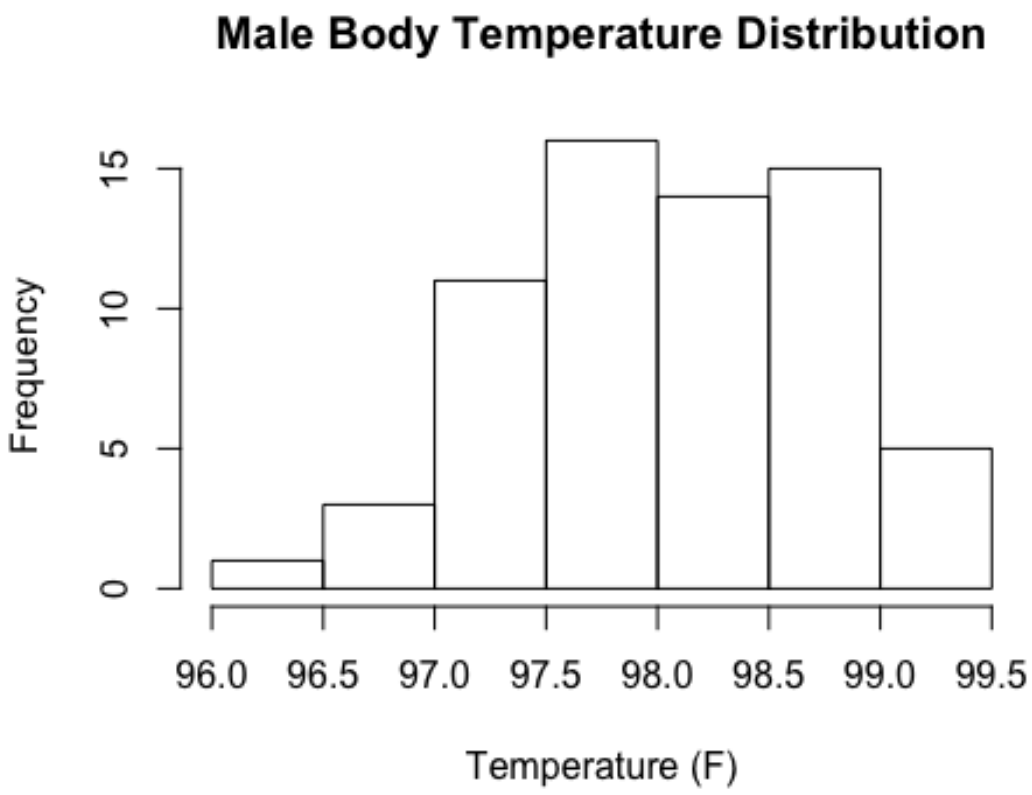
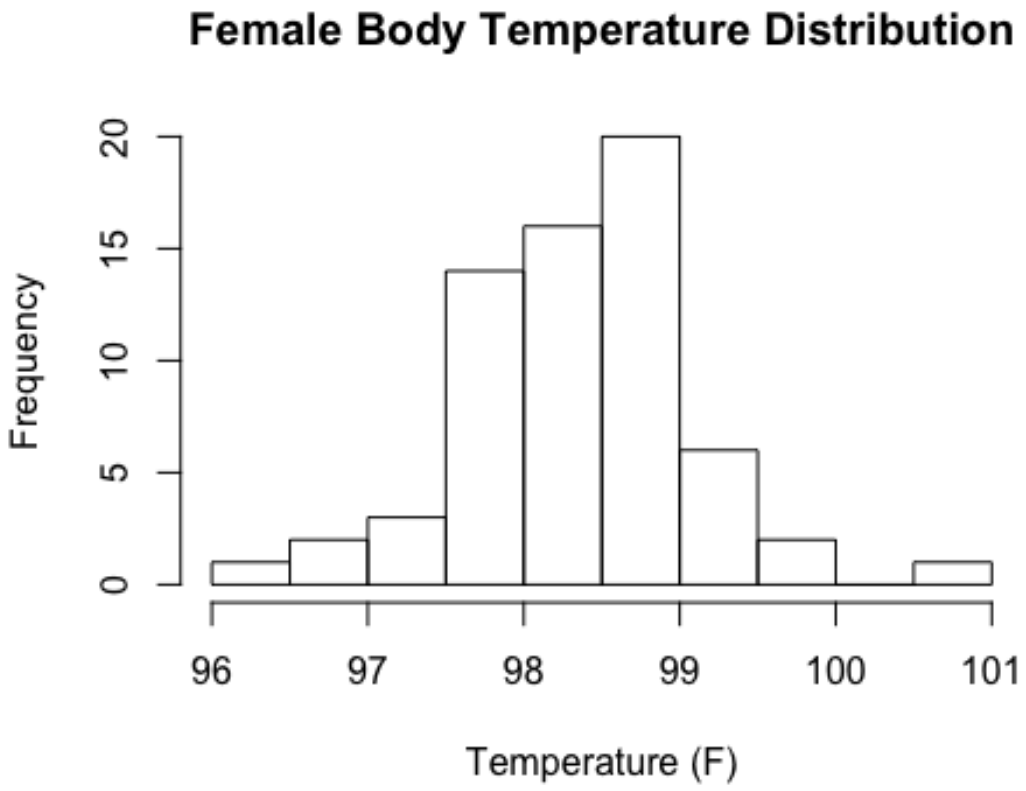
Correlation

Part 1: correlation using linear model.

