Consider the Boston housing data set, from the MASS library. Consider column `medv` as a sample of the population prices of houses in the Boston area.

a) Use `t.test(d0$medv)` to find a 95% confidence interval for the mean of `medv`. Provide an estimate $\hat{\mu}$ for the population mean $\mu$ and of the standard deviation of that estimate $\hat{\mu}$

b) Now estimate that standard deviation using the bootstrap.

c) Based on your bootstrap estimate, find a 95% confidence interval for the mean of `medv`. Compare it to the results obtained using `t.test(d0$medv)`.

d) Provide an estimate, $\hat{\mu}_{med}$, for the median of `medv` in the population.

e) Estimate the standard error of $\hat{\mu}_{med}$. Unfortunately, there is no simple formula for computing the standard error of the median. Instead, estimate the standard error of the median using the bootstrap.

f) Estimate the population tenth percentile $x_{0.1}$ of `medv`. Find the bootstrap estimate $\hat{x}_{0.1}$ of $x_{0.1}$ and its standard error.

```
# 5_9.r      James, p.201

library(MASS)    # Boston data
library(boot)    # boot()  boot.ci()

d0 = Boston
y = d0$medv

setwd("C:/Users/USC Guest/Downloads")  # save figures

# normality?
hist(y)

boxplot(y)

qqnorm(y)
qqline(y)
grid()

# classical estimate std error - MEAN
#=====================================================================
n = nrow(d0)
mean(y)              # [1] 22.53281
sd(y)/sqrt(n)        # [1] 0.4088611

aux=t.test(y,conf.level=0.95)
aux
#        One Sample t-test
# data:  y
# t = 55.111, df = 505, p-value < 2.2e-16
# alternative hypothesis: true mean is not equal to 0
# 95 percent confidence interval:
# 21.72953 23.33608
# sample estimates:
# mean of x
# 22.53281

t.test(y,conf.level=0.95)$conf
t.test(y,conf.level=0.95)$conf[1:2]  # [1] 21.72953 23.33608

# Bootstrap estimate and bootstrap std error - MEAN
#=====================================================================

bfunction1 = function(data,index)  mean(data[index])   # index are rows used to find mean()
bfunction1(y,1:n)      # 22.53281
bfunction1(y,1:n/2)    # 24.29703
```

```
# 1000 bootstrap samples
set.seed(1)
boot(y, bfunction1, 1000)

# ORDINARY NONPARAMETRIC BOOTSTRAP
# Bootstrap Statistics :
#     original      bias    std. error
# t1* 22.53281 0.008517589   0.4119374

aux = boot(y, bfunction1, 1000)
str(aux)
# List of 11
# $ t0       : num 22.5
# $ t        : num [1:1000, 1] 22.5 22.3 22 22.6 23.1 ...
# $ R        : num 1000
# $ data     : num [1:506] 24 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 ...
# $ seed     : int [1:626] 403 624 -169270483 -442010614 -603558397 -222347416 1489374793 86587122
# $ statistic:function (data, index)

# aux$t        the B resampling statistics
# mean(aux$t)  mean of resampling stats (not reported)
# sd(aux$t)    sd   of resampling stats (reported)

# resampling stats
head(aux$t)
         [,1]
[1,] 22.53182
[2,] 22.34328
[3,] 21.97055
[4,] 22.61976
[5,] 23.10079
[6,] 23.22885

# bootstrap mean, sd, bias

mean(aux$t)   # [1] 22.54132    # bootstrap mean (not reported)
sd(aux$t)     # [1] 0.4119374   # sd
mean(aux$t)-mean(y)             # bias
# [1] 0.008517589

# CI on mean
b1=boot(y, bfunction1, 1000)
boot.ci(b1,conf=0.95,type="basic")

# BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
# Based on 1000 bootstrap replicates
# Intervals :
# Level      Basic
# 95%   (21.67, 23.28)
```

```
# Bootstrap estimate and bootstrap std error - MEDIAN
#=====================================================================
median(y)              # [1] 21.2

bfunction2 = function(data,index) median(data[index])
bfunction2(y,1:n)     # [1] 21.2
bfunction2(y,1:n/2)   # [1] 22.5

set.seed(1)
boot(y,bfunction2,1000)
# ORDINARY NONPARAMETRIC BOOTSTRAP
# Bootstrap Statistics :
#     original  bias    std. error
# t1*     21.2 -0.0025    0.374358

aux = boot(y,bfunction2,1000)
mean(aux$t)
# 21.1975
sd(aux$t)
# 0.374358
mean(aux$t)-median(y)    # bias
# -0.0025

# CI on median
b2=boot(y, bfunction2, 1000)
boot.ci(b2,conf=0.95,type="basic")
# BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
# Based on 1000 bootstrap replicates
# Intervals :
# Level      Basic
# 95%   (20.55, 21.95 )
```

```
# Bootstrap estimate and bootstrap std error - quantile
#======================================================================
quantile(y,probs=c(0.1))   # 12.75

bfunction3 = function(data,index) quantile(data[index],probs=c(0.1))
bfunction3(y,1:n)
#  10%
#12.75
bfunction3(y,1:n/2)
# 10%
#15.6

set.seed(1)
boot(y,bfunction3,1000)
# ORDINARY NONPARAMETRIC BOOTSTRAP
# Bootstrap Statistics :
#    original  bias    std. error
#t1*    12.75 0.01005    0.505056

# CI on quantile
b3=boot(y,bfunction3, 1000)
boot.ci(b3,conf=0.95,type="basic")
# BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
# Based on 1000 bootstrap replicates
# Intervals :
# Level      Basic
# 95%   (12.05, 13.75 )
```

