

DATORÖVNING MED R: Labb2-GST2CL

1 Konfidensintervall och test

1.1 Konfidensintervall för väntevärden

1.1.1 normalfördelning (σ är känd)

Exempel: Ett företag observerar antalet inkommande e-mail på 8 slumpmässigt valde dagar. Resultatet blev 141, 150, 163, 139, 175, 174, 173 och 135 e-mail. Av erfarenhet vet man att antalet inkommande mail per dag är en approximativt normalfördelad variabel med standardavvikelsen 17. Beräkna ett 95% konfidensintervall för sanna medelvärdet för antalet inkommande mail.

Standardavvikelse är känd. Från teorin vet vi att intervallet ges av:

$$I_{\mu} = [\bar{x} - \lambda_{\alpha/2} \frac{\sigma}{\sqrt{n}}, \bar{x} + \lambda_{\alpha/2} \frac{\sigma}{\sqrt{n}}]$$

för normalfördelningen används funktionen `qnorm()`.

```
sigma <- 17
stickprov1 <- c(141, 150, 163, 139, 175, 174, 173, 135)
n <- 8
mu_hat <- mean(stickprov1)
lambda_alpha_2 <- qnorm(0.975)
me <- lambda_alpha_2 * (sigma/sqrt(n))
mu_hat - me # vänstra gränsen
mu_hat + me # högra gränsen
```

1.1.2 σ är okänd

Exempel: Ur en sjö har man tagit 8 vattenprover och mätt pH-värdena i dessa. Mata in data för hand och lagra i en vektor kallad `stickprov2`:

```
stickprov2 <- c(7.3, 7.5, 6.8, 7.1, 6.9, 7.2, 7.0, 6.5)
```

Gör ett antagande om normalfördelning. Med datorns hjälp är det enkelt att göra en snabb kontroll av rimligheten om normal fördelning, rita helt enkelt ett histogram och se efter om det åtminstone indikerar en symmetrisk spridning (men det kan vara svårt att bedöma med så pass få observationer):

```
hist(stickprov2)
```

Vi ska nu med hjälp av några få kommandon i R skapa ett 95% konfidensintervall för det genomsnittliga pH-värdet. Standardavvikelse är okänd. Från teorin vet vi att intervallet ges av:

$$I_{\mu} = [\bar{x} - t_{\alpha/2}(n-1) \frac{s}{\sqrt{n}}, \bar{x} + t_{\alpha/2}(n-1) \frac{s}{\sqrt{n}}]$$

Det finns två sätt att numeriskt angripa detta:

- 1) Skriv in intervallet ovan manuellt
- 2) Använd en färdig rutin (vid namn `t.test`)

För den första metoden, skriv

```
mv <- mean(stickprov2)
```

```
stad <- sd(stickprov2)
```

```
n <- 8
```

```
tfaktor <- qt(0.975, n-1)
```

```
mv - tfaktor*stad/sqrt(n)
```

```
mv + tfaktor*stad/sqrt(n)
```

Som du säkert insett ger funktionen `qt()` med lämpliga inargument kvantiler till t -fördelningen. På liknande sätt kan kvantiler för andra vanliga fördelningar erhållas.

För den andra metoden anropas helt enkelt rutinen `t.test()` och man får då ut, bokstavligen i ett enda slag, förutom själva konfidensintervallet en mängd ytterligare information. Dessutom levereras p -värde för det relaterade hypotesprövningsproblemet.

Kommando:

```
t.test(stickprov2)
```

Är man en van användare av R och kan sin statistik används med fördel metod 2). Är man nybörjare och ”vill veta vad man gör” kan metod 1) vara säkrare. Du noterade väl att de gav samma svar?

Konfidensgraden 0.95 är förinställd default i R `t.test()`, men kan ändras. Se hjälptexten för detaljer med att ge de speciellt parametern `conf.level=`. Kolla med `?t.test` att se hur kan man sätt olika input i funktionen.

