LINKÖPINGS UNIVERSITET
Institutionen för datavetenskap
STIMA
Josef Wilzén

2020-03-27
Programmering i R, 7.5 hp
732G33 och 732G83

Tentamen i Programmering i R, 7.5 hp

Skrivtid: 8.00-12.00

Betygsgränser: Tentamen omfattar totalt 20 poäng. 12 poäng ger Godkänt,
16 poäng ger Väl godkänt.

Skriv dina lösningar i **fullständig och läsbar kod**. Kommentera din kod och använd en god kodstil.
Kommentera direkt i din R-fil när något behöver förklaras eller diskuteras.
Eventuella grafer som skapas under tentans gång behöver **INTE** skickas in för rättning,
det räcker med att skicka in den kod som producerar figurerna.

Instruktioner

- Se filen "**Information om hemtenta i 732G33 och 732G83.pdf**" för regler och detaljer kring tentan, där finns bland annat information om:
  - Inlämning
  - Kontakt med lärare
  - Hjälpmedel
  - Hedersintyg
  - Muntligt försvar av lösningar
- När ni är klara med tentan: På Lisam kan ni se ert Anonym-id, vilket är ert unika tenta-id. Detta ska ni använda när ni lämnar in er fil, ni namnger filen på formen [Anonym-id].txt Till exempel om ni har Anonym-id: A-12345, så ska er inlämnade fil ha namnet A-12345.txt Ni ska alltså lämna in en .txt-fil och inte en .R-fil. Eftersom det är en anonym tenta så ska denna fil ska **inte** innehålla ert namn eller Liu-ID.
- Era lösningar ska innehålla lämpliga kommenater, där ni förklarar de övergripande dragen och de viktiga stegen i er kod. Ni behöver inte föklara alla detaljer. Koden ska även ha en god kodstil. Tex så ska ni ha lämplig indentering och bra variabelnamn. Lösningar med fungerade kod men med brister i kommenater eller kodstil får poängavdrag.
- **Spara era lösningar ofta, om R kraschar kan kod förloras.**
- Tentan består av 5 uppgifter som ger 4 poäng vardera.

Uppgifter

Uppgift 1: Statistik och beräkningar 4p

Gammafördelning är inom sannolikhetslära en kontinuerlig sannolikhetsfördelning, som kan anta värden mellan 0 och oänligheten (alltså bara positiva värden). Gammafördelning har två parameterar k (kallas "shape parameter") och θ (kallas "scale parameter"). Se figur 1 för olika exempel på hur olika Gammafördelningar kan se ut. Givet en numerisk vektor med slumptal från en gammafördelning så ska ni skriva en funktion som kan skatta dessa parametrar. Vi kallar skattningarna $\hat{k}$ och $\hat{\theta}$. Funktionen ska heta `estimate_gamma(x,na.rm)`, där `x` är en numerisk vektor med slumptal från en gammafördelning. $\hat{k}$ och $\hat{\theta}$ ska skattas med formlerna nedan

$$\bar{k} = \frac{N\sum_{i=1}^{N}[x_i]}{N\sum_{i=1}^{N}[x_i log(x_i)] - \left(\sum_{i=1}^{N}[log(x_i)]\right)\left(\sum_{i=1}^{N}[x_i]\right)}$$

$$\hat{k} = \bar{k} - \frac{1}{N}\left(3\bar{k} - \frac{2}{3}\left(\frac{\bar{k}}{1+\bar{k}}\right) - \frac{4}{5}\frac{\bar{k}}{\left(1+\bar{k}\right)^2}\right) \tag{1}$$

$$\hat{\theta} = \left(\frac{N}{N-1}\right)\left(\frac{1}{N^2}\right)\left(N\sum_{i=1}^{N}\left[x_i log\left(x_i\right)\right] - \left(\sum_{i=1}^{N}\left[log\left(x_i\right)\right]\right)\left(\sum_{i=1}^{N}\left[x_i\right]\right)\right)$$