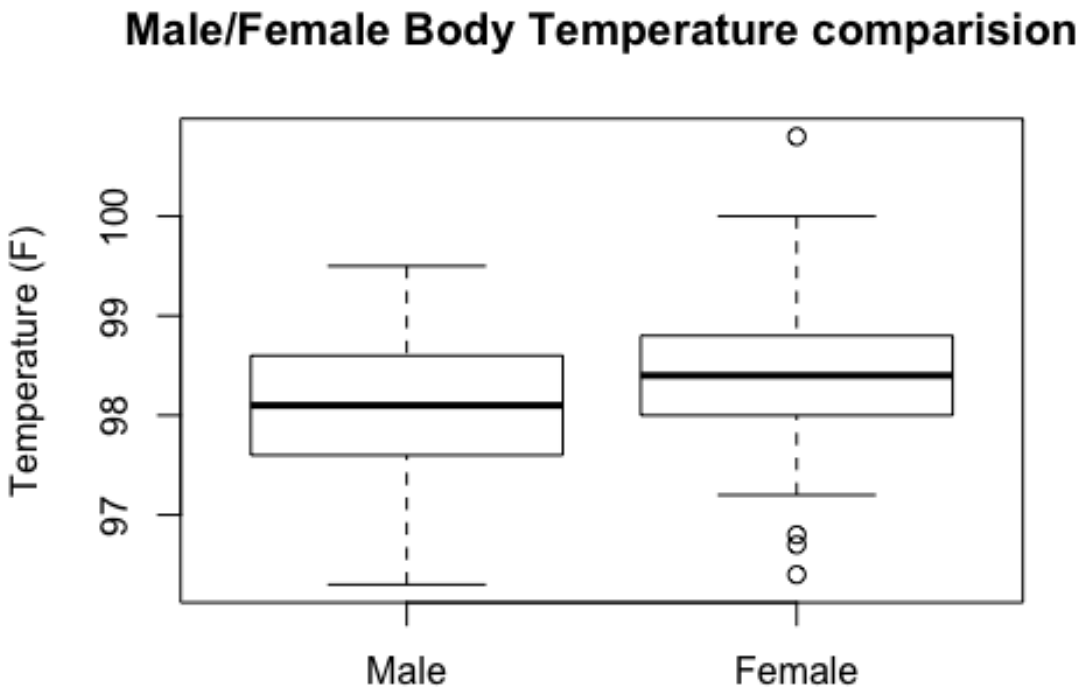
```
file1 <- "bodytemp-heartrate.csv"
data1 <- read.csv(file1,header = T)
# extract male and female data
male <- data1[ which(data1$gender==1), ]
female <- data1[ which(data1$gender==2), ]

# summary statistics function
new.summary <- function(x){
  result1 <- summary(x)
  result2 <- c(result1[-4], IQR = IQR(x), result1[4], SD = sd(x))
  return(result2)
}

# (a): male vs female body temperature
TempRange = c(min(data1$body_temperature),max(data1$body_temperature))
male.temp = male$body_temperature
female.temp = female$body_temperature
hist(male.temp, main="Male Body Temperature Distribution",
     #xlim=TempRange,
     xlab="Temperature (F)",breaks=10)
```



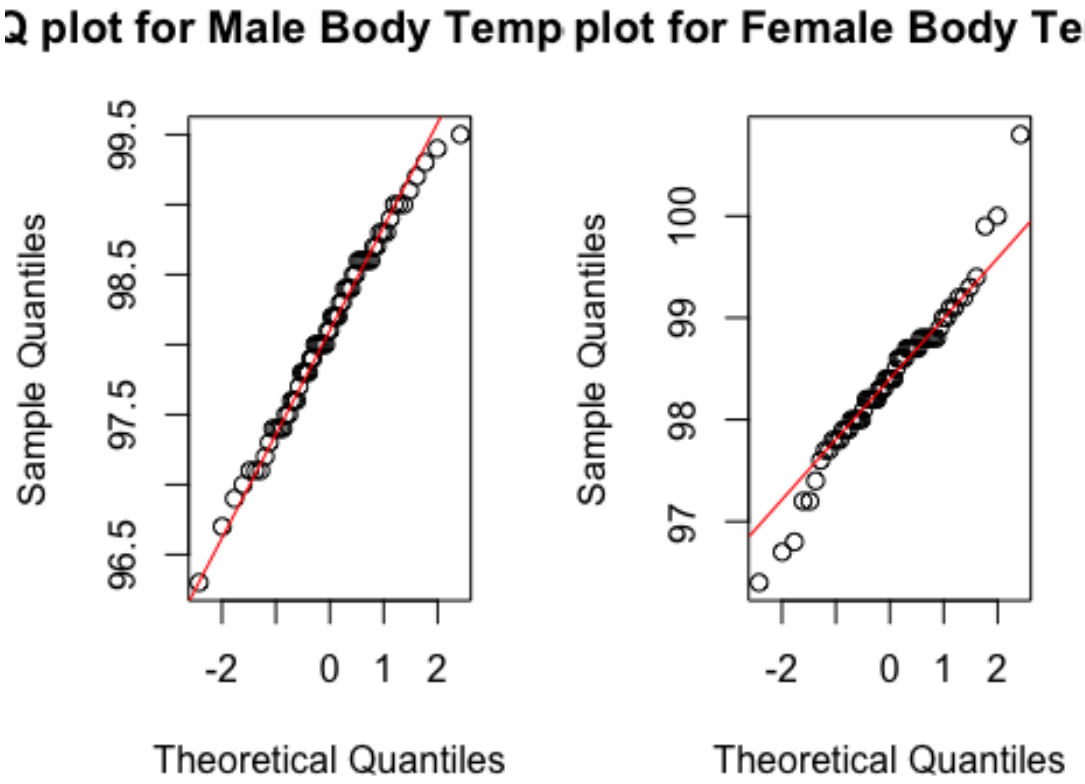

```r
hist(female.temp, main="Female Body Temperature Distribution",
     #xlim=TempRange,
     xlab="Temperature (F)",breaks=10)
```



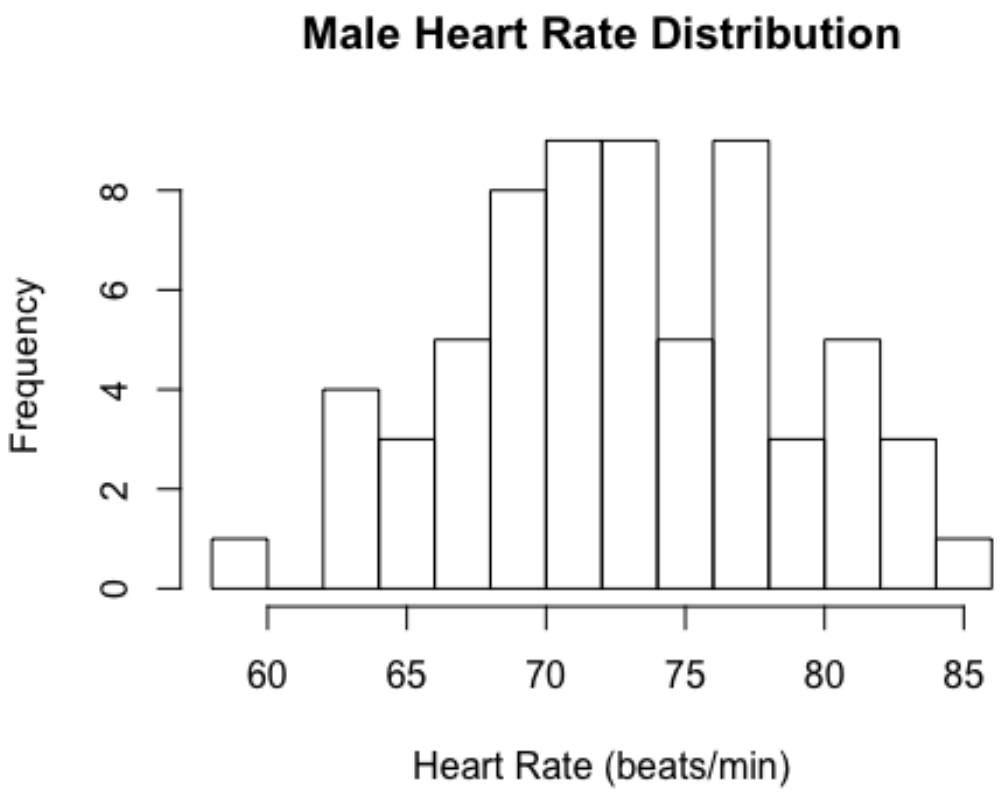

```
boxplot(male.temp,female.temp, main="Male/Female Body Temperature
comparision",
        names=c("Male","Female"), ylab="Temperature (F)")
```



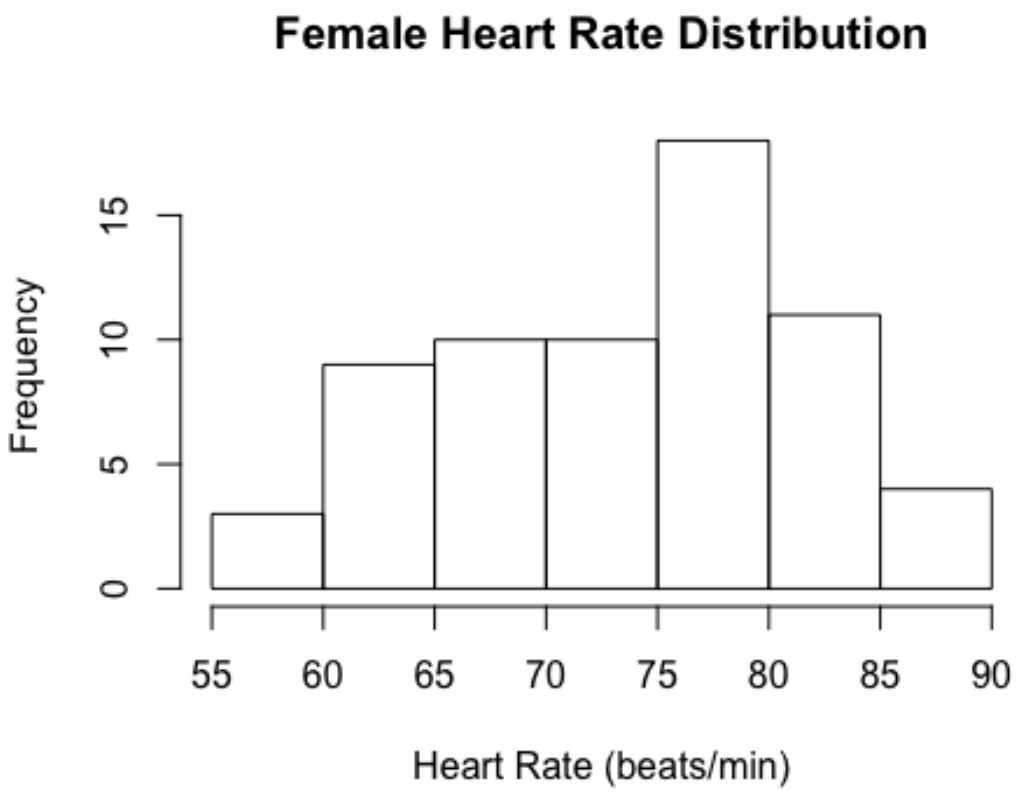

```
