Homework 1

Ginevra Carbone

DAAG

Exercise 4

For the data frame ais (DAAG package) (a) Use the function str() to get information on each of the columns. Determine whether any of the columns hold missing values.

```
library(DAAG)
```

```
## Loading required package: lattice
```

```
str(ais)
```

```
## 'data.frame':    202 obs. of  13 variables:
##  $ rcc   : num  3.96 4.41 4.14 4.11 4.45 4.1 4.31 4.42 4.3 4.51 ...
##  $ wcc   : num  7.5 8.3 5 5.3 6.8 4.4 5.3 5.7 8.9 4.4 ...
##  $ hc    : num  37.5 38.2 36.4 37.3 41.5 37.4 39.6 39.9 41.1 41.6 ...
##  $ hg    : num  12.3 12.7 11.6 12.6 14 12.5 12.8 13.2 13.5 12.7 ...
##  $ ferr  : num  60 68 21 69 29 42 73 44 41 44 ...
##  $ bmi   : num  20.6 20.7 21.9 21.9 19 ...
##  $ ssf   : num  109.1 102.8 104.6 126.4 80.3 ...
##  $ pcBfat: num  19.8 21.3 19.9 23.7 17.6 ...
##  $ lbm   : num  63.3 58.5 55.4 57.2 53.2 ...
##  $ ht    : num  196 190 178 185 185 ...
##  $ wt    : num  78.9 74.4 69.1 74.9 64.6 63.7 75.2 62.3 66.5 62.9 ...
##  $ sex   : Factor w/ 2 levels "f","m": 1 1 1 1 1 1 1 1 1 1 ...
##  $ sport : Factor w/ 10 levels "B_Ball","Field",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
which(is.na(ais))
```

```
## integer(0)
```

(b) Make a table that shows the numbers of males and females for each different sport. In which sports is there a large imbalance (e.g., by a factor of more than 2:1) in the numbers of the two sexes?

```
library(tidyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
df <- as.data.frame.matrix(t(table(ais$sex, ais$sport)))
df
```

```
##          f  m
## B_Ball  13 12
```

```
## Field    7 12
## Gym      4  0
## Netball 23  0
## Row     22 15
## Swim     9 13
## T_400m  11 18
## T_Sprnt  4 11
## Tennis   7  4
## W_Polo   0 17
```

```
df[which(df$f > 2*df$m | df$f*2 < df$m),]
```

```
##          f  m
## Gym      4  0
## Netball 23  0
## T_Sprnt  4 11
## W_Polo   0 17
```

Exercise 6

Create a data frame called `Manitoba.lakes` that contains the lake's elevation (in meters above sea level) and area (in square kilometers) as listed below. Assign the names of the lakes using the `row.names()` function.

```
elevation <- c(217,254,248,254,253,227,178,207,217)
area <- c(24387,5374,4624,2247,1353,1223,1151,755,657)
names <- c("Winnipeg", "Winnipegosis", "Manitoba", "SouthernIndian", "Cedar", "Island", "Gods", "Cross"
Manitoba.lakes <- data.frame(elevation, area, row.names = names)
Manitoba.lakes
```

```
##                elevation  area
## Winnipeg             217 24387
## Winnipegosis         254  5374
## Manitoba             248  4624
## SouthernIndian       254  2247
## Cedar                253  1353
## Island               227  1223
## Gods                 178  1151
## Cross                207   755
## Playgreen            217   657
```

(a) Use the following code to plot `log2(area)` versus `elevation`, adding labeling information (there is an extreme value of area that makes a logarithmic scale pretty much essential):

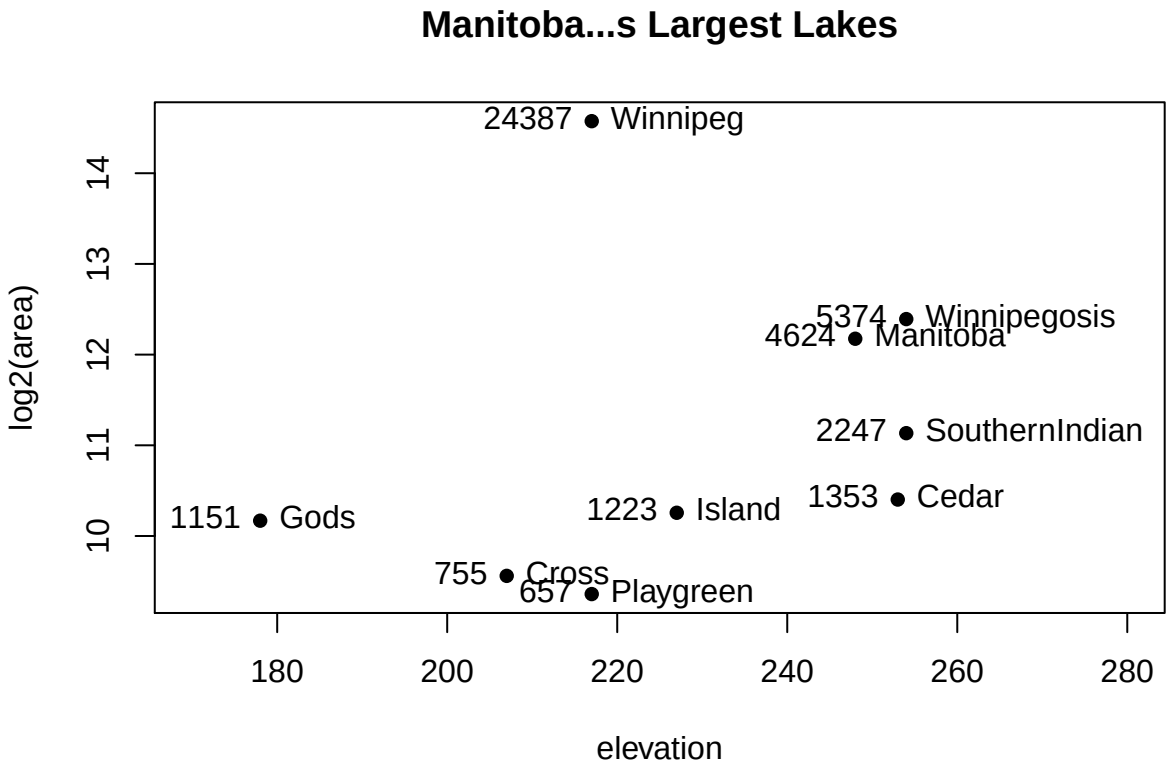
```
attach(Manitoba.lakes)
```

```
## The following objects are masked _by_ .GlobalEnv:
##
##     area, elevation
```

```
plot(log2(area) ~ elevation, pch=16, xlim=c(170,280))
# NB: Doubling the area increases log2(area) by 1.0
text(log2(area) ~ elevation, labels=row.names(Manitoba.lakes), pos=4)
text(log2(area) ~ elevation, labels=area, pos=2)
title("Manitoba's Largest Lakes")
```

```
## Warning in title("Manitoba's Largest Lakes"): conversion failure on
## 'Manitoba's Largest Lakes' in 'mbcsToSbcs': dot substituted for <e2>
```

```
## Warning in title("Manitoba's Largest Lakes"): conversion failure on
## 'Manitoba's Largest Lakes' in 'mbcsToSbcs': dot substituted for <80>

## Warning in title("Manitoba's Largest Lakes"): conversion failure on
## 'Manitoba's Largest Lakes' in 'mbcsToSbcs': dot substituted for <99>
```




```
detach(Manitoba.lakes)
```

Devise captions that explain the labeling on the points and on the y-axis. It will be necessary to explain how distances on the scale relate to changes in area.

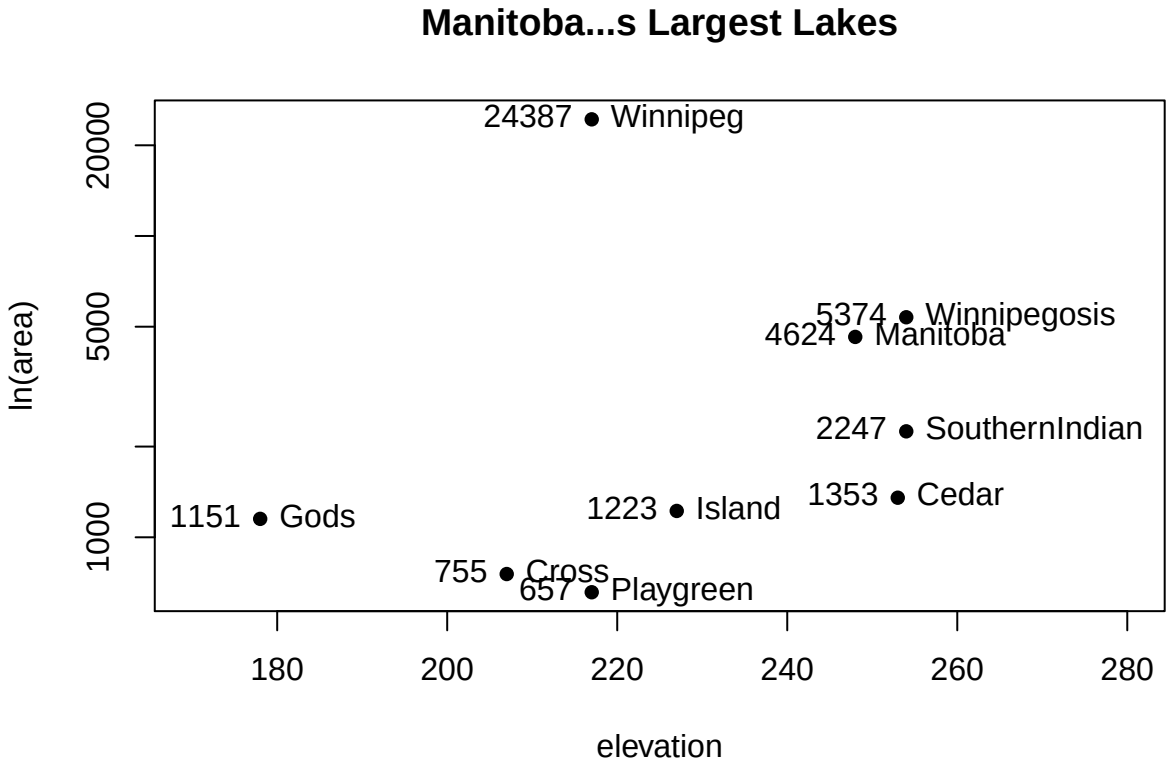
(b) Repeat the plot and associated labeling, now plotting area versus elevation, but specifying log="y" in order to obtain a logarithmic y-scale. [Note: The log="y" setting carries across to the subsequent text() commands. See Subsection 2.1.5 for an example.]

```
attach(Manitoba.lakes)
```

```
## The following objects are masked _by_ .GlobalEnv:
##
##     area, elevation
```

```
plot(area ~ elevation, log="y", pch=16, xlim=c(170,280), ylab="ln(area)")
text(area ~ elevation, labels=row.names(Manitoba.lakes), pos=4)
text(area ~ elevation, labels=area, pos=2)
title("Manitoba's Largest Lakes")
```

```
## Warning in title("Manitoba's Largest Lakes"): conversion failure on
## 'Manitoba's Largest Lakes' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in title("Manitoba's Largest Lakes"): conversion failure on
## 'Manitoba's Largest Lakes' in 'mbcsToSbcs': dot substituted for <80>

## Warning in title("Manitoba's Largest Lakes"): conversion failure on
## 'Manitoba's Largest Lakes' in 'mbcsToSbcs': dot substituted for <99>
```




```
detach(Manitoba.lakes)
```