där $log\left(\right)$ är den naturliga logaritmen med bas e. N är antalet observationer i vektorn `x`. $\bar{k}$ är ett delsteg i beräkningen av $\hat{k}$. Funktionen ska testa att alla värden i x är större än 0, om de inte är de så ska funktionen avbrytas med valfritt felmeddelande. Argumentet `na.rm` är en logisk variabel som styr om NA värden ska tas bort innan beräkningen gör (=TRUE) eller om NA ska returneras (=FALSE). Funktionen ska reutnera en lista med N, $\hat{k}$, $\hat{\theta}$ och vektorn x, döp elementen till `x`, `N`, `k_hat` och `theta_hat`. Notera att ingen funktion som direkt beräknar $\hat{k}$ och $\hat{\theta}$ är tillåten, utan ni ska använda formlerna i 1. Se testfallen nedan.

```
set.seed(33)
x1<-rgamma(n = 100,shape = 4,scale = 10)
a1<-estimate_gamma(x = x1,na.rm = TRUE)
str(a1)

List of 4
 $ N        : int 100
 $ k_hat    : num 4.99
 $ theta_hat: num 7.85
 $ x        : num [1:100] 32.5 66 35.8 27.9 45 ...

set.seed(44)
x2<-rgamma(n = 500,shape = 2,scale = 5)
a2<-estimate_gamma(x = x2,na.rm = TRUE)
a2[2:3]

$k_hat
[1] 1.99808

$theta_hat
[1] 4.7161
```

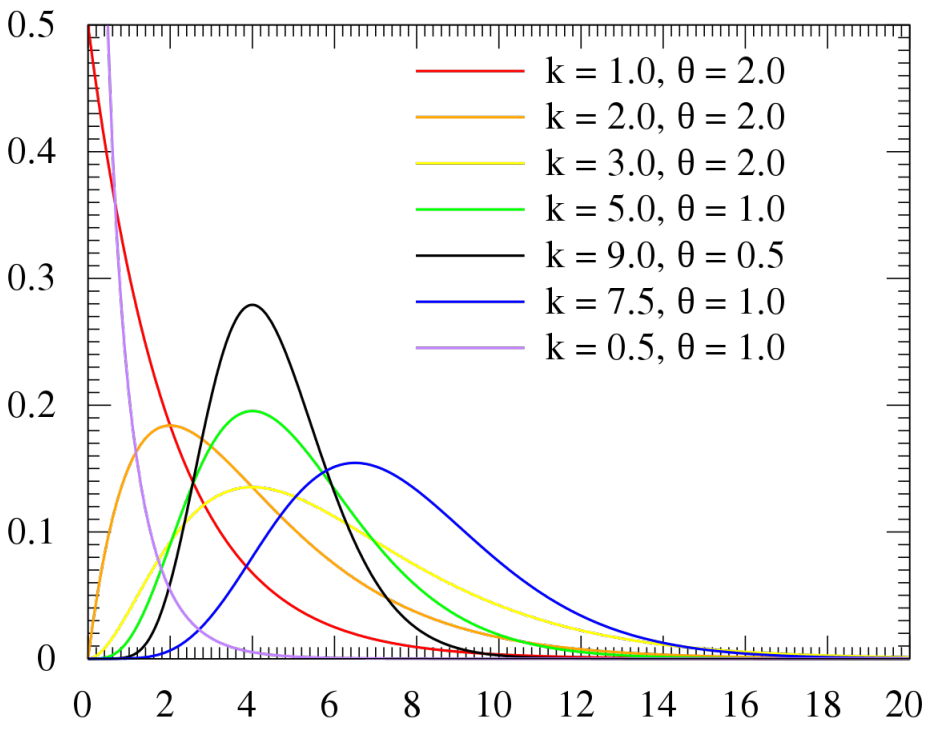


Figure 1: Exempel på olika Gammafördelningar med olika värden på k och θ.

```
estimate_gamma(x = -(1:10),na.rm = TRUE)

Error in estimate_gamma(x = -(1:10), na.rm = TRUE): Some elements in x are <=0

estimate_gamma(x = c(NA,x1,NA),na.rm = FALSE)

[1] NA

a3<-estimate_gamma(x = c(NA,x1,NA),na.rm = TRUE)
a3[2:3]

$k_hat
[1] 4.99115

$theta_hat
[1] 7.85244
```