round(new.summary(male.temp),3)

##    Min. 1st Qu.  Median 3rd Qu.    Max.     IQR    Mean      SD
##  96.300  97.600  98.100  98.600  99.500   1.000  98.105   0.699

round(new.summary(female.temp),3)

##    Min. 1st Qu.  Median 3rd Qu.    Max.     IQR    Mean      SD
##  96.400  98.000  98.400  98.800 100.800   0.800  98.394   0.743

par(mfrow=c(1,2))
qqnorm(male.temp, main="Q-Q plot for Male Body Temperature")
qqline(male.temp,col="red")
qqnorm(female.temp, main="Q-Q plot for Female Body Temperature")
qqline(female.temp,col="red")
```



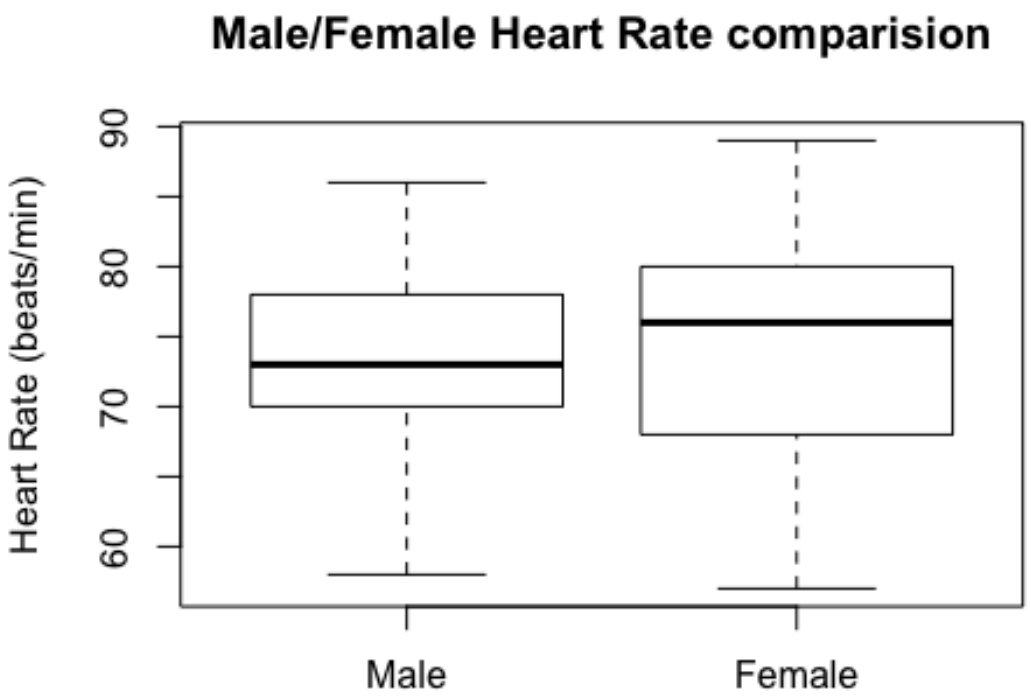

```r
par(mfrow=c(1,1))

# perform T-test to get p-value and confidence interval
r1 <- t.test(male.temp, female.temp, alternative="two.sided")
r1$p.value

## [1] 0.02393826

r1$conf.int

## [1] -0.53964856 -0.03881298
## attr(,"conf.level")
## [1] 0.95

# (b): male vs female heart rate
HRRange = c(min(data1$heart_rate),max(data1$heart_rate))
male.hr = male$heart_rate
female.hr = female$heart_rate
hist(male.hr, main="Male Heart Rate Distribution",
     #xlim=HRRange,
     xlab="Heart Rate (beats/min)",breaks=10)
```



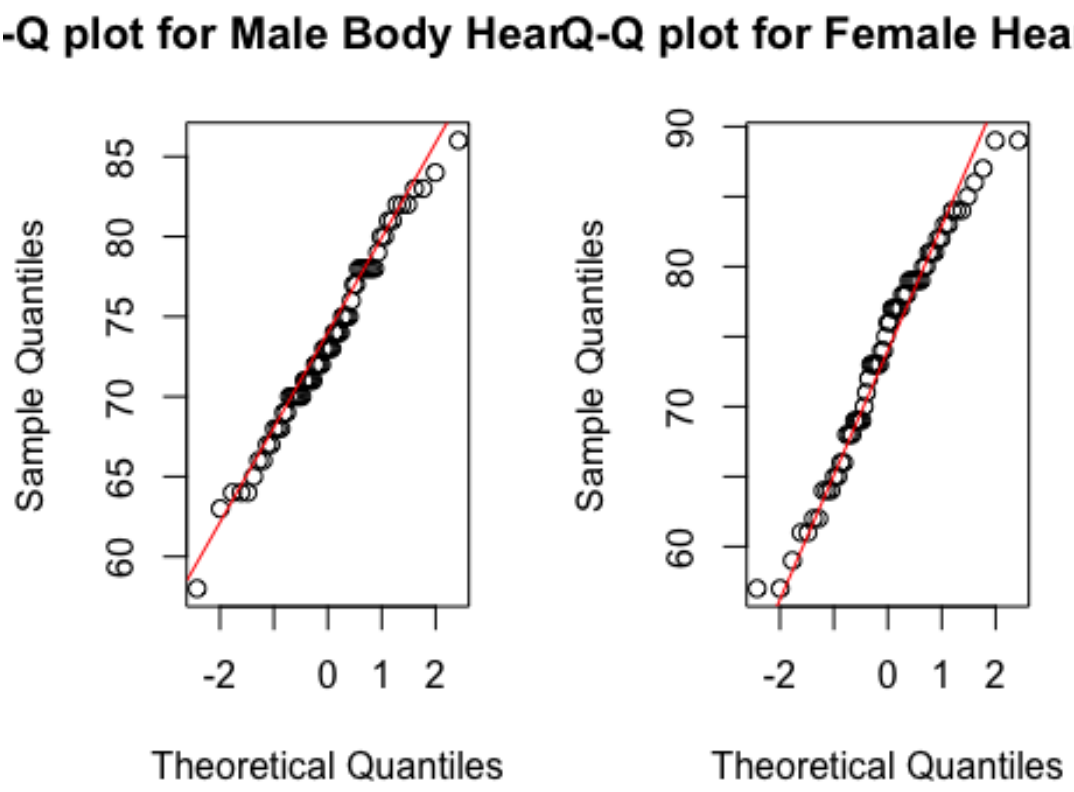