Exercise 11

Run the following code and explain the output from the successive uses of table().

```
gender <- factor(c(rep("female", 91), rep("male", 92)))

table(gender)
```

```
## gender
## female   male
##     91     92
```

gender is a vector with two levels: "female", repeated 91 times, and "male", repeated 92 times.

```
gender <- factor(gender, levels=c("male", "female"))

table(gender)
```

```
## gender
##   male female
##     92     91
```

This commands creates a new vector from ***gender*** *by selecting the levels in reverse order.*

```
gender <- factor(gender, levels=c("Male", "female"))
# Note the mistake: "Male" should be "male"
table(gender)
```

```
## gender
##   Male female
```

```
##      0     91
```

```
table(gender, exclude=NULL)
```

```
## gender
##   Male female   <NA>
##      0     91     92
```

```
rm(gender)
# Remove gender
```

This last table includes a new column for NA values.

Exercise 12

Write a function that calculates the proportion of values in a vector `x` that exceed some value `cutoff`. (a) Use the sequence of numbers 1, 2, . . . , 100 to check that this function gives the result that is expected. (b) Obtain the vector `ex01.36` from the `Devore6` (or `Devore7`) package. These data give the times required for individuals to escape from an oil platform during a drill. Use `dotplot()` to show the distribution of times. Calculate the proportion of escape times that exceed 7 minutes.

```
prop <- function(x, cutoff){
  y <- x[x > cutoff]
  return(length(y)/length(x))
}

x <- c(1:100)
prop(x, 60)
```

```
## [1] 0.4
```

```
library(Devore7)
```

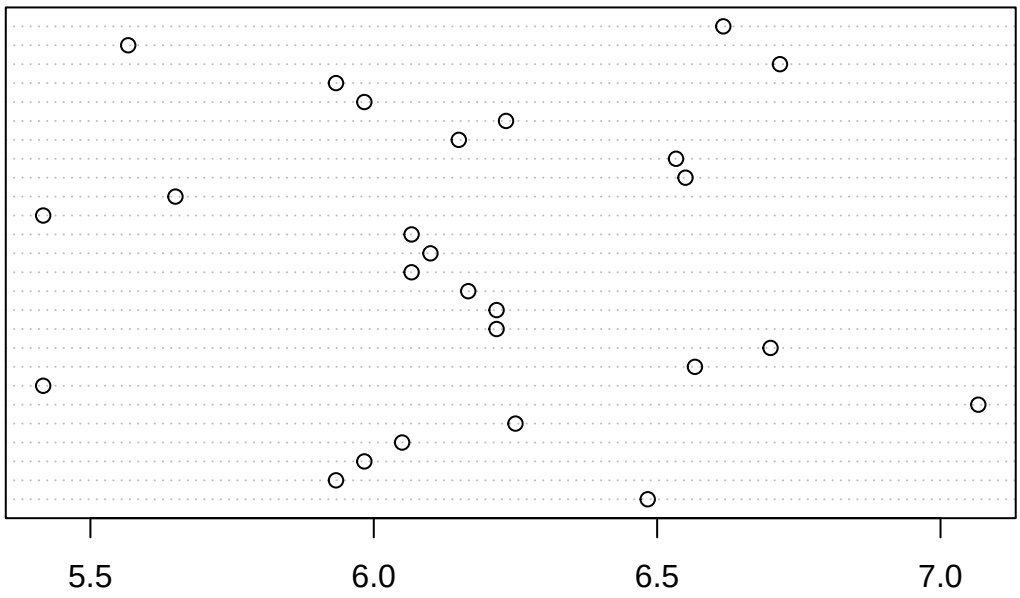
```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
## The following object is masked from 'package:DAAG':
##
##     hills
```

```
library(ggplot2)
vec <- unlist(ex01.36/60, use.names=FALSE)

# dotplot has now become dotchart
dotchart(vec, main="Data from ex01.36")
```



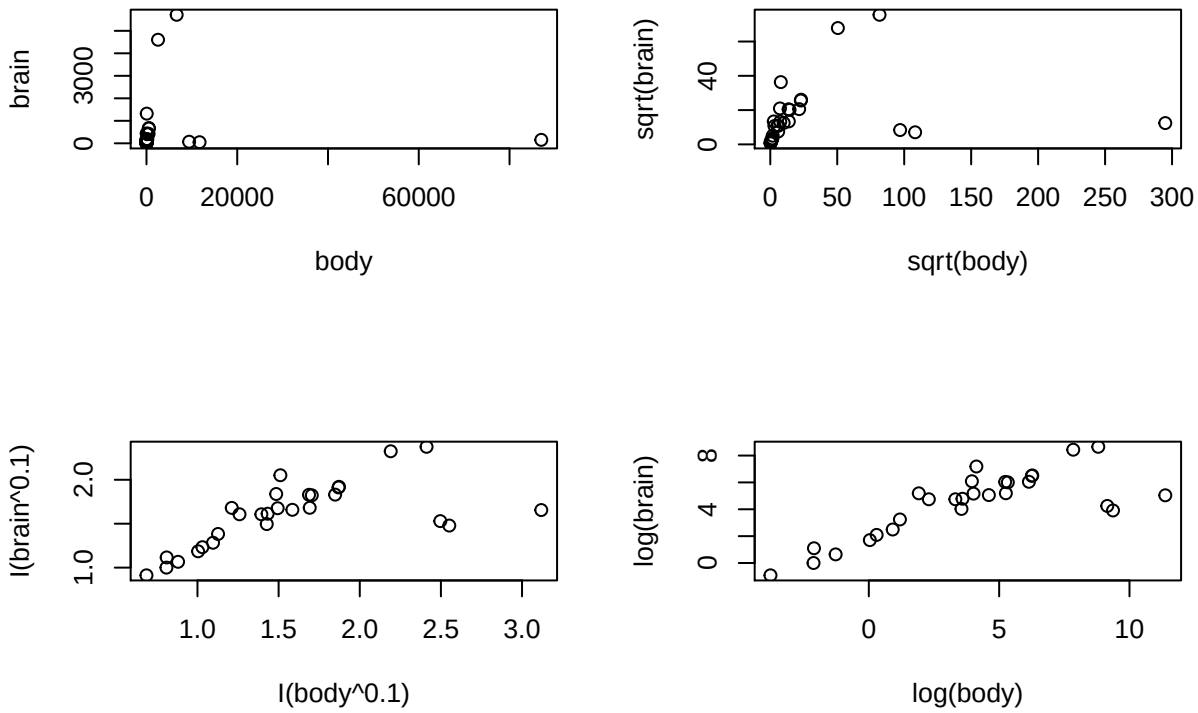

```
prop(vec, 7)
```

```
## [1] 0.03846154
```

Exercise 13

The following plots four different transformations of the Animals data from the MASS package. What different aspects of the data do these different graphs emphasize? Consider the effect on low values of the variables, as contrasted with the effect on high values.

```
par(mfrow=c(2,2))
# 2 by 2 layout on the page
library(MASS)
# Animals is in the MASS package
plot(brain ~ body, data=Animals)
plot(sqrt(brain) ~ sqrt(body), data=Animals)
plot(I(brain^0.1) ~ I(body^0.1), data=Animals)
# I() forces its argument to be treated "as is"
plot(log(brain) ~ log(body), data=Animals)
```



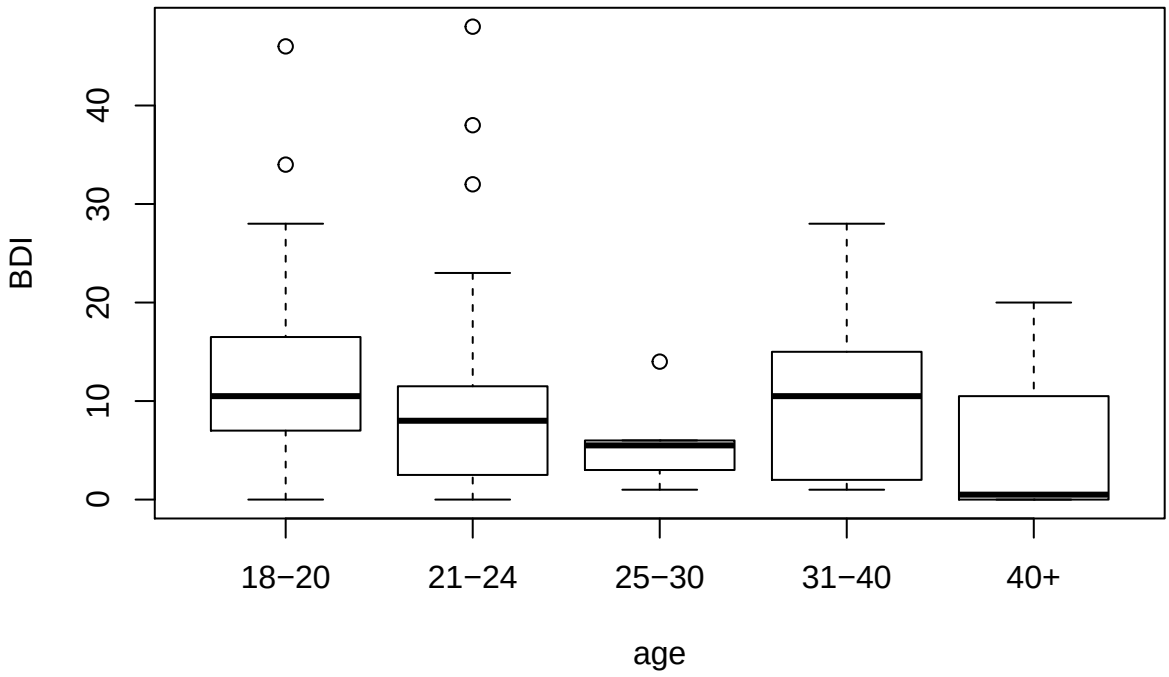

```
par(mfrow=c(1,1))
# Restore to 1 figure per page
```

These graphs show how to obtain a better visualization of data, through the composition with different functions: we can notice that in all cases low values are expanded, while high values are shrinked.