Uppgift 2: Strängar 4p

Ni ska ska nu skapa funktionen
`change_letters(text,first=TRUE,last=TRUE)`, text är en textvektor, `first` och `last` är logiska variabler, notera defualtargumenten. Vi utgår från att text bara innehåller ett ord per element i vektorn, och att alla ord bara har små bokstäver i det engelska alfabetet. Vi antar vidare att alla ord har minst två bokstäver. Om `first=TRUE` så ska den första bokstaven i alla orden i text ändras till stor bokstav. Om `last=TRUE` så ska den sista bokstaven i alla orden i text ändras till stor bokstav. Funktionen ska sedan returnera den ändrade textvekorn. Se testfallen nedan.

```
change_letters(text = "hej",first = FALSE,last = FALSE)

[1] "hej"

change_letters(text = "hej",first = TRUE,last = FALSE)

[1] "Hej"

change_letters(text = "hej",first = FALSE,last = TRUE)

[1] "heJ"

change_letters(text = "hej",first = TRUE,last = TRUE)

[1] "HeJ"

change_letters(text = c("hej","in","skola"),first = TRUE,last = TRUE)

[1] "HeJ"   "IN"    "SkolA"

change_letters(text = c("hej","in","skola","buss","program"),first = TRUE,last = TRUE)

[1] "HeJ"     "IN"      "SkolA"   "BusS"    "PrograM"

change_letters(text = c("hej","in","skola","buss","program"),first = FALSE,last = TRUE)

[1] "heJ"     "iN"      "skolA"   "busS"    "programM"

change_letters(text = c("penna","apa","skola","program"),first = TRUE,last = FALSE)

[1] "Penna"   "Apa"     "Skola"   "Program"
```

Uppgift 3: Kontrollstrukturer

Del A 2p

Skapa funktionen `while_func(tol,l,w)`: `l` och `w` är konstanter, som är större än noll. Utgå från funktionen

$$y_t = e^{-l \cdot t} cos\,(w \cdot t) \qquad t = 0, 1, 2, 3, 4 \ldots$$

Notera att t kan anta heltalen från 0 till oändligheten. Funktion ska använda en while-loop för att göra följande: Givet värdena på `l` och `w` beräkna antalet iterationer som det krävs tills $|y_t - y_{t-1}| < tol$. Ni ska alltså beräkna y_0, y_1, y_2 osv i while-loopen och sen avbryta loopen när $|y_t - y_{t-1}| < tol$ inträffar. Funktionen ska sedan returnera en lista men antalet iterationer och värdet på y_t och y_{t-1} i den sista iterationen. Döp listelementen till `iter`, `y_last` och `y_second_last`. Se testfallen nedan.

```
a<-while_func(tol = 0.001,l = 1,w = 1)
a

$iter
[1] 5

$y_last
[1] 0.00238002

$y_second_last
[1] 0.0019113

a$y_last-a$y_second_last

[1] 0.000468723

while_func(tol = 0.1,l = 1,w = 0.1)

$iter
[1] 2

$y_last
[1] 0.0475634

$y_second_last
[1] 0.132638

while_func(tol = 0.01,l = 0.1,w = 1)

$iter
[1] 12

$y_last
[1] 0.247308

$y_second_last
[1] 0.254164
```

```
while_func(tol = 0.001,l = 0.1,w = 0.1)

$iter
[1] 23

$y_last
[1] -0.0668948

$y_second_last
[1] -0.0668001
```

