LINKÖPINGS UNIVERSITET
Institutionen för datavetenskap
Avdelningen för statistik
Josef Wilzén

2016-08-10
Programmering i R, 7.5 hp
732G33

Tentamen i Programmering i R, 7.5 hp

Skrivtid:	8.00-12.00
Hjälpmedel:	Inget tryckt material, dock finns "R reference card v.2" av Matt Baggot, referenskort för ggplot2 och RStudio tillgängligt elektroniskt.
Betygsgränser:	Tentamen omfattar totalt 20 poäng. 12 poäng ger Godkänt, 16 poäng ger Väl godkänt.

Tänk på följande:
Skriv dina lösningar i **fullständig och läsbar kod**.
Lösningen sparas i en körbar R-fil med namnet **tenta1.R** om du har tenta-ID 1.
Spara din lösning som R-fil i mapppen **Z:\Solutions**
Se filen **DocStudent.pdf** för övrig information om tentamen.
Kommentera direkt i R-filen 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**.

OBS: Glöm inte att spara din fil ofta! Om R krashar kan kod förloras.

1. Datastrukturer och beräkningar (4p)

   (a) Beräkna $y = exp\left(\lfloor 1 + \frac{100}{x^3} \rfloor + \frac{1}{sin(x)}\right)$ där $x = 2000$ och $\lfloor\,\rfloor$avrundar till närmaste nedre heltal **1p**

   (b) Skapa en vektor med följande talstruktur: $1, 1, 1, 2, 2, 2, \ldots, 14, 14, 14, 15, 15, 15$. **1p**

   (c) Skapa en data.frame som innehåller de 3 första och de 3 sista observationerna från det inbyggda datasetet `trees`. **1p**

   (d) Skapa listan enligt nedan. **1p**

```
$a
[1] 4 3 2

$b
[1]  TRUE FALSE  TRUE  TRUE FALSE  TRUE
```

```
$c
NULL

$d
     [,1] [,2] [,3] [,4] [,5]
[1,]   10    8    6    4    2
[2,]    9    7    5    3    1
```

2. Kontrollstrukturer (4p)

   (a) Skapa en for-loop som loopar över talen $1, 2, \ldots, 100$ och beräknar kvadratroten för de tal som är jämt delbara med 13. Dessa tal ska skrivas ut i konsolen och vara avrundade till 1 decimal. Om ni gjort rätt ska det se ut enligt nedan: **2p**

```
[1] 3.6
[1] 5.1
[1] 6.2
[1] 7.2
[1] 8.1
[1] 8.8
[1] 9.5
```

   (b) Skapa en `while`-loop som beräknar den kumulativa summan av alla udda heltal mellan 1 och 13. Den kumulativa summan ska skrivas ut till konsolen med `print()` efter varje gång ett tal läggs till summan. Om ni gjort rätt ska ni erhålla: **2p**

```
[1] 1
[1] 4
[1] 9
[1] 16
[1] 25
[1] 36
[1] 49
```

3. Strängar och datum (4p)

(a) Läs in paketen `lubridate` och `stringr` i R. Läs in csv-filen `production.csv` i R. Filen innehåller information om producerade varor vid olika datum. **0.5p**

(b) Använd funktioner ur lubridate för att svara på följande frågor gällande filen `production.csv`. **1.5p**

i. Hur många enheter `iron` procuderades på lördagar och söndagar?

ii. Hur många enheter `wood` producerades i mars under alla år?

iii. Hur många enheter `wood` och `iron` producerades det totalt mellan datumen 2003-10-13 och 2009-03-12? (gränserna inkulderade)

(c) Nu ska funktionen `letter_conut(x,test)` skapas. Läs in text-filen `wiki_robot.txt` och `wiki_R.txt` i R och spara i variablerna `robot` och `wiki_R`. Dessa textvektorer ska användas för att testa funktionen. Funktionen ska ta en textvektor `x` och undersöka ofta olika bokstäver förekommer i texten, dessa bokstäver anges i textvektorn `test`. Vektorerna `x` och `test` ska kunna ha godtycklig längd. Funktionen ska skapa en frekvenstabell: första kolumnen är bokstaven och den andra är antalet förekomster i texten. Tex: Om `x=c("abc","jkl","aan?oa")` och `test=c("a","j")` så ska funktionen räkna ut att "a" förkommer 4 gånger och "j" förekommer 1 gång i textvektorn `x`. Frekvenstabellen ska returneras som en data.frame. Se testfallen nedan för hur funktionen ska fungera och hur den returnerade data.frame ska se ut: **2p**

```
test1<-letter_conut(x = robot,test = c("a","g"))
test2<-letter_conut(x = robot,test = c("a","b","c"))
test3<-letter_conut(x = wiki_R,test = c("h","z","t","p"))
test4<-letter_conut(x = wiki_R,test = c("i"))
test1

  letters count
1       a   109
2       g    18

test2

  letters count
1       a   109
2       b    18
3       c    31

test3

  letters count
```

```
1       h    33
2       z     0
3       t    79
4       p    26

test4

  letters count
1       i    73
```

4. Funktioner: (4p)

   (a) Skapa en funktion som baserat på en vektor beräknar urvalets skevhet på följande sätt: **1p**

$$\mathbf{skewness}(\mathbf{x}) = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^3}{\left(\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2\right)^{3/2}}$$

   Där $\bar{x}$ är stickprovsmedelvärdet. Inbyggda funktioner som direkt beräknar skevhet är inte tillåtet. *Exempel:*

```
skewness(x=1:100)

[1] 0

data(iris)
skewness(x=iris[,1])

[1] 0.311753
```