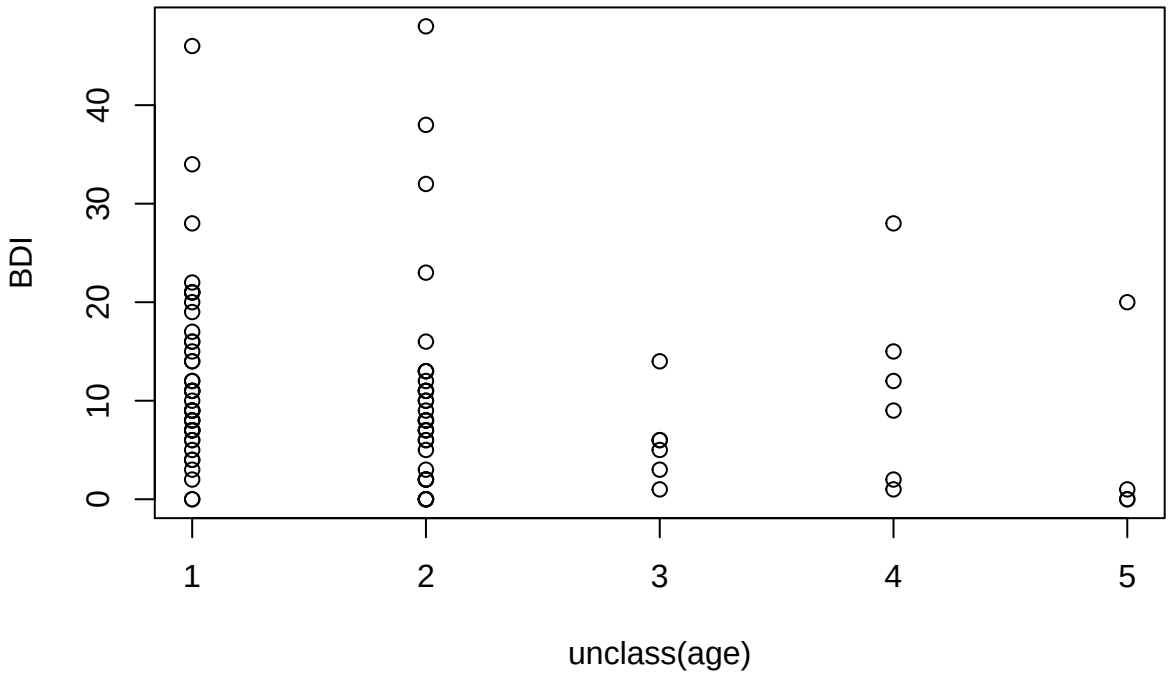
Exercise 15

The data frame socsupport (DAAG) has data from a survey on social and other kinds of support, for a group of university students. It includes Beck Depression Inventory (BDI) scores. The following are two alternative plots of BDI against age:

```
plot(BDI ~ age, data=socsupport)
```




```
plot(BDI ~ unclass(age), data=socsupport)
```



For examination of cases where the score seems very high, which plot is more useful? Explain.

18-20 and 31-40 categories are the ones having higher median. The first boxplot is more useful, since it has a higher number of observations.

Why is it necessary to be cautious in making anything of the plots for students in the three oldest age categories (25-30, 31-40, 40+)?

Because of the low number of observations available.

Exercise 17

Given a vector x, the following demonstrates alternative ways to create a vector of numbers from 1 through n, where n is the length of the vector:

```
x <- c(8, 54, 534, 1630, 6611)
seq(1, length(x))
```

```
## [1] 1 2 3 4 5
```

```
seq(along=x)
```

```
## [1] 1 2 3 4 5
```

Now set `x <- NULL` and repeat each of the calculations `seq(1, length(x))` and `seq(along=x)`. Which version of the calculation should be used in order to return a vector of length 0 in the event that the supplied argument is NULL?

The second one:

```
x <- NULL
seq(along=x)
```

```
## integer(0)
```

Exercise 20

The help page for iris (type help(iris)) gives code that converts the data in iris3 (datasets package) to case-by-variable format, with column names "Sepal.Length", "Sepal.Width", "Petal.Length", "Petal.Width", and "Species". Look up the help pages for the functions that are used, and make sure that you understand them. Then add annotation to this code that explains each step in the computation.

```
# iris3 is a tri-dimensional matrix and dimnames allows to extract the names of all categories availabl
dni3 <- dimnames(iris3)
dim(iris3)
```

```
## [1] 50  4  3
```

```
# swaps second and third dimensions of the matrix
dimnames(aperm(iris3, c(1,3,2)))
```

```
## [[1]]
## NULL
##
## [[2]]
## [1] "Setosa"     "Versicolor" "Virginica"
##
## [[3]]
## [1] "Sepal L." "Sepal W." "Petal L." "Petal W."
```

```
# substitutes each "W." with ".Width" and each "L." with ".Length" in the names of second dimension cat
sub(" L.",".Length",sub(" W.",".Width", dni3[[2]]))
```

```
## [1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"
```

```
# substites upper case "S" and "V" with lower case letters in dni3[[3]]
sub("S", "s", sub("V", "v", dni3[[3]]))
```

```
## [1] "setosa"     "versicolor" "virginica"
```

```r
# creates an enumerated vector having three levels with 50 replications each. The associated labels are
head(gl(3, 50, labels = sub("S", "s", sub("V", "v", dni3[[3]]))))
```

```
## [1] setosa setosa setosa setosa setosa setosa
## Levels: setosa versicolor virginica
```

```r
# creates a new dataframe applying all the above changes and using the last vector as a new column
ii <- data.frame(matrix(aperm(iris3, c(1,3,2)), ncol = 4, dimnames = list(NULL, sub("L.",".Length",sub(
head(ii)
```

```
##   Sepal..Length Sepal.Width Petal..Length Petal.Width Species
## 1           5.1         3.5           1.4         0.2  setosa
## 2           4.9         3.0           1.4         0.2  setosa
## 3           4.7         3.2           1.3         0.2  setosa
## 4           4.6         3.1           1.5         0.2  setosa
## 5           5.0         3.6           1.4         0.2  setosa
## 6           5.4         3.9           1.7         0.4  setosa
```

```r
# this command compares iris dataset to the one just created
all.equal(ii, iris) # TRUE
```

```
## [1] "Names: 2 string mismatches"
```

CS

Exercise 1.1

Exponential random variable, $X0$, has p.d.f. $f(x) = exp(x)$. 1. Find the c.d.f. and the quantile function for X.

*The p.d.f. is

$$f_X(x) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0 \\ 0 & x < 0 \end{cases}$$

so the corresponding c.d.f. is given by

$$F_X(x) = P(X \leq x) = \int_{-\infty}^{x} f_X(t)dt = \int_0^x \lambda e^{-\lambda t} dt \tag{1}$$

$$= [-e^{-\lambda t}]_0^x = -e^{-\lambda x} - (-1) = 1 - e^{-\lambda x} \tag{2}$$

when $x \geq 0$ and it is $F_X(x) = 0$ for $x < 0$.*

The quantile function is the inverse of the cumulative distribution function, and is given by

$$F^{-1}(p) = -\frac{ln(1-p)}{\lambda}$$

$\forall\, 0 \leq p < 1$.

2. Find $Pr(X <)$ and the median of X.

$$P(X < \lambda) = \begin{cases} 1 - e^{-\lambda^2} & x \geq 0 \\ 0 & x < 0 \end{cases}$$

The median is

$$m(X) = F^{-1}\left(\frac{1}{2}\right) = \frac{ln(2)}{\lambda}.$$

3. Find the mean and variance of X.

$$E(X) = \int_{-\infty}^{\infty} \lambda x e^{-\lambda x} dx = \lim_{t\to\infty} \int_0^t \lambda x e^{-\lambda x} dx = \lim_{t\to\infty} \int_0^t \frac{u}{\lambda} e^{-\lambda u} du \tag{3}$$

$$= \frac{1}{\lambda} lim_{t\to\infty}(1 - (t+1)e^{-t}) = \frac{1}{\lambda} \tag{4}$$

$$var(X) = E(X^2) - E(X)^2 = \int_0^{\infty} x^2 \lambda e^{\lambda x} dx - \frac{1}{\lambda^2} = \frac{1}{\lambda^2}$$

Exercise 1.2

Evaluate $Pr(X < 0.5, Y < 0.5)$ if X and Y have joint p.d.f.

$$f(x,y) = \begin{cases} x + 3y^2/2 & 0 < x, y < 1 \\ 0 & otherwise \end{cases}.$$

$$Pr(X < 0.5, Y < 0.5) = \int_0^{1/2} \int_0^{1/2} x + \frac{3}{2} y^2 \, dx \, dy \tag{5}$$

$$= \int_0^{1/2} \left[\frac{x^2}{2} + \frac{3}{2} x y^2\right]_0^{1/2} dy \tag{6}$$

$$= \int_0^{1/2} \frac{1}{8} + \frac{3y^2}{4} \, dy \tag{7}$$

$$= \left[\frac{y}{8} + \frac{y^3}{4}\right]_0^{1/2} \tag{8}$$

$$= \frac{3}{32} \tag{9}$$

Exercise 1.6

Let X and Y be non-independent random variables, such that $var(X) = \sigma_x^2$, $var(Y) = \sigma_y^2$ and $cov(X,Y) = \sigma_{xy}^2$. Using the result from Section 1.6.2, find $var(X+Y)$ and $var(XY)$.

Since the expected value is a linear operator, we have

$$var(X+Y) = cov(X+Y, X+Y) = E((X+Y)^2) + E(X+Y)^2 \tag{10}$$

$$= var(X) + var(Y) + 2cov(X,Y) \tag{11}$$

$$= \sigma_x^2 + \sigma_x^2 + 2\sigma_{xy}^2 \tag{12}$$

$$var(X-Y) = E((X-Y)^2) + E(X-Y)^2 \tag{13}$$

$$= var(X) + var(Y) - 2cov(X,Y) \tag{14}$$

$$= \sigma_x^2 + \sigma_x^2 - 2\sigma_{xy}^2 \tag{15}$$

.

Exercise 1.8

If $log(X) \sim N(\mu, \sigma^2)$, find the p.d.f. of X.

Let's consider $Y = logX$, then X has log-normal distribution:

$$f_X(x) = f_Y(y)\left|\frac{dy}{dx}\right| = f_Y(logx)\left|\frac{dlogx}{dx}\right| = \frac{1}{x\sqrt{2\pi\sigma^2}}e^{-\frac{(logx-\mu)^2}{2\sigma^2}}$$

Exercise 1.9

Discrete random variable Y has a Poisson distribution with parameter λ if its p.d.f. is $f(y) = \lambda^y e^{-\lambda}/y!$, for $y = 0, 1, \ldots$.

a. Find the moment generating function for Y (hint: the power series repre-sentation of the exponential function is useful).