```r
hist(female.hr, main="Female Heart Rate Distribution",
     #xlim=HRRange,
     xlab="Heart Rate (beats/min)",breaks=10)
```



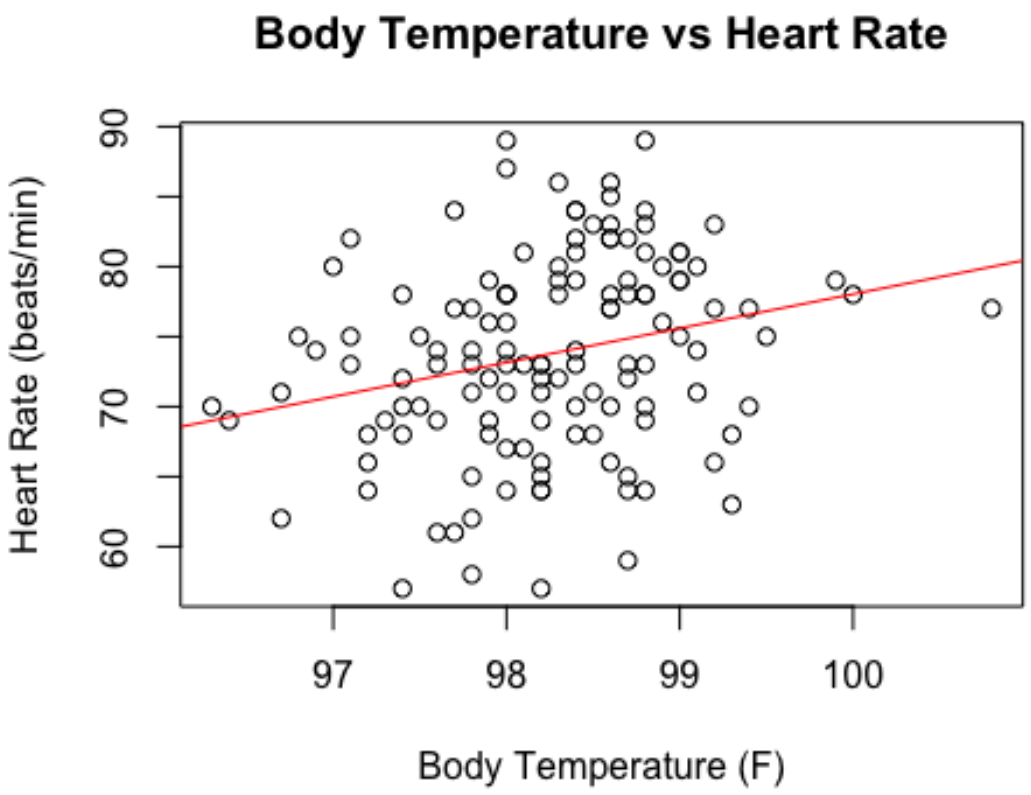

```
boxplot(male.hr,female.hr, main="Male/Female Heart Rate comparision",
        names=c("Male","Female"), ylab="Heart Rate (beats/min)")
```




```
round(new.summary(male.hr),3)

##    Min. 1st Qu.  Median 3rd Qu.    Max.     IQR    Mean      SD
##  58.000  70.000  73.000  78.000  86.000   8.000  73.369   5.875

round(new.summary(female.hr),3)

##    Min. 1st Qu.  Median 3rd Qu.    Max.     IQR    Mean      SD
##  57.000  68.000  76.000  80.000  89.000  12.000  74.154   8.105

par(mfrow=c(1,2))
qqnorm(male.hr, main="Q-Q plot for Male Body Heart Rate")
qqline(male.hr,col="red")
qqnorm(female.hr, main="Q-Q plot for Female Heart Rate")
qqline(female.hr,col="red")
```

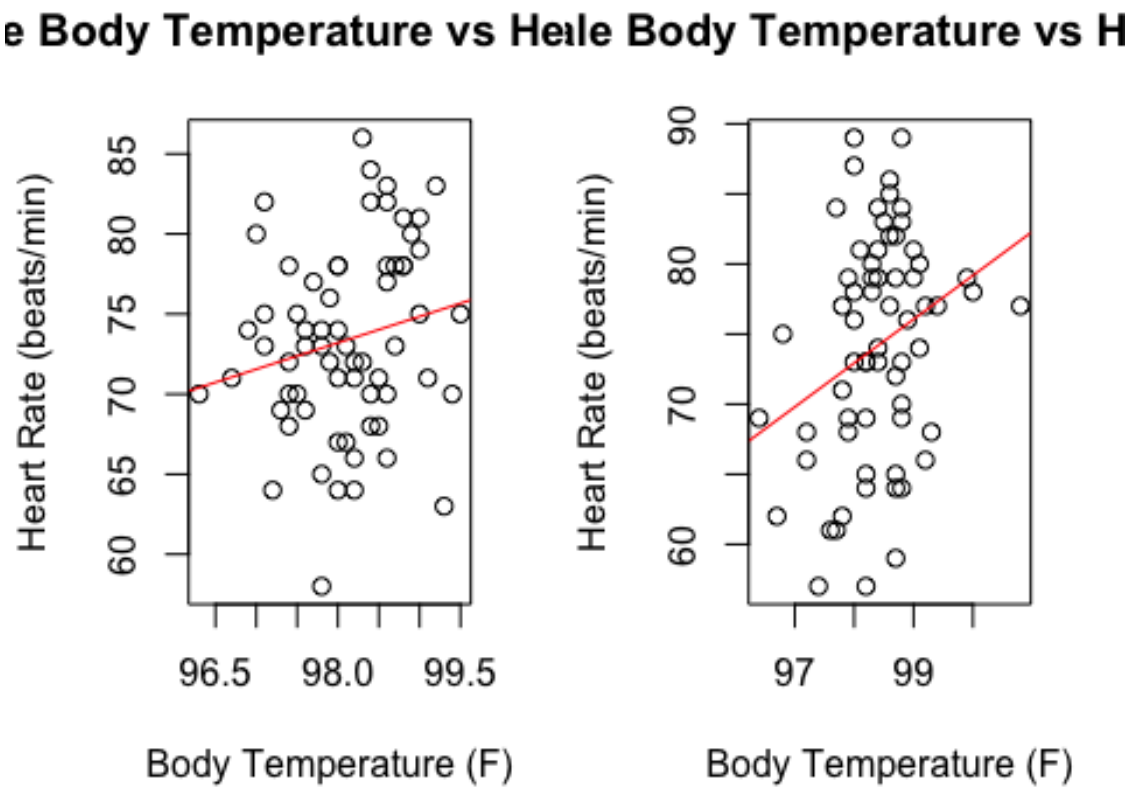



```r
par(mfrow=c(1,1))

# perform T-test to get p-value and confidence interval
r2 <- t.test(male.hr, female.hr, alternative="two.sided")
r2$p.value

## [1] 0.5286842

r2$conf.int

