STAC67 A1

2026-01-14

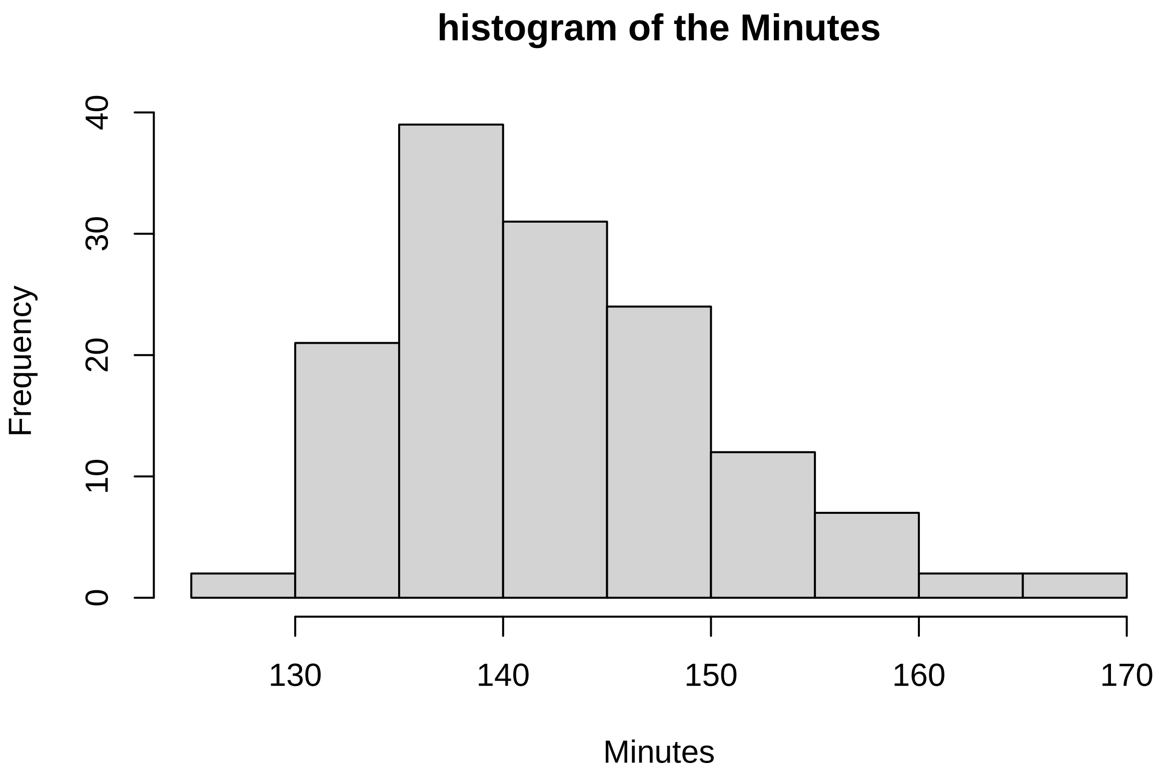
Q1

(a)

```r
csvFile = "OlympicMarathon2016.csv"
df = read.csv(csvFile)
mu = mean(df[["Minutes"]])
sigma2 = var(df[["Minutes"]])
sprintf("Minutes has mean %f and variance %f", mu, sigma2)
```

```
## [1] "Minutes has mean 142.367143 and variance 59.638923"
```

```r
hist(df[["Minutes"]], xlab="Minutes", main="histogram of the Minutes")
```



(b)

```r
set.seed(1008494620)
population = df[["Minutes"]]
sampleCnt = 10000
n = 30
```

```
yBar = numeric(sampleCnt)
stats = numeric(sampleCnt)
for (i in 1:sampleCnt) {
  y = sample(population, size=n, replace=TRUE)
  yBar[i] = mean(y)
  s2 = var(y)
  stats[i] = (n-1) * s2 / sigma2
}
```

(b)(i)