Vi kan också köra `t.test()` för situationen 1.1.1 när σ är känd

`t.test (stickprov1)`

Jämföra resultat som vi gjorde manuellt med `qnorm()`

Om n är stort (det brukar vara >100 men inget standard nummer som det skulle vara), enligt centrala gränsvärdessatsen säger att summan av n stycken slumpvariabler är approximativt normalfördelad om n är tillräckligt stort. Då `t.test()` kan användas lämpligt även om σ är känd

1.2 Konfidensintervall för väntevärdesskillnad

Tryckhållfastheten för två olika betongblandningar, av typen M20 respektive M25 ska jämföras. Räkna ut medelvärde för respektive datamaterial och rita lådagram med boxplot för att undersöka om det verkar finnas någon skillnad mellan väntevärdet (μ_{M20} respektive μ_{M25}) för tryckhållfastheten för de två blandningarna.

Vi läser in data i R:

```
M20<-c(35.50, 27.80, 35.80, 30.10, 27.60, 32.45, 30.20, 26.85,  
        31.10, 19.20, 25.86, 31.20, 25.60, 31.15, 35.80, 27.50,  
        28.73, 23.20, 18.95, 24.50, 22.45, 29.80, 35.65, 30.80,  
        24.01, 25.25, 27.55, 30.15, 24.50, 22.60)
```

```
M25<-c(31.20, 35.86, 31.00, 39.01, 35.60, 38.00, 29.68, 27.26,  
        30.88, 35.50, 28.88, 38.50, 27.60, 26.00, 37.10, 30.80,  
        34.45, 38.00, 33.51, 35.80, 31.20, 36.52, 29.82, 37.80,  
        35.01, 36.60, 32.25, 31.50, 28.65, 27.55)
```

Ett 99 % konfidensintervall för väntevärdesskillnaden $\mu_{M25} - \mu_{M20}$ ges av

```
mean(M20)
```

```
mean(M25)
```

```
par(mfrow=c(1,2))
```

```
boxplot(M20)
```

```
boxplot(M25)
```

```
t.test (M25,M20, conf.level=0.99)
```

1.3 Konfidensintervall för andel

Exempel: I en by, ett slumpmässigt urval om 1000 av dessa tillfrågas om sina partisympatier, och 520 av dessa angav att de var moderater. Ange ett 95% konfidensintervall för den sanna andelen moderater i byn.

Man kan själv klart att räkna konfidensintervall för andel manuellt enligt funktion:

$$\bullet \quad I_p = [\hat{p} - \lambda_{\alpha/2} \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}, \hat{p} + \lambda_{\alpha/2} \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}]$$

Här kör vi olika alternativ i R som är lättare

[1] Alternativ 1: prop.test() från paket stats

```
library(stats)

# om du fick error eller warning om det finns ingen stats, behöver du installera
# paket stats först

install.packages("stats") #Det kan ta lite stund

library(stats)

x <- 520

n <- 1000

prop.test(x, n, conf.level=0.95, correct = FALSE)

# du kan skriva värde av x and n direkt
```

[2] Alternativ 2: binconf() from package Hmisc

```
library(Hmisc)

# om du fick error eller warning om det finns ingen binom, behöver du installera
# paket binom först

install.packages("Hmisc ") #Det kan ta lite stund

library(Hmisc)

x <- 520

n <- 1000

binconf(x, n, alpha=0.05)

# alpha: probability of a type I error, so confidence coefficient = 1-alpha

# om 99% konfidensintervall, alpha=0.01
```

[3] Alternativ 3: BinomCI() from package DescTools och kan få fem konfidensintervall med olika metoder (agresti.coull; exact; jeffreys; wald; wilson)

```
library(DescTools)

# om du fick error eller warning om det finns ingen binom, behöver du installera
```

```
# paket binom först  
install.packages("DescTools") #Det kan ta lite stund  
library(DescTools)  
x <- 520  
n <- 1000  
BinomCI(x, n, conf.level = 0.95)
```

[4] Alternativ 4: propCI() from package prevalence

```
library(prevalence)  
# om du fick error eller warning om det finns ingen binom, behöver du installera  
# paket binom först  
install.packages("prevalence") #Det kan ta lite stund  
library(prevalence)  
x <- 520  
n <- 1000  
propCI(x, n, level = 0.95)
```

[5] Det finns andra alternative t.ex. binom.test(), kolla gärna på det

1.4 Konfidensintervall för andelsskillnad

Exempel: Var det rätt låt som vann? Denna för hela nationen så viktiga fråga ställdes av en kvällstidning till ett slumpmässigt urval om 400 män och 600 kvinnor dagen efter den svenska uttagningen till Melodifestivalen. Bland männen var det 202 och bland kvinnorna 334 som svarade Ja.

