Løsningsforslag øving 4, ST1301

Oppgave 1 *Bruk av R som statistisk tabell.*
Oppg. 2.1 og 2.2 i Dalgaard.

Oppg 2.1 a)

```
> pnorm(q=3,lower.tail=F)
[1] 0.001349898
```

b)

```
> pnorm(q=42,mean=35,sd=6,lower.tail=F)
[1] 0.1216725
```

c)

```
> dbinom(x=10,size=10,prob=0.8)
[1] 0.1073742
```

d)

```
> punif(q=0.9)
[1] 0.9
```

e)

```
> pchisq(q=6.5,df=2,lower.tail=F)
[1] 0.03877421
```

Oppg. 2.2:

```
> qnorm(p=1-0.01/2)
[1] 2.575829
> qnorm(p=1-0.005/2)
[1] 2.807034
> qnorm(p=1-0.001/2)
[1] 3.290527
```

Oppgave 2 *Undersøke om en estimator er forventningsrett.*

Anta at $X_1, X_2, \ldots, X_n$ er uavhengige kontinuerlig uniformt fordelte variabler på intervallet fra a til b. Det kan vises at variansen til uniformt fordelte

variable er

$$\sigma^2 = \mathrm{Var}(X) = \frac{(b-a)^2}{12}. \tag{1}$$

Anta at vi ønsker å estimere σ^2 ved hjelp av estimatoren

$$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2. \tag{2}$$

Undersøk om denne estimatoren er forventningsrett ved å simulere 1000 tilfeldige utvalg av størrelse $n = 5$ fra den uniforme modellen over. Velg passende verdier for a og $b = 5$. Merk at estimatoren (2) allerede finnes som funksjonen `var` i R.

```
sigma2boot <- function(a,b,n,m=1000) {
  sigma2 <- rep(NA,n)
  for (i in 1:m) {
     sigma2[i] <- var(runif(n=n,min=a,max=b))
  }
  return(sigma2)
}
```

Vi kan så undersøke om snittet av 1000 (eller 100000) bootstrap-replikater:

```
> mean(sigma2boot(a=0,b=5,n=5))
[1] 2.100403
> (5-0)^2/12
[1] 2.083333
> mean(sigma2boot(a=0,b=5,n=5,m=100000))
[1] 2.088823
```

Vi ser at forventningen nærmer seg den sanne parameterverdien når vi gjør mange simuleringer, altså er estimatoren forventningsrett. Det kan vises at S^2 gitt ved (2) alltid er forventningsrett uavhengig av fordeling.

Oppgave 3 *Forstå tolkningen av konfidensintervaller og beregne dekningsgrad gjennom å simulere. Bruk av for-løkker, logiske uttrykk, if-setninger, tellevariabler, lister.*

I forbindelse med en konsekvensutredning ønsker vi å anslå hvor stort innslaget av rømt oppdrettslaks er i en gitt lakseelv. Anta at vi fanger n laks hvorav X identifiseres som oppdrettslaks. Vi antar også at den totale mengden laks i elva er mye større enn n. Da er det rimelig å anta at X er

binomisk fordelt med parametere n og p hvor p er andelen av rømt fisk i hele elva.

I brukerkurset i statistikk har vi sett hvordan vi kan konstruere konfidensintervall for parameteren p i binomisk modell på grunnlag av observasjoner av typen over. Vi har at størrelsen $(\hat{p}-p)/\sqrt{\hat{p}(1-\hat{p})/n}$ (hvor $\hat{p}=X/n$) er tilnærmet standard normalfordelt slik at

$$P\left(-z_{\alpha/2} < \frac{\hat{p}-p}{\sqrt{\hat{p}(1-\hat{p})/n}} < z_{\alpha/2}\right) \approx 1-\alpha. \tag{3}$$

Isolerer vi p i midten får vi

$$P\left(\hat{p} - z_{\alpha/2}\sqrt{\frac{1}{n}\hat{p}(1-\hat{p})} < p < \hat{p} + z_{\alpha/2}\sqrt{\frac{1}{n}\hat{p}(1-\hat{p})}\right) \approx 1-\alpha. \tag{4}$$

Endepunktene i intervallet (det er disse som er stokastiske) skal altså ligge rundt den ukjente parameteren p med sannsynlighet $1-\alpha$ (konfidensnivået).

Programmer en funksjon `konfint` som har X, n, og α som argumenter (inndata) og som returnerer endepunktene i konfidensintervallet i form av en vektor med elementer lik nedre og øvre intervallgrense. Bruk så funksjonen for å beregne et 95% konfidensintervall for p for det tilfelle at 15 utav 50 laks er oppdrettslaks.

```
konfint <- function(X,n,alpha=0.05) {
   zkvantil <- qnorm(p=alpha/2,lower.tail=F)
   phat <- X/n
   nedre <- phat - zkvantil * sqrt(phat*(1-phat)/n)
   ovre <- phat + zkvantil * sqrt(phat*(1-phat)/n)
   c(nedre,ovre)
}
```

95% konfidensintervallet blir

```
> konfint(15,100)
$nedre
[1] 0.08001529

$ovre
[1] 0.2199847
```

Som nevnt er konfidensintervallet over ikke eksakt; dekningsgraden er bare tilnærmet lik $1-\alpha$. Vi skal nå beregne dekningsgraden ved å simulere. Programmer en funksjon som simulerer 10000 realisasjoner av av konfidensintervallet gitt ved (4) for gitte verdier av p, n og α (la disse parameterne være funksjonsargumenter). Bruk en tellevariabel og øk dennes verdi inne i for-løkken hver gang det simulerte konfidensintervallet ligger rundt p. Til slutt må dekningsgraden beregnes og returneres som funksjonsverdi.

