eks. $X = 5$ og $n = 50$). Sammenlign med hva du får om du bruker "standard"-intervallet fra øving 4.

Beregn så dekningsgraden til det alternative intervallet på samme måte og for de samme parameterkombinasjonene som i øving 4, d.v.s.:

| $p$ | $n$ | $\alpha$ |
|---|---|---|
| 0.5 | 100 | 0.05 |
| 0.1 | 100 | 0.05 |
| 0.05 | 100 | 0.05 |
| 0.5 | 20 | 0.05 |
| 0.1 | 20 | 0.05 |
| 0.05 | 20 | 0.05 |

Skriv et avsnitt hvor du diskuter forskjellene i simuleringsresultater!

```
konfint2 <- function(X,n,alpha=0.05) {
   z <- qnorm(p=alpha/2,lower.tail=F)
   phat <- X/n
   a <- z^2/n
   pmerket <- (phat+a/2)/(1+a)
   rotuttrykk <- sqrt(pmerket^2-phat^2/(1+a))
   nedre <- pmerket - rotuttrykk
   ovre <- pmerket + rotuttrykk
   return(list(nedre=nedre,ovre=ovre))
}
```

I tillegg må funksjonen `dekningsgrad` (se løsningsforslag øving 4) forandres slik at denne kaller `konfint2`. Dekningsgraden for ulike parameterverdier blir nå

```
> dekningsgrad(.5,100,.05)
[1] 0.9393
> dekningsgrad(.1,100,.05)
[1] 0.9407
> dekningsgrad(.05,100,.05)
[1] 0.966
> dekningsgrad(.5,20,.05)
[1] 0.9585
> dekningsgrad(.1,20,.05)
[1] 0.9564
> dekningsgrad(.05,20,.05)
[1] 0.9252
```

Vi ser at dette konfidensintervallet fungerer svært mye bedre, selv når $np(1-p) < 5$ ligger dekningsgraden i nærheten av det nominelle nivået på 95%.

**Utledning av alternativt konfidensintervall for parameteren $p$ i binomisk modell:** I øving 4 studerte vi det reelle konfidensnivået (dekningsgraden) til "standard" konfidensintervall for parameter $p$ i et binomisk forsøk. Vi så at dekningsgraden i mange tilfeller ble liggende langt unna det det nominelle (søkte) konfidensnivået. Konfidensintervallet vi så på var utledet med utgangspunkt i $z$-transformen til

$$\hat{p} = \frac{X}{n}, \tag{2}$$

altså størrelsen

$$\frac{\hat{p}-p}{\sqrt{\hat{p}(1-\hat{p})/n}}, \tag{3}$$

som er tilnærmet standard normalfordelt. Legg imidlertid merke til at vi har en stokastisk størrelse også i nevner, noe som i praksis vil bidra ytterligere til avvik fra standardnormalfordelingen.

Et alternativt konfidensinterval for $p$ kan konstrueres med utgangspunkt i størrelsen

$$\frac{\hat{p}-p}{\sqrt{p(1-p)/n}}, \tag{4}$$

som vil være bedre tilnærmet av standardnormalfordelingen enn (3) fordi det eksakte standardavviket (uttrykkt ved parameteren $p$) inngår i nevner. Vi har da

$$P\left(-z_{\alpha/2} < \frac{\hat{p}-p}{\sqrt{p(1-p)/n}} < z_{\alpha/2}\right) \approx 1-\alpha. \tag{5}$$

Vi ønsker å løse denne dobbeltulikheten slik at vi får isolert parameteren $p$ i midten. Hendelsen vi ser på sannsynligheten til er ekvivalent med

$$\frac{(\hat{p}-p)^2}{p(1-p)/n} < z^2_{\alpha/2}. \tag{6}$$

Innfører vi

$$a = \frac{z^2_{\alpha/2}}{n}, \tag{7}$$

får vi etter litt omskriving

$$p^2 - 2\frac{\hat{p}+a/2}{1+a}p + \frac{\hat{p}^2}{1+a} < 0. \tag{8}$$

Innfører vi også

$$p' = \frac{1 \cdot \hat{p} + a \cdot \frac{1}{2}}{1 + a}, \tag{9}$$

og fullfører kvadratet i (8) får vi

$$(p - p')^2 < p'^2 - \frac{\hat{p}^2}{1 + a}, \tag{10}$$

som er ekvivalent med

$$p' - \sqrt{p'^2 - \frac{\hat{p}^2}{1 + a}} < p < p' + \sqrt{p'^2 - \frac{\hat{p}^2}{1 + a}}. \tag{11}$$

Fordi (11) er samme hendelse som i (5) er

$$\left( p' - \sqrt{p'^2 - \frac{\hat{p}^2}{1 + a}}, p' + \sqrt{p'^2 - \frac{\hat{p}^2}{1 + a}} \right) \tag{12}$$

et tilnærmet $(1 - \alpha)$ konfidensintervall for $p$. I motsetning til "standardintervallet" er dette intervallet ikke sentrert runt $\hat{p}$ men rundt $p'$. Fra (9) ser at $p'$ (midtpunktet i intervallet) kan betraktes som et vektet snitt mellom $\hat{p}$ (punktestimatet av $p$) og $1/2$ (vektene er 1 og $a$) — midtpunktet i det forbedrede konfidensintervallet blir altså trukket i retning av $1/2$.