Uppskatta skillnaden mellan männen och kvinnorna i hela nationen vad gäller andelen Ja-svarare med ett intervall med konfidensgraden 95%.

[1] Alternativ 1: prop.test() från paket stats

```
library(stats) # du har installerat paketet  
x <- c(202, 334)  
n <- c(400, 600)  
prop.test(x, n, conf.level=0.95, correct = FALSE)
```

[2] Alternativ 2: BinomDiffCI()from package DescTools

```
library(DescTools) # du har installerat paketet
```

```
x <- c (202, 334)
n <- c (400, 600)
BinomDiffCI(x, n, conf.level = 0.95)
```

[3] Det finns också andra alternativ,

Funtioner `diffscoreci()` eller `wald2ci()` i packet *PropCIs*

Funtionen `binomDiffCI()` I packet *MKinfer*

kolla gärna på det

2 R för att beräkna sannolikheter

Vi kan använda R för att beräkna $P(X \leq x)$ för dem flesta fördelningsfunktioner.

Normal fördelning:

Om $X \sim N(\text{mean}, \text{sd})$, vi får $P(X \leq x)$ med funktionen `pnorm(x, mean, sd)`.

`?pnorm()`

$P(x < x1) = \text{pnorm}(x1, \text{mean} = \mu, \text{sd} = \sigma, \text{lower.tail}=\text{TRUE})$

$P(x > x2) = \text{pnorm}(x2, \text{mean} = \mu, \text{sd} = \sigma, \text{lower.tail}=\text{FALSE})$

$P(x2 < x < x1) = \text{pnorm}(x1, \text{mean} = \mu, \text{sd} = \sigma, \text{lower.tail}=\text{TRUE}) - \text{pnorm}(x2, \text{mean} = \mu, \text{sd} = \sigma, \text{lower.tail}=\text{TRUE})$

2.1 Beräkna $P(X \leq 1)$ om $X \sim N(1, 1)$

```
pnorm (1, mean=1, sd=1)
```

```
# default menar lower.tail=TRUE, behöver inte at skriva mean=, sd=
```

2.2 Beräkna $P(0.5 \leq X \leq 2)$ om X följer normalfördelning med väntevärde =5, standardavvikelse 2?

```
pnorm(2, 5, 2, lower.tail=TRUE) - pnorm(0.5, 5, 2, lower.tail=TRUE)
```

2.3 Beräkna $P(X > 19)$ om $X \sim (17.46, 375.67)$?

```
pnorm(19, 17.46, sqrt(375.67), lower.tail=FALSE)
```

```
# eller pga. normal fördelning är symmetrisk, vi kan köra alternative
```

```
#  $P(X > 19) = 1 - P(X \leq 19)$ 
```

```
1 - pnorm(19, mean=17.46, sd=sqrt(375.67))
```

3 χ^2 -test

Vid statistisk analys där χ^2 -metoder används är det centrala kommandot `chisq.test()`. Vi skall här studera ett exempel för enkelt χ^2 -test.

Enkelt χ^2 -test

Vid ett korsningsförsök med en viss blomma förekommer färgerna lila, röd och vit. Enligt en genetisk teori som bygger på vissa antaganden ska färgerna förekomma i proportionerna 27 : 9 : 28. Detta medför att

$$p_1 = P(\text{lilafärgad blomma}) = \frac{27}{27+9+28} = \frac{27}{64}$$

och analogt gäller $p_2 = 9/64$ och $p_3 = 28/64$ för röd resp. vit avkomma.

En empirisk studie resulterade i följande tabell:

Färg	lila	Röd	Vit
Antal	158	19	123

Pröva på felrisknivån 5% hypotesen

$$H_0 : p_1 = \frac{27}{64}, \quad p_2 = \frac{9}{64}, \quad p_3 = \frac{28}{64}$$

Med R gör vi som följer:

```
blomfarg <-c(158,19,123);  
ph0 <-c(27/64,9/64,28/64);  
chisq.test(blomfarg, p=ph0)
```

4 Läs in /Importera data

Ibland går det att skriva in data manuellt som en vektor/matrix/data frame, men det brukar vara att det finns mycket data input och det blir jobbigt att skriva in all. Mycket vanligt och användbart är att data finns lagrat externt i en fil och vi kan läsa in/importera data direkt i R. Vi ska här studera tre fall som kan uppträda; först fallet med data i en .txt fil. Sen för inläsning av Excel filer sparas csv, xls, och xlsx format.