## [1] -3.243732  1.674501
## attr(,"conf.level")
## [1] 0.95

# (c): body temperature vs heart rate among male and female
bodytemp = data1$body_temperature
heartrate = data1$heart_rate
plot(bodytemp, heartrate,
     main="Body Temperature vs Heart Rate",
     xlab="Body Temperature (F)", ylab="Heart Rate (beats/min)")
abline(lm(heartrate~bodytemp), col="red") # regression line
```




```
# Correlation
cor(bodytemp,heartrate)

## [1] 0.2536564

# Get the fitted regression line
(body.reg <- lm (heartrate ~ bodytemp))

##
## Call:
## lm(formula = heartrate ~ bodytemp)
##
## Coefficients:
## (Intercept)     bodytemp
##    -166.285        2.443

male.bodytemp = data1[ which(data1$gender==1), ]$body_temperature
male.heartrate = data1[ which(data1$gender==1), ]$heart_rate
par(mfrow=c(1,2))
plot(male.bodytemp, male.heartrate,
     main="Male Body Temperature vs Heart Rate",
     xlab="Body Temperature (F)", ylab="Heart Rate (beats/min)")
abline(lm(male.heartrate~male.bodytemp), col="red")
```

```r
# Correlation
cor(male.bodytemp, male.heartrate)

## [1] 0.1955894

# Get the fitted regression line
(male.body.reg <- lm (male.heartrate ~ male.bodytemp))