**Oppgave 3** Følgende funksjon simulerer en realisasjon fra en bestemt sannsynlighetsfordeling. Hvilken? Forklar hvordan?

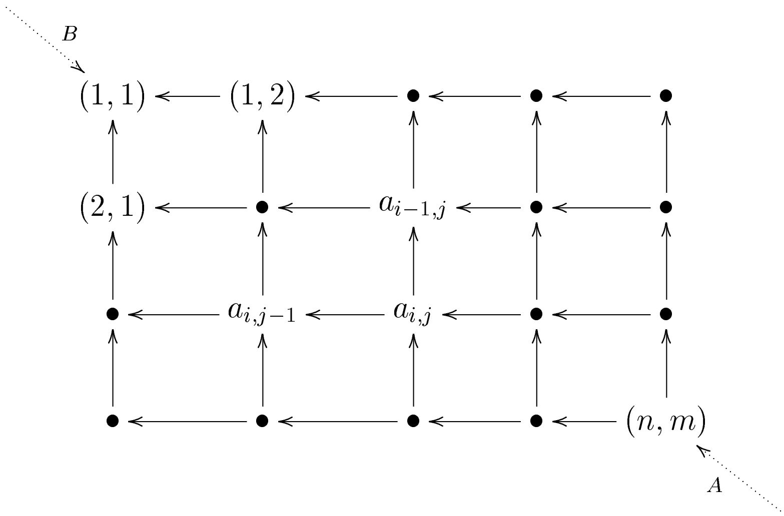
```
xxx <- function(n,p) {
   x <- 0
   nsuksess <- 0
   while (nsuksess < n) {
      y <- rbinom(n=1,size=1,prob=p)
      x <- x+1
      if (y==1) {
        nsuksess <- nsuksess + 1
      }
   }
   x
}
```

Funksjonen simulerer Bernoulli fordelte variable med parameter `p`, tilordner resultatet til `y`, teller opp totalt antall Bernoulli forsøk `x`, og antall suksesser `nsuksess`. Dette utføres helt til antall suksesser blir lik $n$. Tilslutt returneres antall delforsøk som dermed vil være negativt binomisk fordelt.

**Oppgave 4** *Bruk av nøstede løkker og matriser.*

Les først om hvordan om matriser og indeksering av matriser fungerer i Dalgaard (kap. 1.2.6 og 1.2.10). Vi refererer til et element i en matrise ved å skrive f.eks. `a[2,3]`, flere element ved å skrive `a[1:3,2:5]`, og hele rader eller kolonner ved å skrive f.eks. `a[2,]` eller `a[,4]`.

Anta at vi står i punkt A (kafé de ni muser) i rutenettet vist i figur 1 og at vi skal velge en kortest mulig vei (langs pilene) til punkt B (krambua). Programmer en funksjon som beregner antall veier fra A til B for vilkårlig valg av $m$ og $n$.




Figur 1: Kart over midtbyen...

Tips: La $a_{i,j}$ betegne antall mulige veivalg til punkt B fra kryss $(i, j)$ i rutenettet. I fra kryss langs øvre og venstre kant i rutenettet vil det åpenbart være bare en vei til B ($a_{i,j} = 1$ for $i = 1$ og $j = 1$). Anta i tillegg at vi har greid å beregne $a_{i-1,j}$ og $a_{i,j-1}$ (se figur). Hva kan vi da si om antall veier fra kryss $(i, j)$? Bruk svaret på dette spørsmålet når du skal programmere løsningen av oppgaven.

I denne oppgaven kan eksempelfasen i designplanen være spesielt nyttig for å tenke igjennom hvilke operasjoner som må gjennomføres.

```
# Kontrakt: veivalg: heltall, heltall -> heltall
#
# Hensikt:
#   Beregne antall veier fra et hjørne til et annet
#   i et rutenett ned m x n kryss
#
# Eksempel:
#   veivalg(2,1) skal gi 1 som svar
#   veivalg(1,2) skal gi 1
#   veivalg(2,2) skal gi 2
#   veivalg(2,3) skal gi 3
#   veivalg(3,2) skal gi 3
#   veivalg(3,3) skal gi 6
#
veivalg <- function(n,m) {
  a <- matrix(NA,nrow=n,ncol=m)
  a[1,] <- 1
  a[,1] <- 1
  for (i in 2:n) {
     for (j in 2:m) {
        a[i,j] <- a[i-1,j] + a[i,j-1]
     }
  }
  a[n,m]
}
# Tester:
veivalg(3,3)
veivalg(10,5)
```

Tilleggspørsmål (krever ikke programmering): Finnes det noen analytisk måte å finne antall veier på uten bruk av datamaskin? Hint: Tenk kombinatorikk og antall skritt til venstre ut av totalt antall skritt.

For å komme fra A til B må vi totalt gå $(m-1)+(n-1)$ skritt hvorav $n-1$ skritt skal være av typen V (vestre) og $m-1$ av typen H (høyre). Et bestemt veivalg vil svare til en bestemt ordning av skritt-typer, f.eks., H,V,V,H,H,H,V,V. Totalt antall mulige veivalg vil være lik antall måter denne sekvensen kan permuteres på, altså

$$\binom{m+n-2}{m-1} \tag{13}$$

Dersom vi lar funksjonen returnere hele matrisen vil vi også se sammenhengen til kombinatorikken ved at øvre venstre del av matrisen inkludert diagonalen jo er Pascals trekant.