For $t \in \mathbb{R}$

$$M_Y(t) = E(e^{tY}) = \sum_{y=0}^{\infty} e^{ty}\frac{\lambda^y e^{-\lambda}}{y!} = e^{-\lambda}\sum_{y=0}^{\infty}\frac{(\lambda e^t)^y}{y!} = e^{\lambda(e^t-1)}$$

b. If $Y_1 \sim Poi(\lambda_1)$ and independently $Y_2 \sim Poi(\lambda_2)$, deduce the distribution of $Y_1 + Y_2$, by employing a general property of m.g.f.s.

The condition

$$M_{Y_1+Y_2}(t) = M_{Y_1}(t)M_{Y_2}(t) = e^{\lambda_1(e^t-1)}e^{\lambda_2(e^t-1)} = e^{(\lambda_1+\lambda_2)(e^t-1)}$$

holds for every $t \in \mathbb{R}$, so $Y_1 + Y_2 \sim Poi(\lambda_1 + \lambda_2)$.

c. Making use of the previous result and the central limit theorem, deduce the normal approximation to the Poisson distribution.

If $X_1, \ldots, X_n \sim Poi(\lambda)$ are independent identically distributed random variables with mean $\mu = \lambda$ and variance $\sigma^2 = \lambda$, their sum is still a Poisson distribution with mean $\mu = n\lambda$ and variance $\sigma^2 = \lambda$. Thanks to central limit theorem, as $n \to \infty$ the following approximation holds

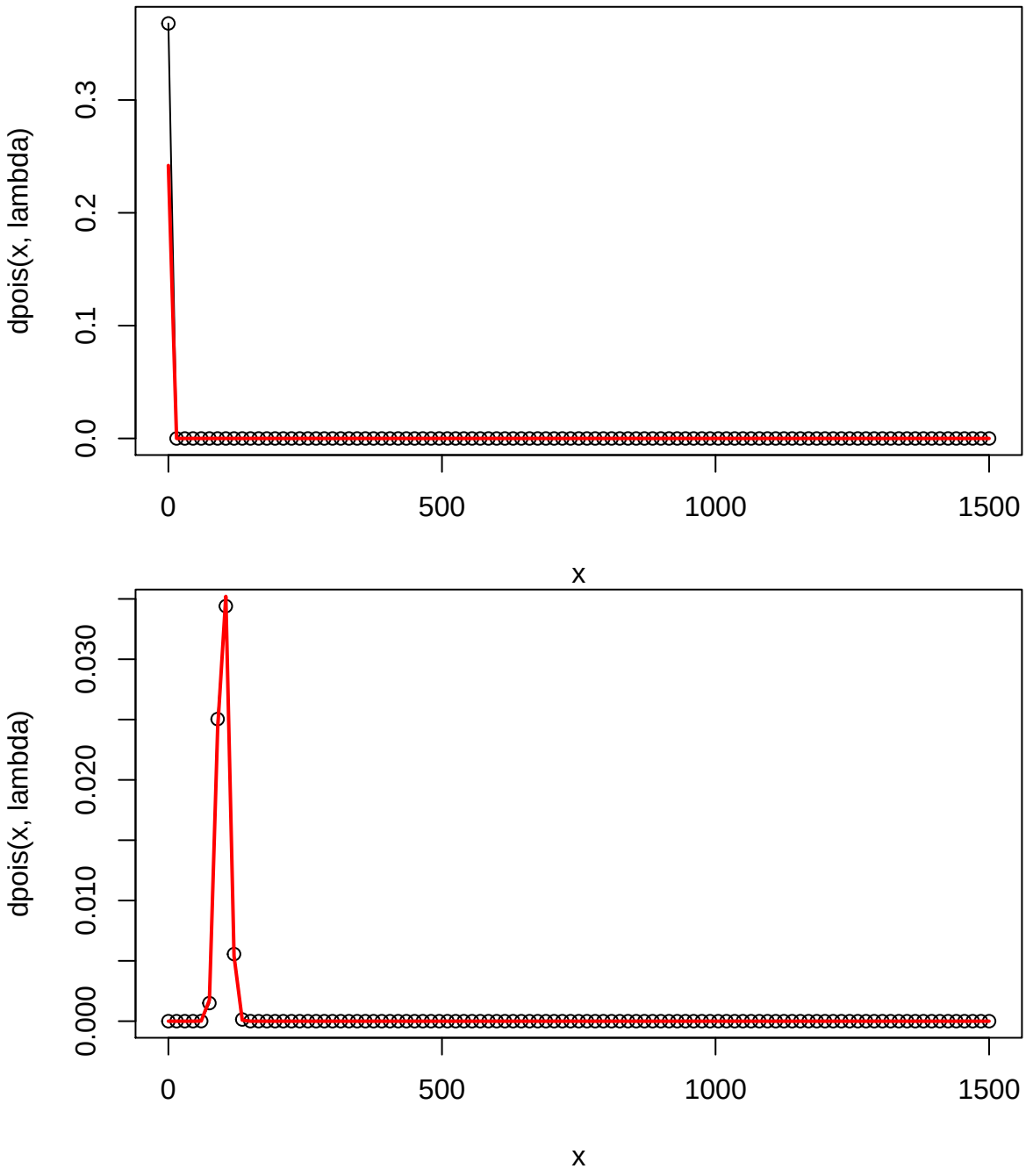
$$\overline{X}_n \sim \mathcal{N}\left(\lambda, \frac{\lambda}{n}\right)$$

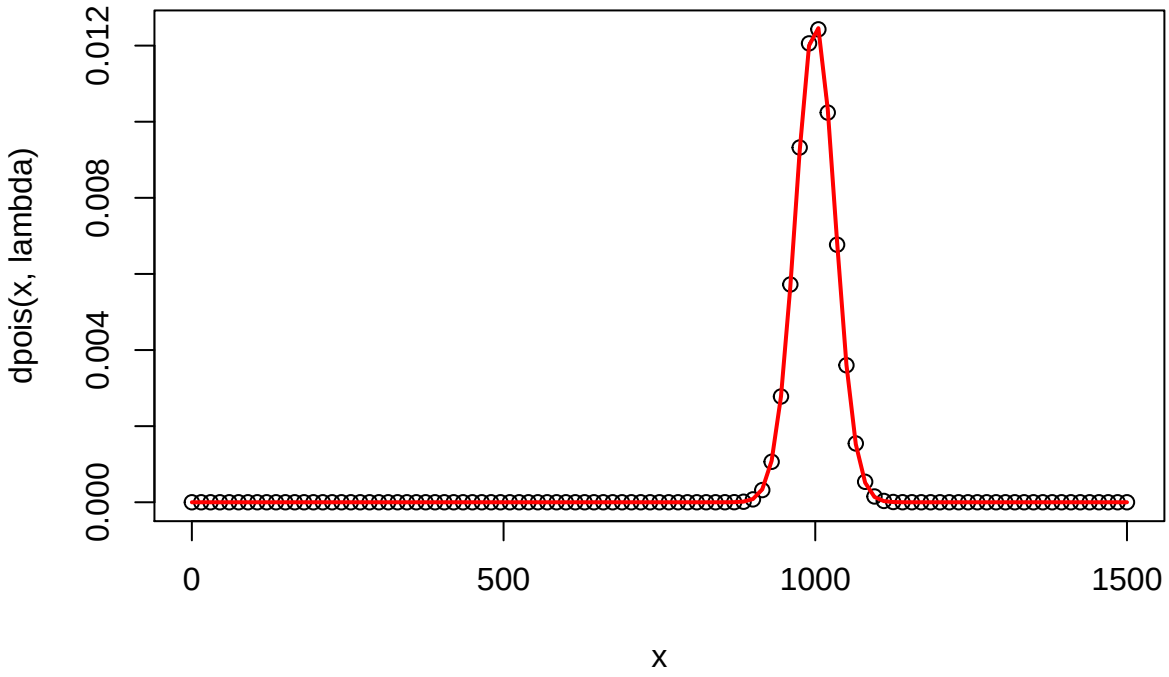
In particular if X is the sum of n i.i.d. random variables with distribution $Poi(\frac{\lambda}{n})$, then

$$X \sim \mathcal{N}(\lambda, \lambda).$$

d. Confirm the previous result graphically, using R functions `dpois` , `dnorm` , `plot` or `barplot` and `lines`. Confirm that the approximation improves with increasing .

```
for (lambda in c(1, 100, 1000)){
  curve(dpois(x, lambda), xlim=c(0,1500), type = "o")
  curve(dnorm(x, lambda, sqrt(lambda)), col="red", lwd=2, add=TRUE)
}
```


dpois(x, lambda)
0.0
0.1
0.2
0.3
0
500
1000
1500
x
dpois(x, lambda)
0.000
0.010
0.020
0.030
0
500
1000
1500
x



LAB

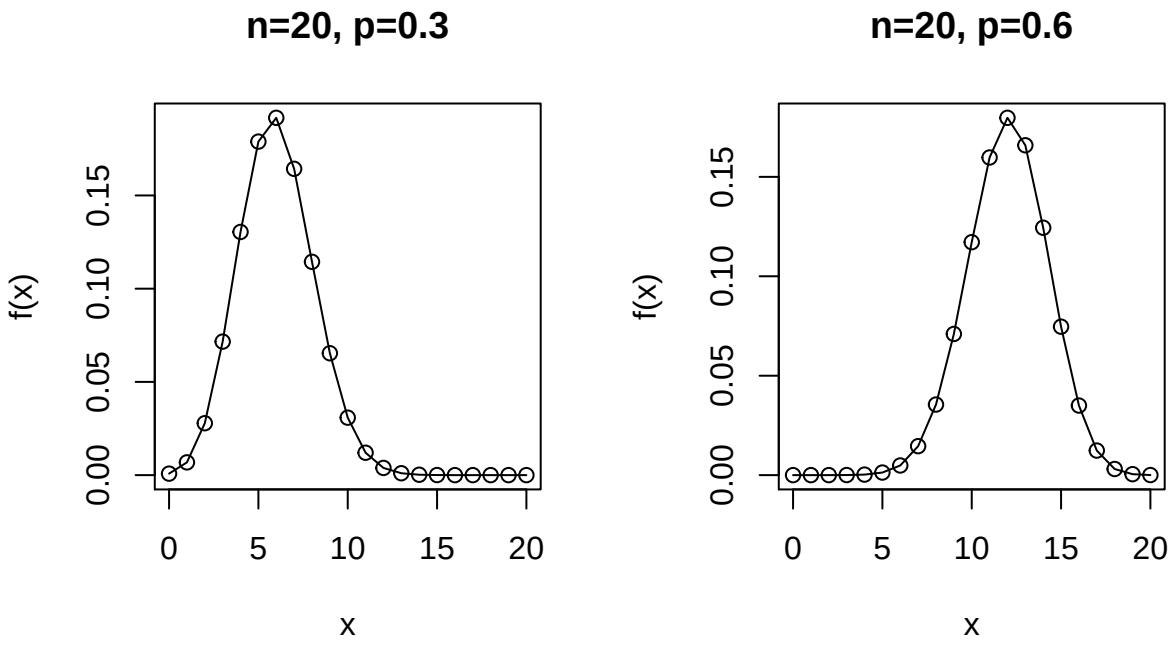
Exercise 1

- Write a function binomial(x, n, p) for the binomial distribution, depending on parameters x, n, p, and test it with some prespecified values. Use the function choose() for the binomial coefficient.

```
binomial <- function(x,n,p){
 y=choose(n,x)*p^{x}*(1-p)^{n-x}
return(y)
}
```

- Plot two binomials with $n = 20$, and $p = 0.3, 0.6$ respectively.