##
## Call:
## lm(formula = male.heartrate ~ male.bodytemp)
##
## Coefficients:
##   (Intercept)  male.bodytemp
##       -87.967          1.645

female.bodytemp = data1[ which(data1$gender==2), ]$body_temperature
female.heartrate = data1[ which(data1$gender==2), ]$heart_rate
plot(female.bodytemp, female.heartrate,
     main="Female Body Temperature vs Heart Rate",
     xlab="Body Temperature (F)", ylab="Heart Rate (beats/min)")
abline(lm(female.heartrate~female.bodytemp), col="red")
```



```r
par(mfrow=c(1,1))
# Correlation
cor(female.bodytemp, female.heartrate)
```

```
## [1] 0.2869312
```

```r
# Get the fitted regression line
(female.body.reg <- lm (female.heartrate ~ female.bodytemp))
```

```
##
## Call:
## lm(formula = female.heartrate ~ female.bodytemp)
##
## Coefficients:
##     (Intercept)  female.bodytemp
##         -233.624            3.128
```

Part 2: correlation using paramatric bootstrap.

```r
L <- c(0.01, 0.1, 1)
N <- c(5, 10, 30, 100)

library(boot)
# function to compute coverage probabilities for a given (n, lambda)
cov.prob.fun <- function(n, l, nsim = 500) {