Innan du börja, ladda ner data: Table.txt, esoph.csv, Adata.xls, Bdata.xlsx

`getwd()`

#kolla work directory och kolla om data filen ligger i samma work directory

Rätt work directory är väldigt viktig! (tecken / i directory)

Set work directory, klicka Session sen välj Set Working Directory, sen välj den som du skulle jobba med

 RStudio

File Edit Code View Plots **Session** Build Debug Profile Tools Help

Set work director kan underlätta att importera filer utan att skriva in **mapp path** varje gång.

#OBS! Filer som du laddade ner skulle finnas i exakt samma work directory

4.1 Importera .txt filer `read.table()`

```
#importera filen Table.txt  
Tdata<- read.table("Table.txt")
```

4.2 Importera .csv filer `read.csv()`

```
#importera filen esoph.csv  
Cdata<- read.csv("esoph.csv", header = FALSE, sep = ",")
```

4.3 Importera Excel filer:

Alternativ 1:

Importera Excel filer (xls|xlsx) i R med **readxl** packet

Installera och ladda up readxl packet

```
install.packages("readxl")  
library("readxl")  
Edata1<- read_excel("Adata.xls", sheet = 1)  
#OR  
Edata1<- read.xlsx("Adata.xls",1)
```

Alternativ 2:

Använda xlsx paket

```
install.packages("xlsx")  
library("xlsx")  
Edata2<- read_excel("Bdata.xlsx", sheet = 1)  
#OR  
Edata2<- read.xlsx ("Bdata.xlsx",1)  
#OR  
Edata2<- read.xlsx2("Bdata.xlsx",1)
```

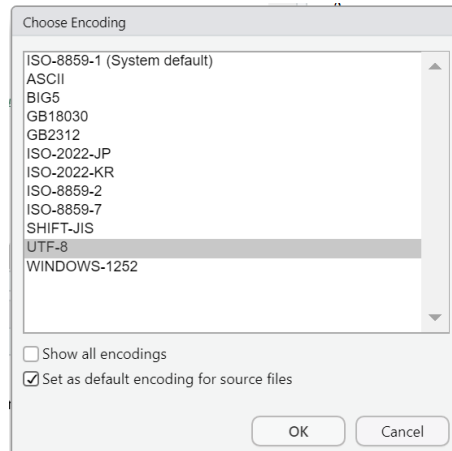
Viktig: Spara din .R fil

På grund av det finns speciella tecken i svenska: Å, Ö, Ä.

Det är mycket viktig att ni spara din .R fil med encoding.

Klicka **File** sen välja **Save as Encoding...**, sen välja **UTF-8** och **Set as default encoding for**

source file och **Save** (som figuren visas nedan).



Ni behöver inte lämna in någonting för denna labb.

Kod som jag skrev för övningarna ovan kommer finnas på kursrummet.

	$p(x)$ eller $f(x)$	$F(x)$	Simulering
Binomial, $\text{Bin}(n, p)$	<code>dbinom(x,n,p)</code>	<code>pbinom(x,n,p)</code>	<code>rbinom(N,n,p)</code>
Poisson, $\text{Po}(m)$	<code>dpois(x,m)</code>	<code>ppois(x,m)</code>	<code>rpois(N,m)</code>
Normalfördelning, $N(m, s^2)$	<code>dnorm(x,m,s)</code>	<code>pnorm(x,m,s)</code>	<code>rnorm(N,m,s)</code>
Likformig, $\text{Re}(a, b)$	<code>dunif(x,a,b)</code>	<code>punif(x,a,b)</code>	<code>runif(N,a,b)</code>
Exponential, $\text{Exp}(a)$	<code>dexp(x,1/a)</code>	<code>punif(x,1/a)</code>	<code>runif(N,1/a)</code>