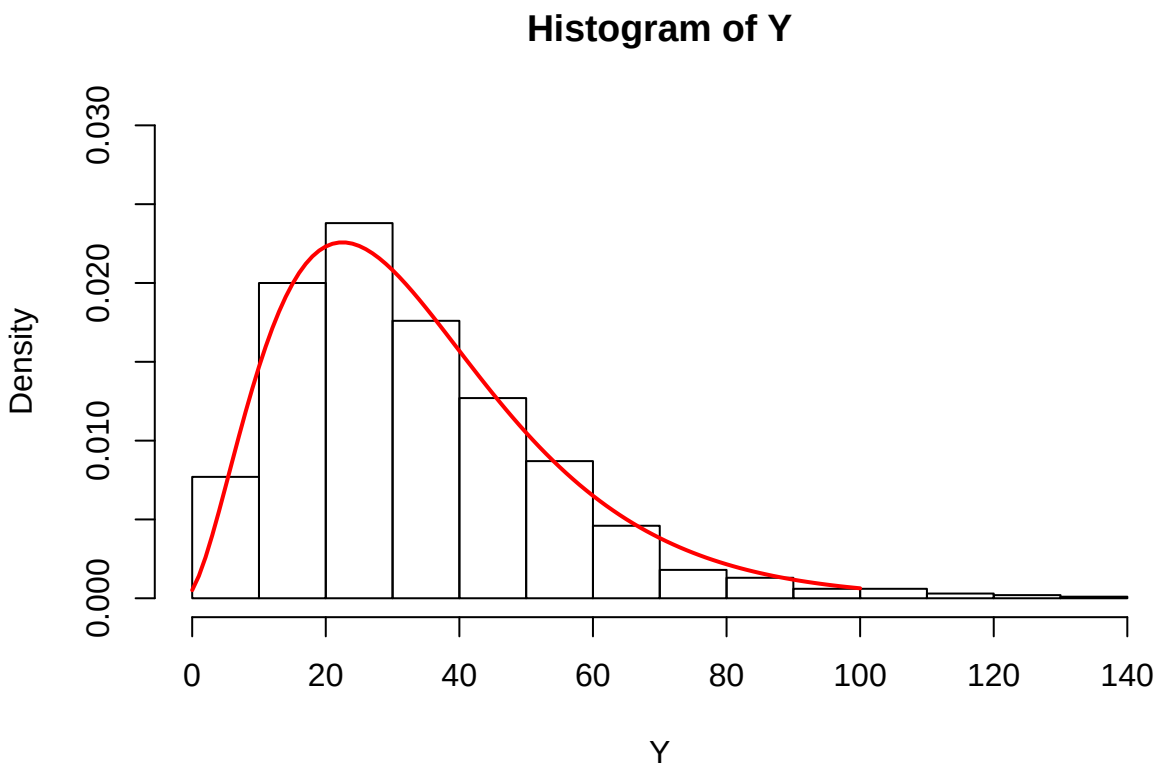
```
#graphical setting for margins and type of points
par(mfrow=c(1,2), pty="s")
#plot the binomial distributions with different input
plot(0:20, binomial(0:20, 20, 0.3), type = "o", xlab = "x", ylab = "f(x)", main="n=20, p=0.3")
plot(0:20, binomial(0:20, 20, 0.6),  type = "o", xlab = "x", ylab = "f(x)", main="n=20, p=0.6")
```



Exercise 2

- Generate in R the same output, but using rgeom() for generating the random variables. *Hint*: generate n times three geometric distribution $X_1, \ldots, X_3$ with $p = 0.08$, store them in a matrix and compute then the sum Y.

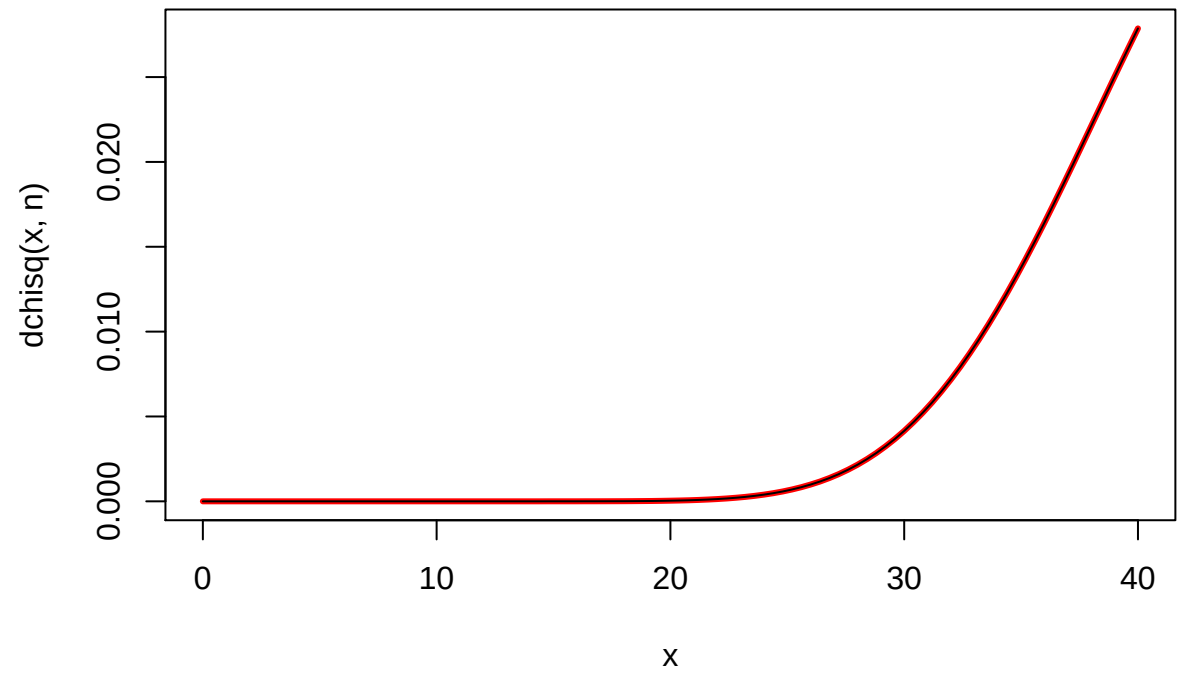
```
par(mfrow=c(1,1))
n <- 1000 # sample size
p <- 0.08 # success probability
k <- 3    # predef. number of successes

X <- matrix(NA, n, 3)
for (i in 1:n){
  X[i,] <- rgeom(3, p) # store in the i-th row
}
Y <- apply(X, 1, sum) # sum over rows

hist(Y, ylim=c(0,0.03) ,probability = TRUE)
curve(dnbinom(x, 3, p), col="red", lwd=2, add=TRUE, xlim=c(0,100))
```



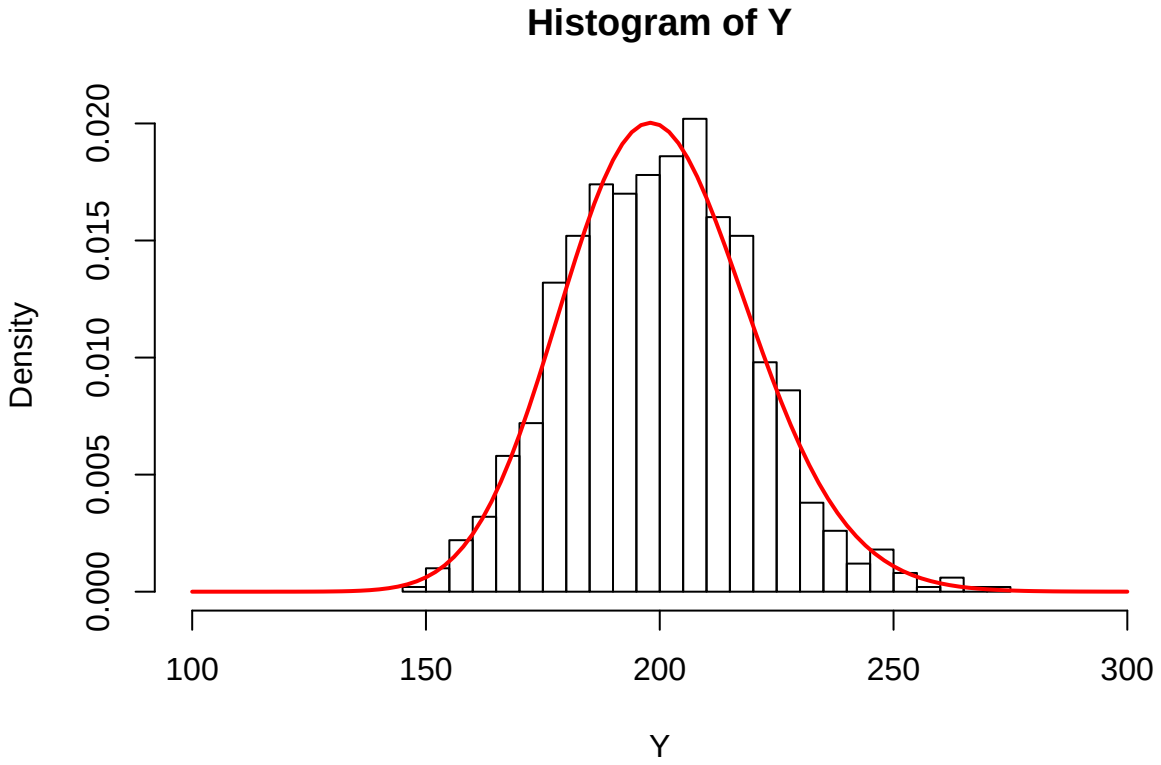
Exercise 3

- Show in R, also graphically, that Gamma$(n/2, 1/2)$ coincides with a χ^2_n.

```
n<-50          #sample size
curve(dchisq(x, n), col="red", lwd=3, xlim=c(0,40))
curve(dgamma(x, n/2, 1/2), col="black", lwd=1, add=TRUE,  xlim=c(0,40))
```



```r
n<-1000          #sample size
k<-2
sample_rep<-100 # number of distributions
X<-matrix(NA, n, sample_rep)
for (h in 1:n){
  X[h,]<-rchisq(sample_rep, k)
}
Y<-apply(X,1,sum)
hist(Y, breaks=40, probability=TRUE, xlim=c(100,300))

curve(dgamma(x, sample_rep*k/2, 1/2), col="red", lwd=2, add=TRUE)
```



- Find the 5% and the 95% quantiles of a Gamma$(3, 3)$.