Del B 2p

Skapa funktionen `matrix_func(n,m)`: Argmenten `n` och `m` är heltal större än 0. Funktion ska ska skapa en matris med `n` rader och `m` kolumner. Beräkna sedan

$$z_{i,j} = (i \cdot j)^2$$

för alla kombinationer av $i = 1, 2, \ldots n$ och $j = 1, 2, \ldots, m$. Fyll sedan matrisen med dessa värden. i är radindex och j är kolumnindex. Om ett värde $z_{i,j}$ inte är jämt delbart med 4 så ska det ersättas med 0. Denna uppgift ska lösas med en nästlad loop. Funktionen ska sedan returnera matrisen. Se testfallen nedan.

```
matrix_func(n = 1,m = 1)

     [,1]
[1,]    0

matrix_func(n = 2,m = 2)

     [,1] [,2]
[1,]    0    4
[2,]    4   16

matrix_func(n = 3,m = 3)

     [,1] [,2] [,3]
[1,]    0    4    0
[2,]    4   16   36
[3,]    0   36    0

matrix_func(n = 5,m = 3)

     [,1] [,2] [,3]
[1,]    0    4    0
[2,]    4   16   36
[3,]    0   36    0
[4,]   16   64  144
[5,]    0  100    0
```

```r
matrix_func(n = 2,m = 4)
```

```
     [,1] [,2] [,3] [,4]
[1,]    0    4    0   16
[2,]    4   16   36   64
```

```r
matrix_func(n = 5,m = 5)
```

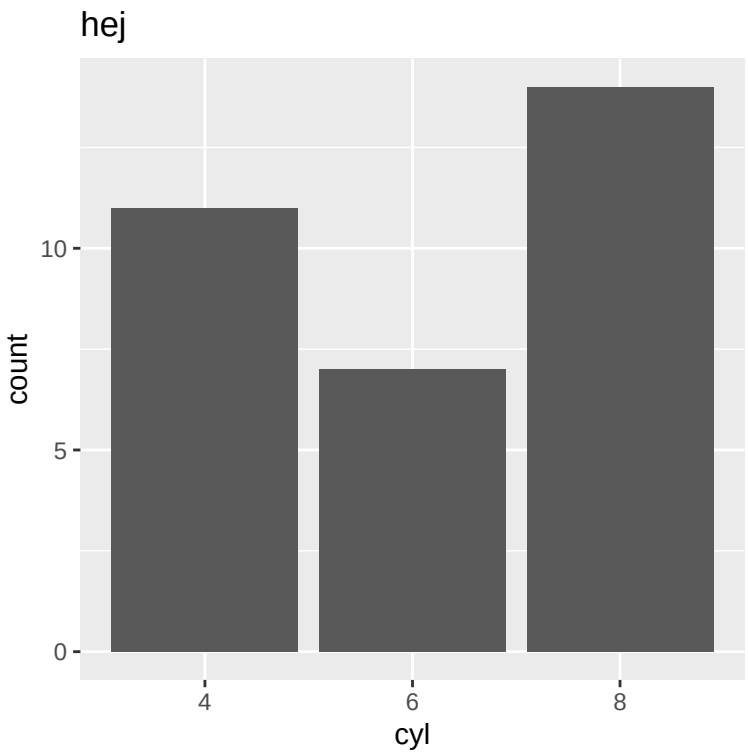
```
     [,1] [,2] [,3] [,4] [,5]
[1,]    0    4    0   16    0
[2,]    4   16   36   64  100
[3,]    0   36    0  144    0
[4,]   16   64  144  256  400
[5,]    0  100    0  400    0
```