  # function to compute coverage
  acc.cover <- function(mu, low.bound, up.bound){
    return(mu >= low.bound && mu <= up.bound)
  }

  # function to compute parametric bootstrap
  boot.mean <- function(x){
    # get E(lambda) from given data
    mle.l <- 1/mean(x)

    # use E(lambda) to generate random exponential data
    # then, set xbar of this bootstrap to average of new data
    xbar.star <- mean(rexp(length(x), mle.l))
    return(xbar.star)
  }

  # function to simulate one sample and compute the two estimates
  sim.fun <- function(n, l) {
    # true mean
    mu = 1/l

    # simulate n sample data
    x <- rexp(n, rate=l)
```

```r
    # z interval method
    # calculate confidence interval from sample data
    z <- t.test(x, alternative="two.sided", mu=mu)

    # bootstrap method
    # use sample data to generate E(lambda)
    # next, bootstrap 99 times
    # then calculate confidence interval using bootstrap sample
    boot.rep <- 99
    mean1.boot <- boot(x, boot.mean, R = boot.rep, sim="parametric")
    bci <- boot.ci(mean1.boot, conf = 0.95, type = "perc")

    # use true mean and confidence interval to get coverage accuracy
    ztest <- acc.cover(mu, z$conf.int[1], z$conf.int[2])
    bootstrap <- acc.cover(mu, bci$percent[4], bci$percent[5])

    return(c(ztest = ztest, bootstrap = bootstrap))
  }

  # replicate nsim = 500 times
  est <- replicate(nsim, sim.fun(n, l))

  # compute coverage probabilities of all 5000 estimates
  return(rowMeans(est))

}

# create matrix to store coverage probabilities
interval1 <- matrix(NA, nrow = length(N), ncol = length(L))
interval2 <- matrix(NA, nrow = length(N), ncol = length(L))

# loop through each N and L to calculate coverage probability for each
interval
for (i in 1:length(N)) {
  for (j in 1:length(L)) {
    result <- cov.prob.fun(N[i], L[j])
    interval1[i, j] <- result["ztest"]
    interval2[i, j] <- result["bootstrap"]
  }
}

# change matrix names for ease of use
rownames(interval1) <- N
colnames(interval1) <- L
rownames(interval2) <- N
colnames(interval2) <- L

# how much better is interval 2
```

```
interval.diff <- interval2-interval1
interval.diff

##       0.01   0.1     1
## 5    0.020 0.014 0.010
## 10   0.022 0.030 0.016
## 30   0.014 0.018 0.018
## 100 -0.004 0.014 0.000
```