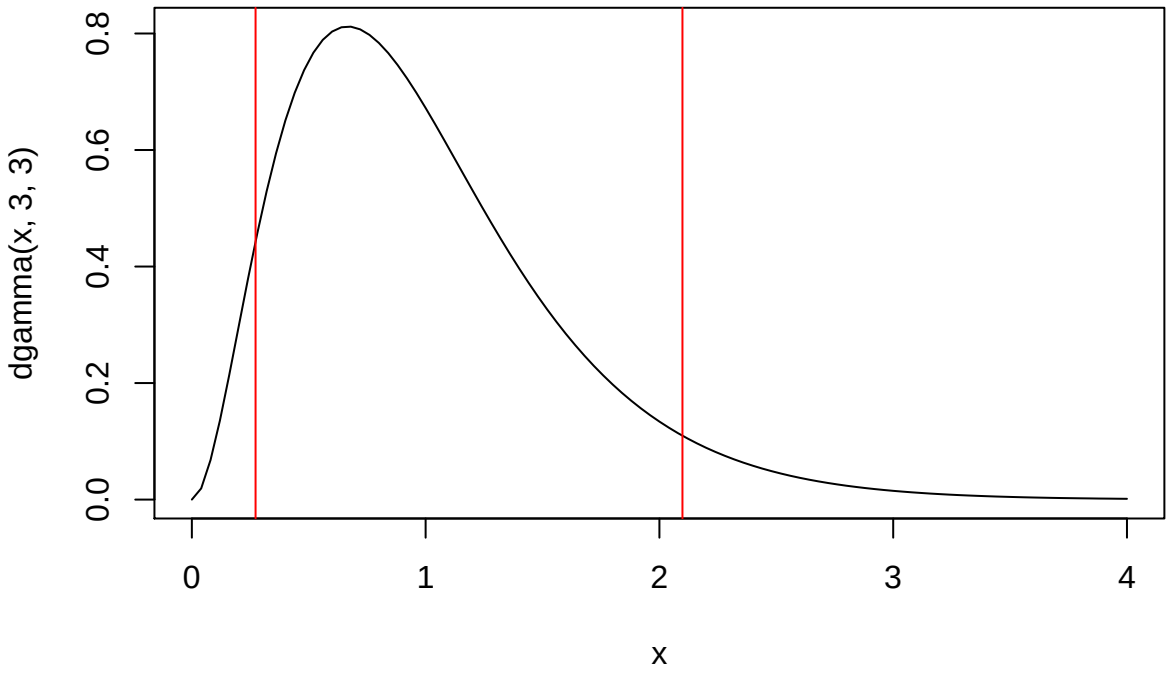
```r
q1<-qgamma(0.05,3,3)
q2<-qgamma(0.95,3,3)
q1
```

```
## [1] 0.2725638
```

```r
q2
```

```
## [1] 2.098598
```

```r
curve(dgamma(x,3,3), from=0, to=4)
abline(v=q1, col = "red")
abline(v=q2, col = "red")
```



Exercise 4

- Generate $n = 1000$ values from a Beta$(5, 2)$ and compute the sample mean and the sample variance.

```
z <- rbeta(1000, 5, 2)
mean(z)
```

```
## [1] 0.7175938
```

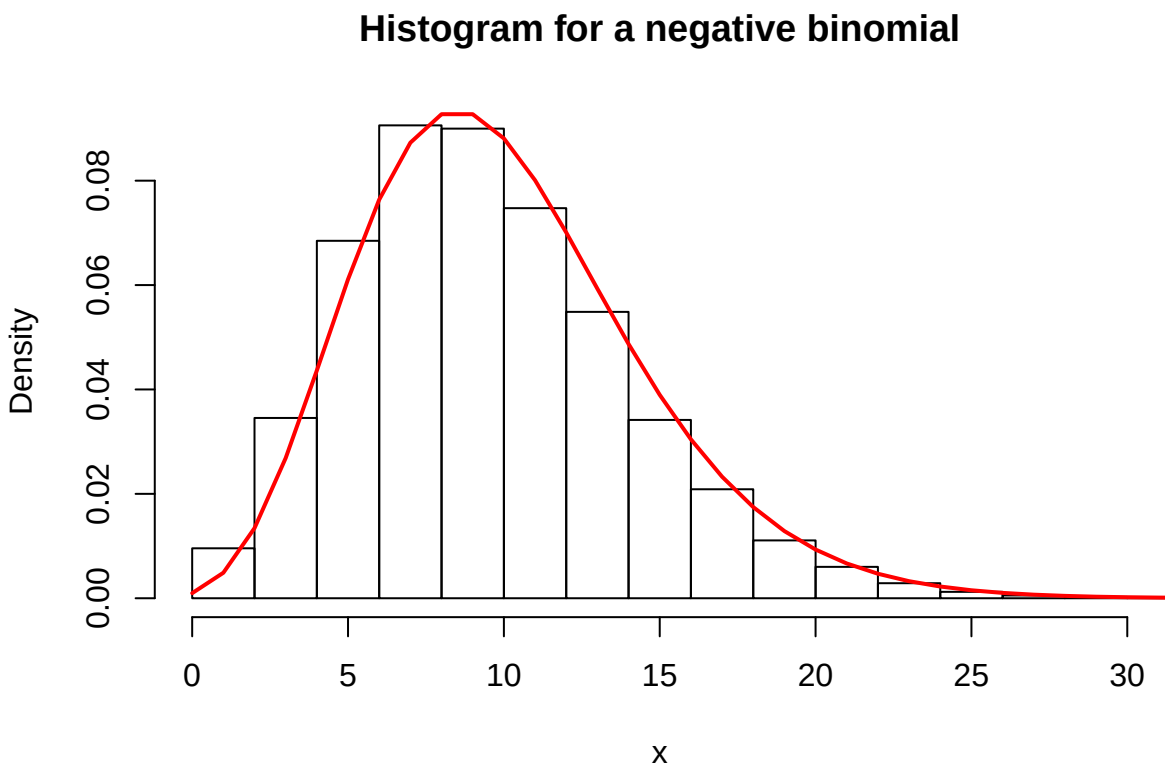
```
var(z)
```

```
## [1] 0.0264765
```

Exercise 5

- Analogously, show with a simple R function that a negative binomial distribution may be seen as a mixture between a Poisson and a Gamma. In symbols: $X|Y \sim \mathcal{P}(Y)$, $Y \sim \text{Gamma}(\alpha, \beta)$, then $X \sim \ldots$.

```
mixture <- function(r, p, n){

  Y=rgamma(n, r, (1-p)/p)
  X=rpois(n, Y)
  return(X)

}

p<-0.5
r<-10
n<-100000
hist(mixture(r, p, n), probability=TRUE, breaks=20, main="Histogram for a negative binomial", xlab="x",
curve(dnbinom(x, r, p), xlim=c(0,100), add=TRUE, col="red", lwd=2)
```

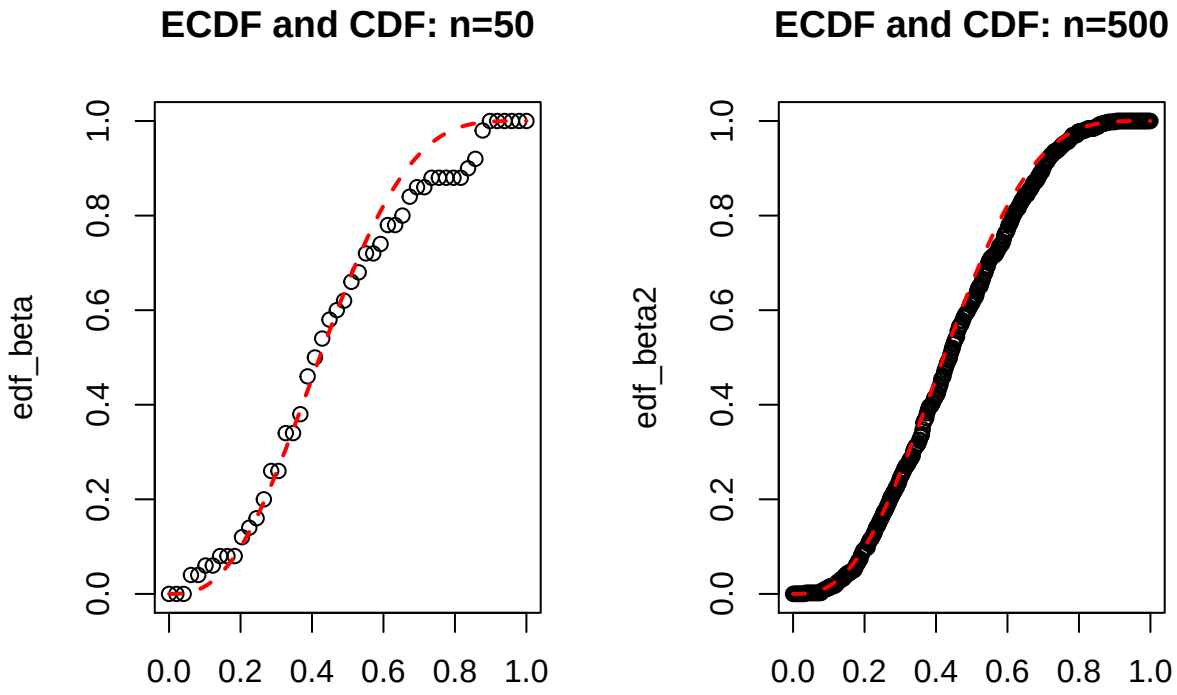


Exercise 6

- Instead of using the built-in function ecdf(), write your own R function for the empirical cumulative distribution function and reproduce the two plots above.

```
my_ecdf <- function(y){
  out <- c()
  x <- seq(0,1,1/length(y))
  for (i in 0:length(y)) {
    out[i] <- length(which(y<=x[i]))/length(y)
  }
  return(out)
}

set.seed(2)

n<-50
par(mfrow=c(1,2))
y<-rbeta(n, 3, 4)
edf_beta <- my_ecdf(y)
tt<-seq(from=0, to=1, by=0.01)
plot(x=seq(from=0, to=1, by=1/(n-1)), y=edf_beta, main="ECDF and CDF: n=50", xlab=' ')
lines(tt, pbeta(tt,3,4), col=2, lty=2, lwd=2)

n2<-500
y2<-rbeta(n2, 3,4)
edf_beta2<-my_ecdf(y2)
tt<-seq(from=0, to=1, by=0.01)
plot(x=seq(from=0, to=1, by=1/(n2-1)), edf_beta2, main="ECDF and CDF: n=500", xlab=' ')
```

```r
lines(tt, pbeta(tt,3,4), col=2, lty=2, lwd=2)
```