Bruk så funksjonen til å beregne dekningsgraden for følgende verdier av p, n, og α:

p	n	α
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

Diskuter hvordan denne typen konfidensintervall fungerer i praksis på grunnlag av simuleringsresultatene.

```
dekningsgrad <- function(p,n,alpha,nsim=10000) {
  ntreff <- 0
  for (i in 1:nsim) {
     X <- rbinom(n=1,size=n,prob=p)
     ki <- konfint(X,n,alpha)
     if (ki[1]<p & p<ki[2]) {
        ntreff <- ntreff + 1
     }
  }
  return(ntreff/nsim)
}
```

Legg merke til bruken av tellevariabelen `ntreff`, og bruken av if-setning og logiske uttrykk. Legg også merke til hvordan vi unngår å gjøre mer enn ett kall til `konfint` inne i løkken ved å ta vare på resultatet av kallet i `ki`.

```
> dekningsgrad(.5,100,.05)
[1] 0.943
> dekningsgrad(.1,100,.05)
```

```
[1] 0.9334
> dekningsgrad(.05,100,.05)
[1] 0.8777
> dekningsgrad(.5,20,.05)
[1] 0.9556
> dekningsgrad(.1,20,.05)
[1] 0.8713
> dekningsgrad(.05,20,.05)
[1] 0.6473
```

Vi ser at den faktiske dekningsgraden ligger i nærheten det *nominelle* konfidensnivået $1 - \alpha = 0.95$ (dekningsgraden vi ønsker å oppnå) bare når $np(1-p) \geq 10$ (se også Løvås, 1999, s. 197). Vi ser også at dekningsgraden i mange tilfeller blir langt mindre enn det nominelle nivået. Hvis vi som forfatter av en konsekvensutredning stoler på at den faktiske dekningsgraden er lik det nominelle nivået vil vi altså i mange tilfelle framstå som overkonfidente. Moralen er at tilnærmingen denne type konfidensintervall bygger på ikke nødvendigvis alltid er veldig god!

Oppgave 4 Lag et program som tegner 4 formlike ansikter av ulik størrelse i grafikkvinduet. Lag først en funksjon som tegner en sirkelbue på en sirkel med sentrum i et vilkårlig punkt (x, y) og radius r, mellom punkter tilsvarende vinkler lik θ_1 og θ_2. Bygg så videre på denne funksjonen når du definerer nye funksjoner for å tegne sirkler, øyne, munn, og hele ansikter. Du får bruk for funksjonen `lines`.

Når du skal teste de ulike funksjonene programmet består av er det hensiktsmessig først å initiere eller tømme grafikkvinduet ved å skrive f.eks.

```
plot(NA,xlim=c(0,10),ylim=c(0,10),xlab="x",ylab="y")
```

For å løse denne programmeringsoppgave vil du måtte anvende dine domenekunnskaper innen anatomi og geometri.

```
# sirkelbue
#
# Hensikt: Tegne en sirkelbue gitt ved theta1, theta2, radius r
#    og sentrum i x,y
#
# Eksempel:
#   sirkelbue(0,pi/2,1,0,0) skal tegne den delen av
```

```
#   enhetssirkelen som befinner seg i første kvadrant.
#
# Definisjon:
sirkelbue <- function(theta1,theta2,r,x,y) {
   theta <- seq(theta1,theta2,length=100)
   xx <- x+r*cos(theta)
   yy <- y+r*sin(theta)
   lines(xx,yy)
}
# Tester
plot(NA,xlim=c(0,10),ylim=c(0,10),xlab="x",ylab="y")
sirkelbue(0,pi/2,1,0,0)


# sirkel
#
# Hensikt: tegne sirkel med radius r og sentrum i x,y
#
sirkel <- function(r,x,y)
   sirkelbue(0,2*pi,r,x,y)
# Tester
sirkel(2,4,4)


# ansikt:
#
# Hensikt: tegne et ansikt med sentrum i x,y, radius r
#   og øyne, munn, og nese i passende posisjoner
#
# Eksempel
#   ansikt(1,5,5) skal gi et ansikt med radius 1 plasser
#   i punktet 5,5 i grafikkvinduet
#
ansikt <- function(r,x,y) {
   sirkel(r,x,y)  # hode
   sirkel(r/4,x+r/2,y+r/2) # høyre øyne
   sirkel(r/4,x-r/2,y+r/2) # venstre øyne
   sirkelbue(5*pi/4,7*pi/4,3*r/4,x,y) # munn
   lines(c(x,x),c(y-r/4,y+r/4)) # nese
}
```

```
# Tester:
ansikt(2,2,2)


# familie:
#
# Hensikt: Tegne ansiktene til en kjernefamilie (løsning
#   av hele oppgaven)
#
familie <- function() {
  plot(NA,xlim=c(0,10),ylim=c(0,10),xlab="x",ylab="y")
  ansikt(2,   2, 7)
  ansikt(1.8, 8, 6)
  ansikt(1.6, 3, 2)
  ansikt(1.4, 6, 2)
}
# Tester:
familie()
```