Uppgift 4: Grafik 4p

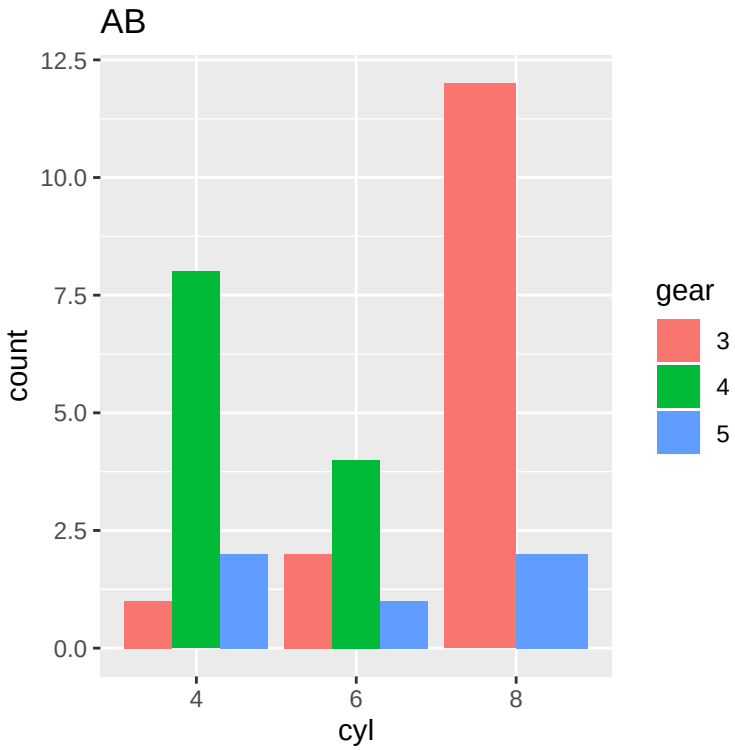
Skapa en funktion som heter `factor_plot(my_data,title)`. Funktionen ska ta ett data.frame (`my_data`) och skapa barpots (med ggplot2) . `my_data` ska ha en eller två variabler. Om inte alla variabler är factorer eller om det är fler än två variabler så ska funktion avbryta med valfritt felmeddelande. Om `my_data` har en variabel ska en vanlig barplot skapas. Om my_data har två variabler så ska grupperade barplots skapas. `title` är en textsträng och ska ange grafens titel. Se testfallen nedan[1].

```
B1<-data.frame(cyl=as.factor(mtcars$cyl))
B2<-data.frame(cyl=as.factor(mtcars$cyl),gear=as.factor(mtcars$gear))
B3<-data.frame(gear=as.factor(mtcars$gear),cyl=as.factor(mtcars$cyl))
B4<-data.frame(Species=iris$Species)

factor_plot(my_data = iris,title = "IRIS")

Error in factor_plot(my_data = iris, title = "IRIS"): too many variables

factor_plot(my_data = iris[,1:2],title = "IRIS")

Error in factor_plot(my_data = iris[, 1:2], title = "IRIS"): non factor data!

C1<-factor_plot(my_data = B1,title = "hej")
class(C1)

[1] "gg"     "ggplot"

print(C1)
```

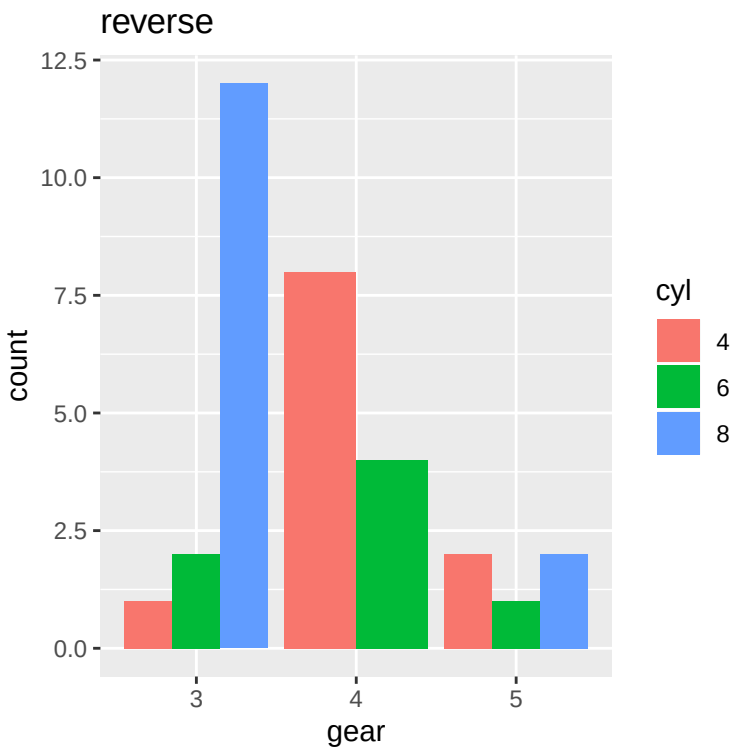
[1] Utseendet bestämmer ni själva, tex färger mm. Testfallen är exempel