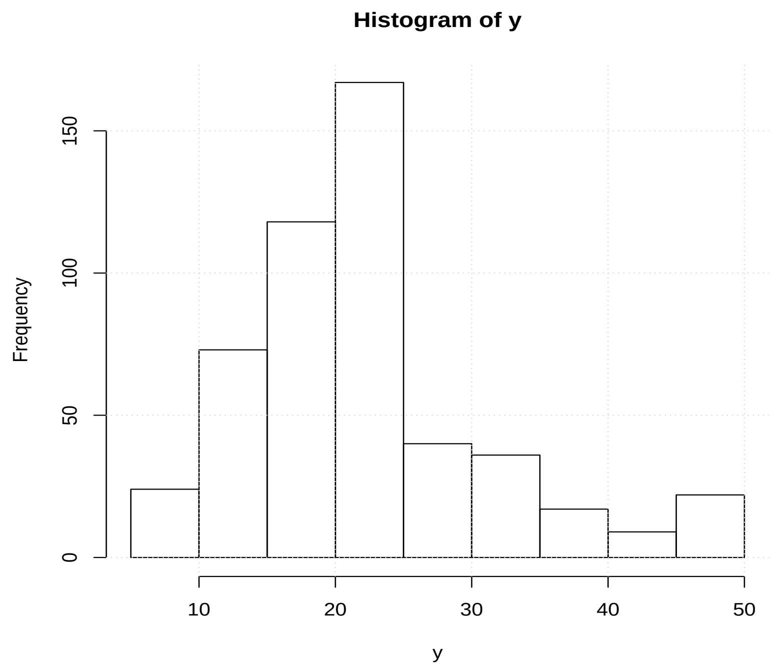


Figure 1: Histogram for prices

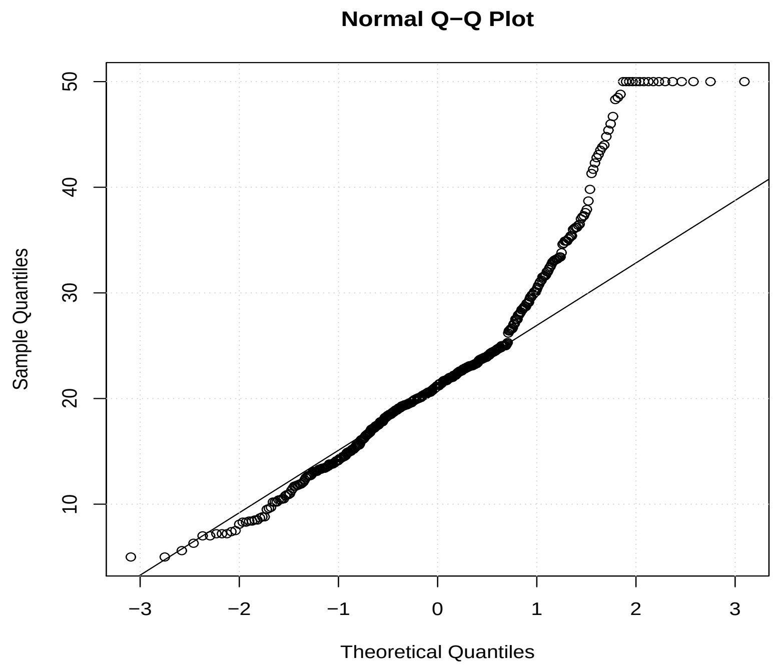


Figure 2: Qqplot for prices