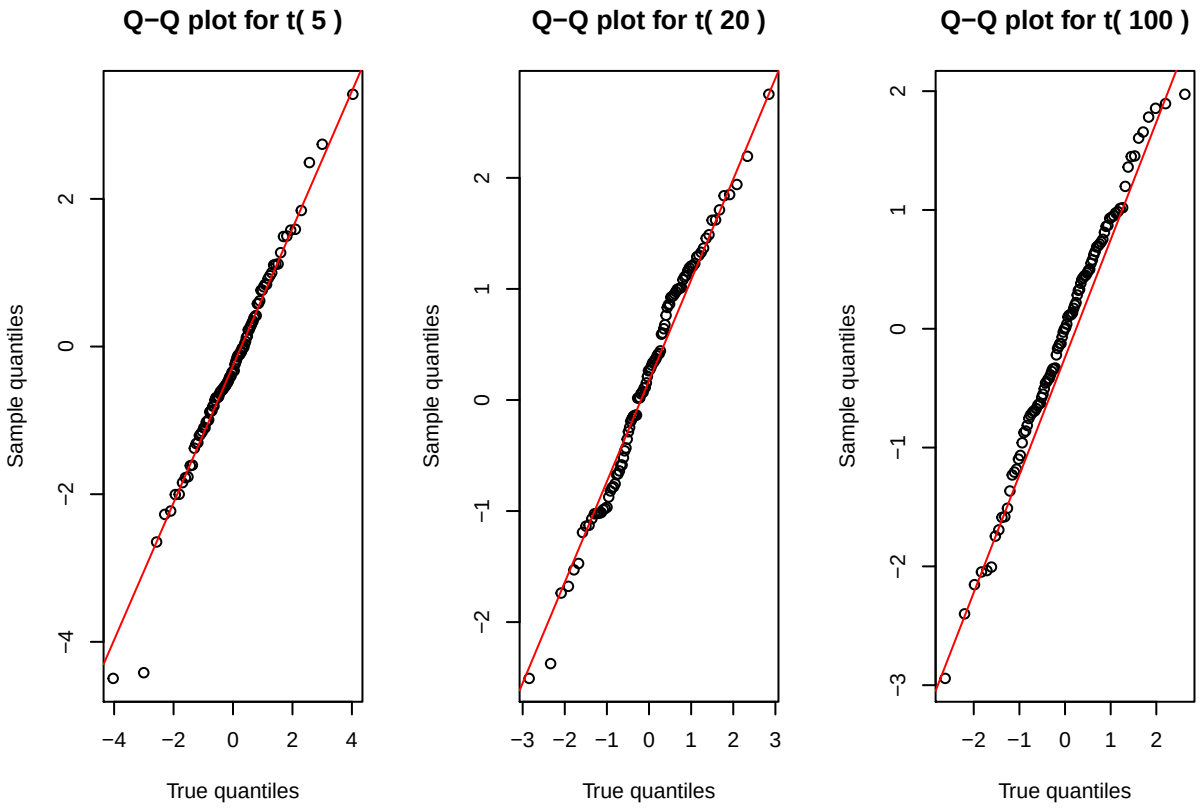


Exercise 7

Compare in R the assumption of normality for these samples:

- $y_1, \ldots, y_{100} \sim t_\nu$, with $\nu = 5, 20, 100$. What happens when the number of degrees of freedom ν increases?

```r
par(mfrow=c(1,3))
n<-100

for (v in c(5,20,100)) {
  y<-rt(n, v)
  qqplot(qt(ppoints(n),v), y,
    xlab="True quantiles", ylab="Sample quantiles",
    main = paste("Q-Q plot for t(",v,")"))
  qqline(y, distribution = function(p) qt(p, v),
       prob = c(0.1, 0.9), col = 2)
}
```

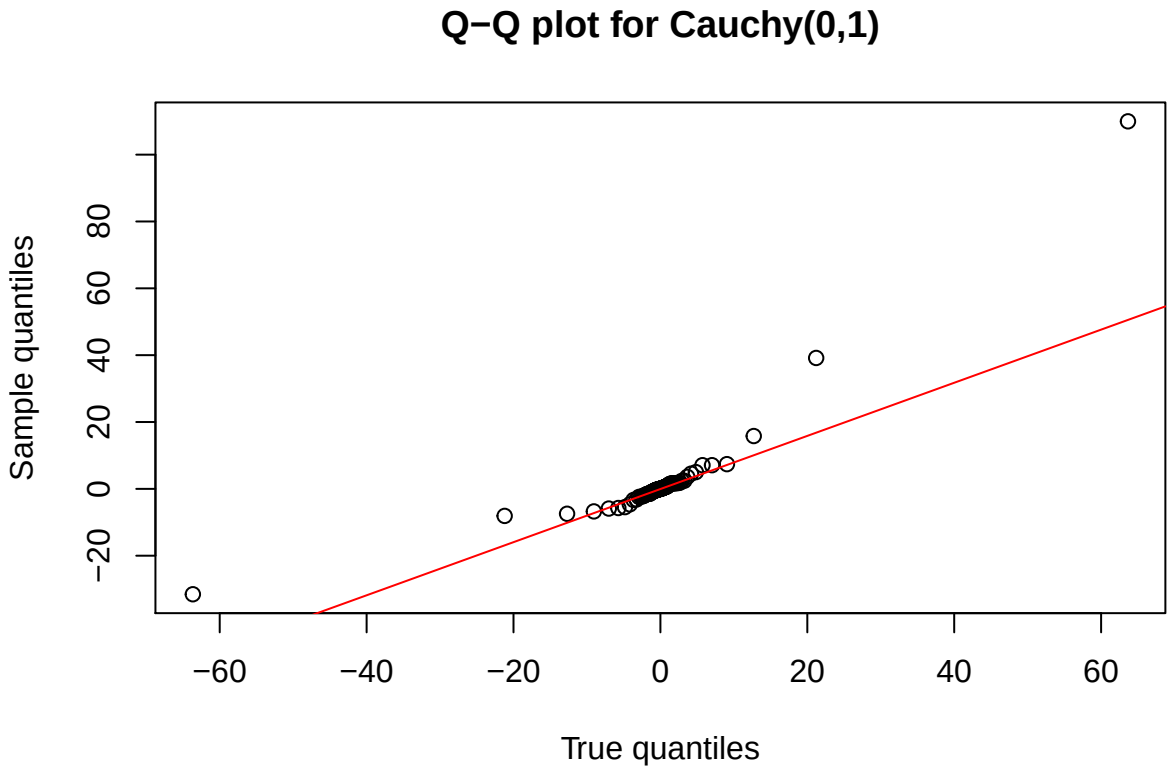



```
par(mfrow=c(1,1))
```

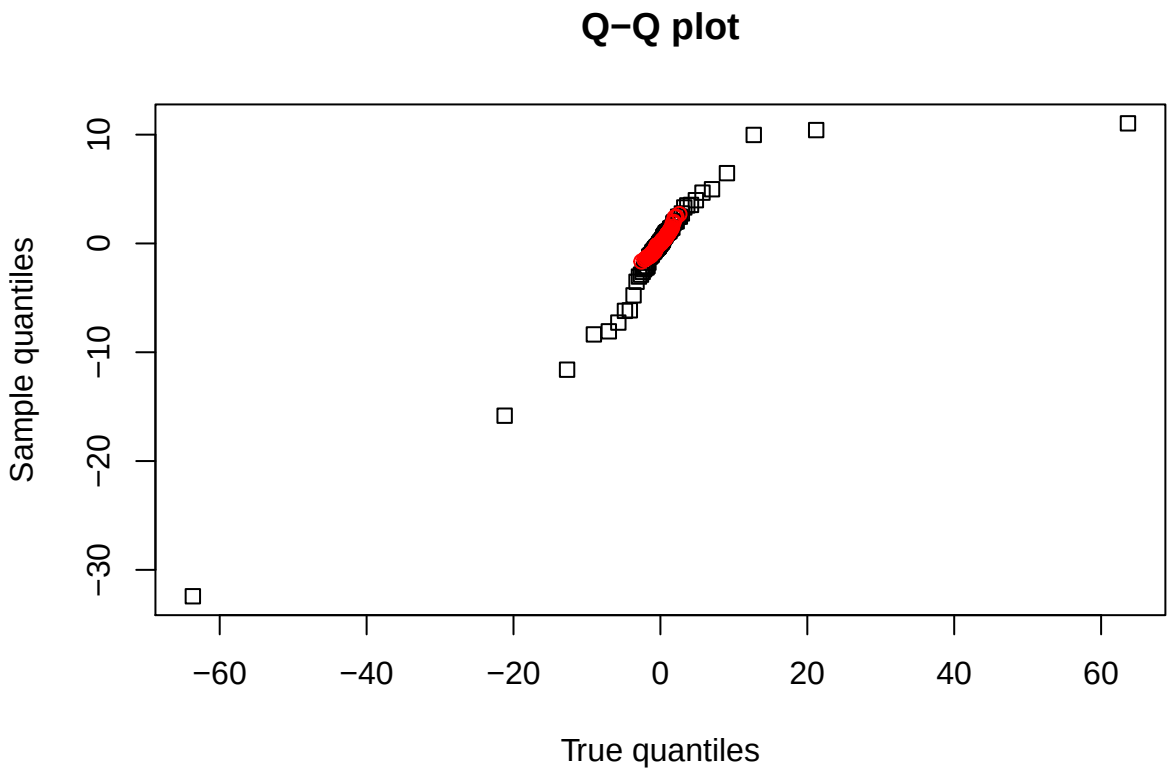
When the number of degrees of freedom increases tails become lighter and the normal approximation becomes more plausible.