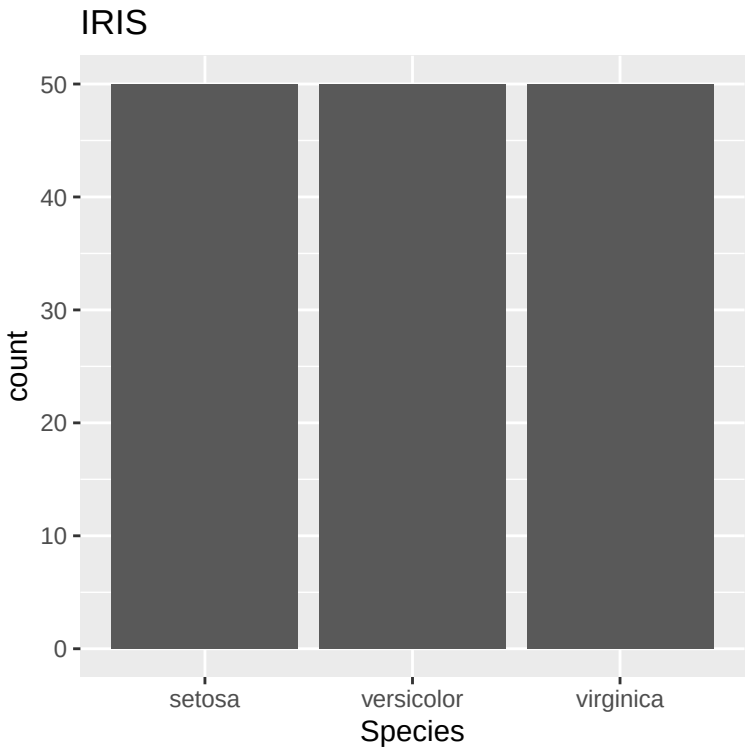

```
factor_plot(my_data = B2,title = "AB")
```



```
factor_plot(my_data = B3,title = "reverse")
```




```
factor_plot(my_data = B4,title = "IRIS")
```



Uppgift 5: Datum och dataanalys 4p

Läs in datamaterialet "Polls.csv"[2] i R och spara som en data.frame. Datamaterialet innehåller information om olika optionsundersöknginar för partisympatier i Sverige. I tabell 1 finns beskrivningar av variablerna i datamaterialet.

1. Rensa bort alla de observationer som har minst ett saknat värde (NA) på någon av följande variabler: n, PublDate, collectPeriodFrom och collectPeriodTo. Spara som en ny data.frame. Utgå från denna nya data.frame i resten av uppgiften.

2. Skapa en vektor (kalla den `no_days`) som innehåller antalet dagar för alla insamlingsperioder. Räkna sedan ut medelvärde och standardavvikelse för antalet dagar för insamlingsperiod, och spara dessa värden i två variabler.

3. Ta reda på vilken månad som det publicerades flest respektive minst undersökningar.

4. Gör ett korrelationstest mellan antalet dagar på undersökningen (`no_days`) och antalet deltagare i undersöknignen (observationer). Testa nollhypotesen att korrelation är 0. Spara den skattade korrelationen i en variabel och spara p-värdet i en annan variabel.

Variable	Description
PublYearMonth	Month and year of publication
Company	Company name at publication
M - Fi	Poll results for the different parties
Uncertain	Uncertain voters
n	The number of observations
PublDate	Date of publication
collectPeriodFrom	Start date of data collection
collectPeriodTo	End date of data collection
approxPeriod	Indicator if the period is known or if it is an approximation of the period
house	The latest companyname (if the name has been changed)

Table 1: Beskrivning av variablerna i "Polls.csv"

Kom ihåg: När ni är klara med tentan: På Lisam kan ni se ert Anonym-id, vilket är ert unika tenta-id. Detta ska ni använda när ni lämnar in er fil, ni namnger filen på formen [Anonym-id].txt Till exempel om ni har Anonym-id: A-12345, så ska er inlämnade fil ha namnet A-12345.txt Ni ska alltså lämna in en .txt-fil och inte en .R-fil. Eftersom det är en anonym tenta så ska denna fil ska **inte** innehålla ert namn eller Liu-ID.

Lycka till!

[2]Finns även här