   (b) Nu ska funktionen `skewness_list(x)` skapas. `x` är här en lista (av godtycklig längd) där varje element innehåller en numerisk vektor. Funktionen ska gå igenom elemten i `x` och beräkna skevheten (med `skewness()` från a)) och antalet observationer. Skevheten och antalet observationer ska sparas i vektor som sedan sparas i en lista, som sedan returneras. Listan som returneras ska ha samma namn som `x`. Se testfallen hur funktionen ska fungera. **1p**

```
a<-list(hej=1:100)
b<-list(h=c(1,40,12,9),g=1:100,tree=trees[,1])
test1<-skewness_list(x = a)
test2<-skewness_list(x = b)
test1

$hej
[1]   0 100
```

```
test2

$h
[1] 0.891495 4.000000

$g
[1]   0 100

$tree
[1]  0.526316 31.000000
```

(c) Nu ska ni skapa en funktion som beräknar priset på bussbiljetter `bus_price(X)`. X är en data.frame som har kolumnerna `age`, `zones` `type`. För varje rad i X ska det aktuella busspriset beräknas enligt följande regler: `type` anger dygnsbiljett eller månadsbiljett. Grundpriset för dygnsbiljett är 100 kr och grundpris för månadsbiljett är 1000 kr. Grundpriset multicpliceras med antalet zoner för att erhålla zonpriset. Om `age` är "adult" så erhålls ingen rabbat, om `age` är "senior" då erhålls en rabbat på 10 % på zonpriset, om `age` är "youth" så erålls en rabbat på 30 % på zonpriset. Funktionen ska returnerna en vektor med priser, där varje element mostavar en rad i `X`. Se testfallen för hur funktionen ska fungera. **2p**

```
a1<-expand.grid(age=c("adult","youth","senior"),
zones=c(1:2),type=c("day","month"))
a2<-data.frame(age=c("youth","senior"),zones=4:5,type=c("month","day"))
a3<-data.frame(age=c("adult"),zones=7,type=c("day"))

b1<-bus_price(X = a1)
b2<-bus_price(X = a2)
b3<-bus_price(X = a3)
b1

 [1]  100   70   90  200  140  180 1000  700  900 2000 1400 1800

b2

[1] 2800  450

b3

[1] 700
```

5. Statistik och grafik: (4p)

(a) Skapa `X` och `y` enligt nedan. Skatta de linjära regressionskoefficenterna "för hand" genom att beräkna utrycket $\beta = \left(X^T X\right)^{-1} X^T y$

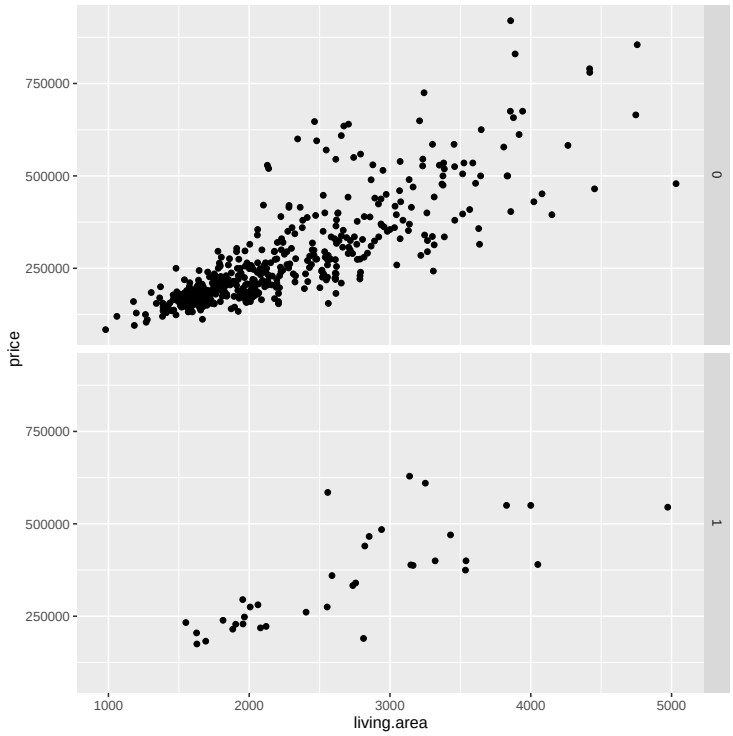
med hjälp av linjär algebra-funktioner. Funktionen `lm()` är inte tillåten här. **1p**

```
data(iris)
y<-iris[,1]
X<-cbind(1,as.matrix(iris[2:4]))
```

(b) Låt `x=c(a,b,c,d)`. Gör ett urval med återläggning från `x` med en stickprovsstorlek på 10. Sätt seeden till 967 innan urvalet görs. **0.5p**

(c) Läs in filen `HUS_eng.txt` i R och spara som en data.frame. **2.5p**

i. Använd ggplot2 för att återskapa plotten nedan. De radvisa panelerna beror på variabeln `pool`.



ii. Låt $x = \textit{living.area}$. Gör att t-test på för att testa $H_0 : \bar{x} = 2100$ mot $H_1 : \bar{x} \neq 2100$ med $\alpha = 0.01$, där $\bar{x}$ är populationsmedelvärdet. Ta fram t-värdet för testet och p-värde, spara dessa i egna variabler.

iii. Räkna ut medianen för `price` grupperat på variablen `no.bedroom`. Om ni gjort rätt ska ni erhålla 8 olika medianvärden.

Lycka till!