- $y_1, \ldots, y_{100} \sim$ Cauchy$(0, 1)$. Do you note something weird for the extremes quantiles? Plot together a Cauchy$(0, 1)$ and a Normal$(0, 1)$ and give an intuition of this.

```
n<-100
y<-rcauchy(n, 0,1)
qqplot(qcauchy(ppoints(n),0,1), y,
  xlab="True quantiles", ylab="Sample quantiles",
  main = "Q-Q plot for Cauchy(0,1)")
qqline(y, distribution = function(p) qcauchy(p, 0,1),
       prob = c(0.1, 0.9), col = 2)
```




```
n <- 100
z1 <- rnorm(n, 0, 1)
z2 <- rcauchy(n, 0, 1)
##########
q1 <- qqplot(qcauchy(ppoints(n),0,1), rcauchy(n, 0,1), plot.it = FALSE)
q2 <- qqnorm(z1, plot.it = FALSE)
plot(range(q1$x, q2$x), range(q1$y, q2$y), type = "n", xlab="True quantiles", ylab="Sample quantiles", m
points(q1, col = "black", pch = 0)
points(q2, col = "red", pch = 1)
```

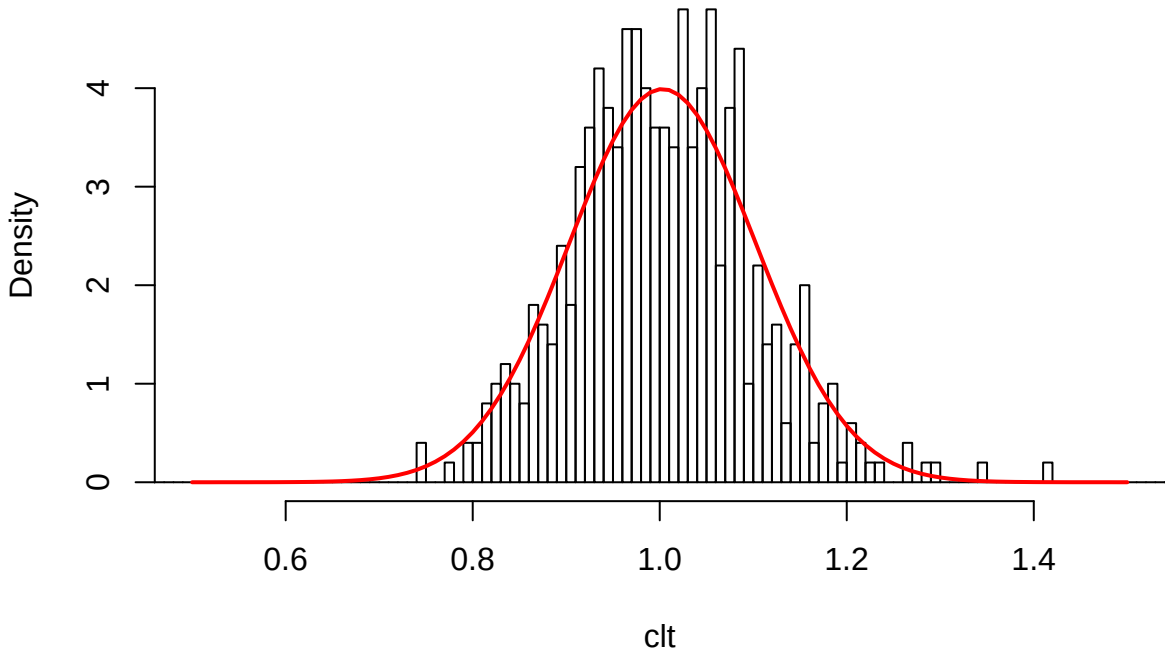


Extreme quantiles in Cauchy distribution enhance the fact that it's a fat-tailed distribution.

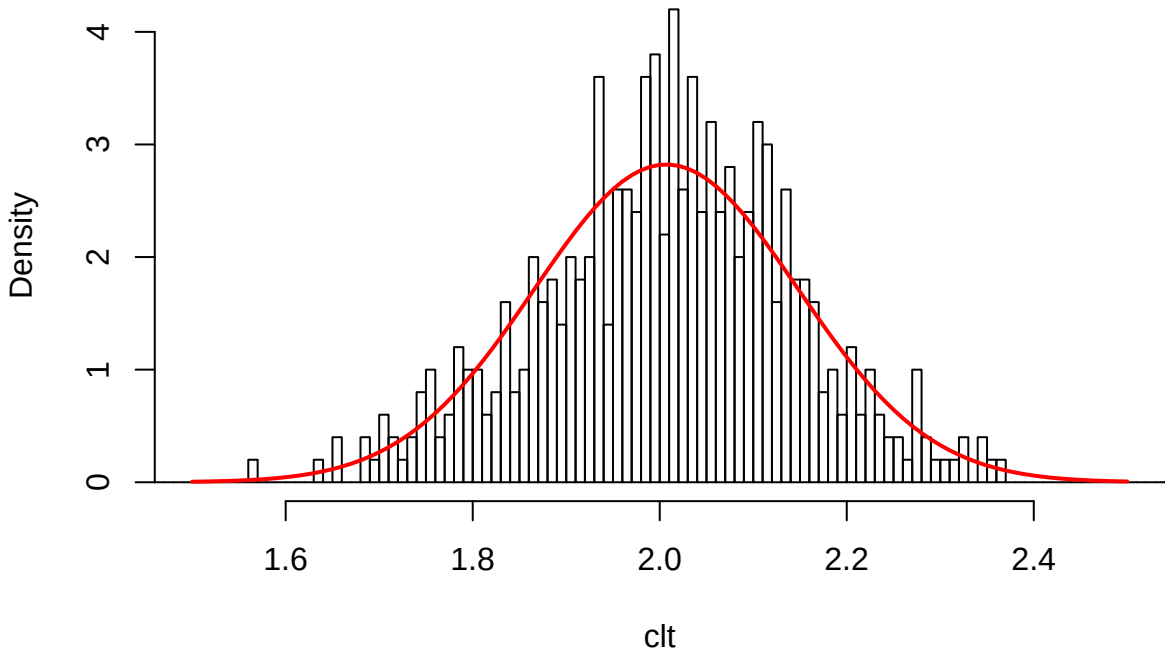
Exercise 8

Write a general R function for checking the validity of the central limit theorem. *Hint* The function will consist of two parameters: clt_function <- function(n, distr), where the first one is the sampe size and the second one is the kind of distribution from which you generate. Use plots for visualizing the results.

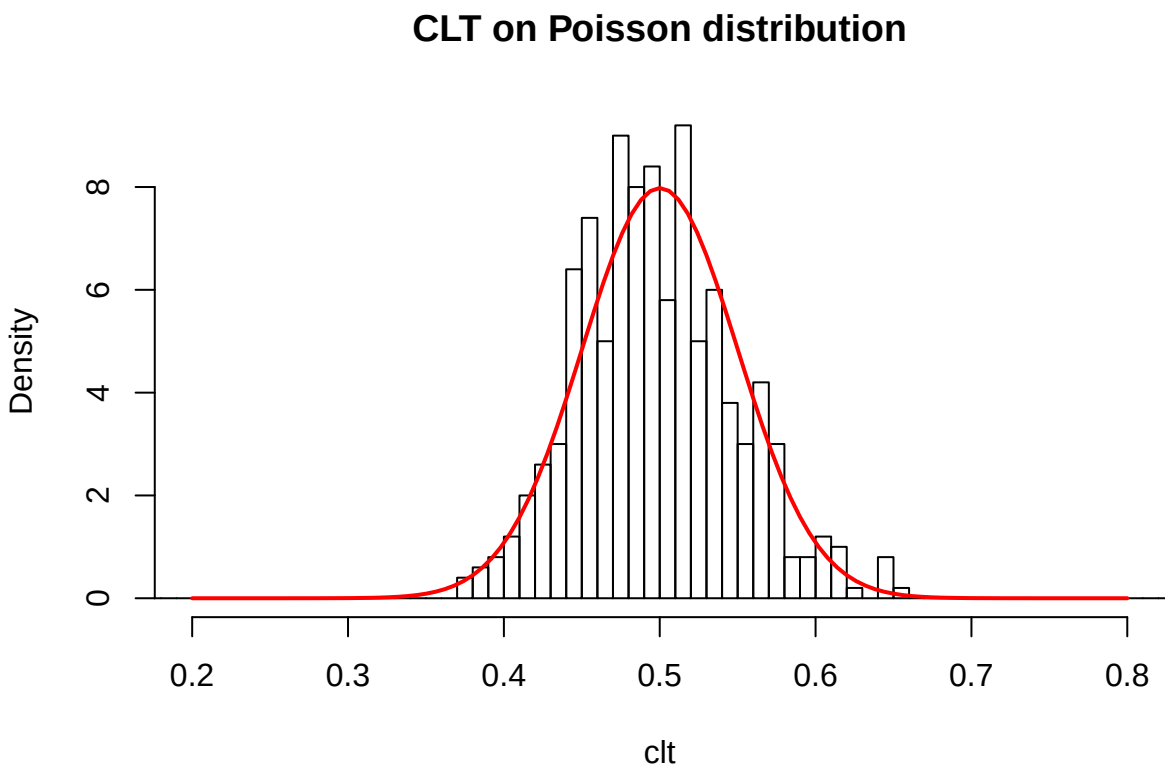
```
set.seed(123)

clt_function <- function(n,distr){
   size  = length ( distr ) / n
   sample_means = rep(0,size+1)
   for (i in 1:size) {
    sample_means[i] = mean ( distr[(i*n + 1):(i*n + n)] )
   }
  return(sample_means)
}

sample_size=100000
n=200
distr = rchisq ( sample_size , df = 1 ,ncp=0)
clt <- clt_function(n, distr)
hist(clt, breaks = c(seq(from = 0, to = 10, by = 0.01)),xlim = c(0.5,1.5),probability = TRUE, main="CLT
curve(dnorm(x,mean=mean(distr),sqrt(var(distr)/n)),add=TRUE,lwd=2,col="red")
```




```
sample_size=100000
n=200
distr = rgamma ( sample_size , 1, 0.5)
clt <- clt_function(n, distr)
hist(clt, breaks = c(seq(from = 0, to = 10, by = 0.01)),xlim = c(1.5,2.5),probability = TRUE, main="CLT
curve(dnorm(x,mean=mean(distr),sqrt(var(distr)/n)),add=TRUE,lwd=2,col="red")
```




```
sample_size=100000
n=200
distr = rpois (sample_size , 0.5)
clt <- clt_function(n, distr)
hist(clt, breaks = c(seq(from = 0, to = 10, by = 0.01)),xlim = c(0.2,0.8),probability = TRUE,main="CLT
curve(dnorm(x,mean=mean(distr),sqrt(var(distr)/n)),add=TRUE,lwd=2,col="red")
```


CLT on Poisson distribution
Density
0
2
4
6
8
0.2
0.3
0.4
0.5
0.6
0.7
0.8
clt