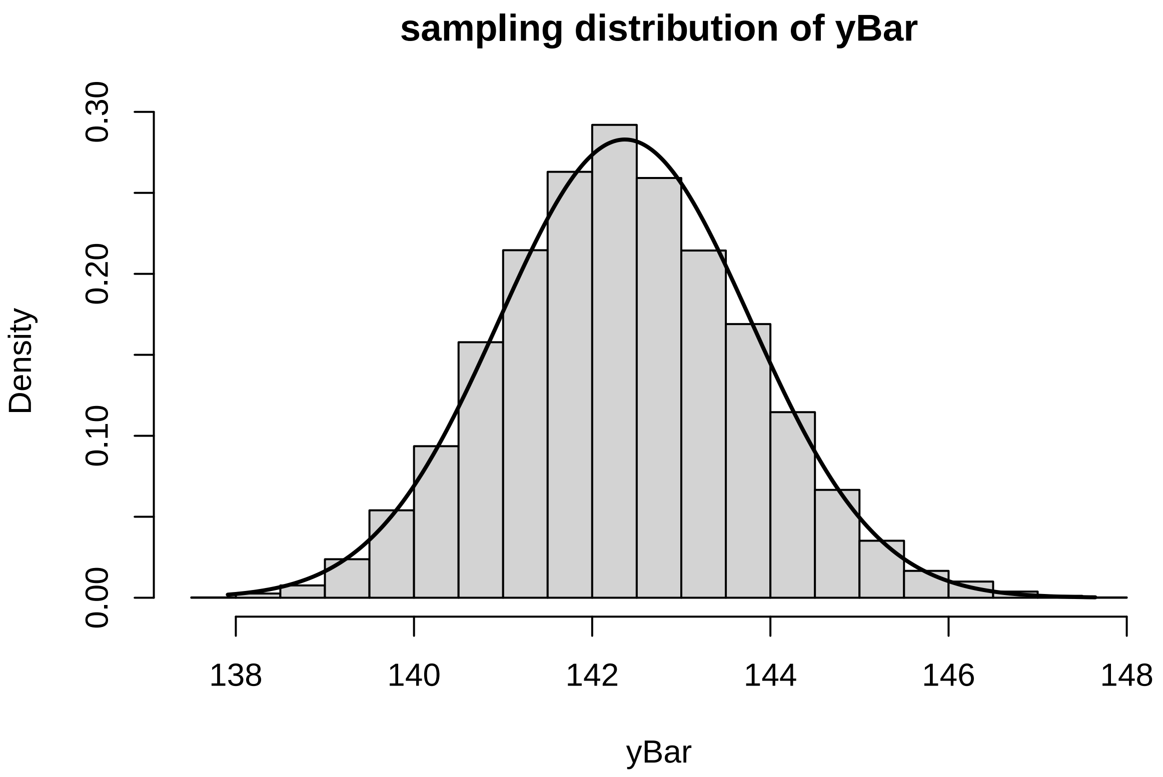
```
mean_yBar = mean(yBar)
var_yBar = var(yBar)
mean_stats = mean(stats)
var_stats = var(stats)

percentiles = c(0.025, 0.25, 0.5, 0.75, 0.975)
percentiles_pop = mapply(
  function(p) {
    quantile(df[["Minutes"]], p)
  },
  percentiles
)
percentiles_yBar = mapply(
  function(p) {
    quantile(yBar, p)
  },
  percentiles
)
percentiles_stats = mapply(
  function(p) {
    quantile(stats, p)
  },
  percentiles
)

cat(
  sprintf("theoretical values\n"),
  sprintf("==================\n"),
  sprintf("mean = %f, var = %f\n", mu, sigma2),
  sprintf("percentiles:\n"),
  percentiles_pop
)

cat(
  sprintf("\n\nresult of sample means\n"),
  sprintf("======================\n"),
  sprintf("mean = %f, var = %f\n", mean_yBar, var_yBar),
  sprintf("percentiles:\n"),
  percentiles_yBar
)

cat(
```

```r
  sprintf("\n\nresult of scaled sample var\n"),
  sprintf("===========================\n"),
  sprintf("mean = %f, var = %f\n", mean_stats, var_stats),
  sprintf("percentiles:\n"),
  percentiles_stats
)
```

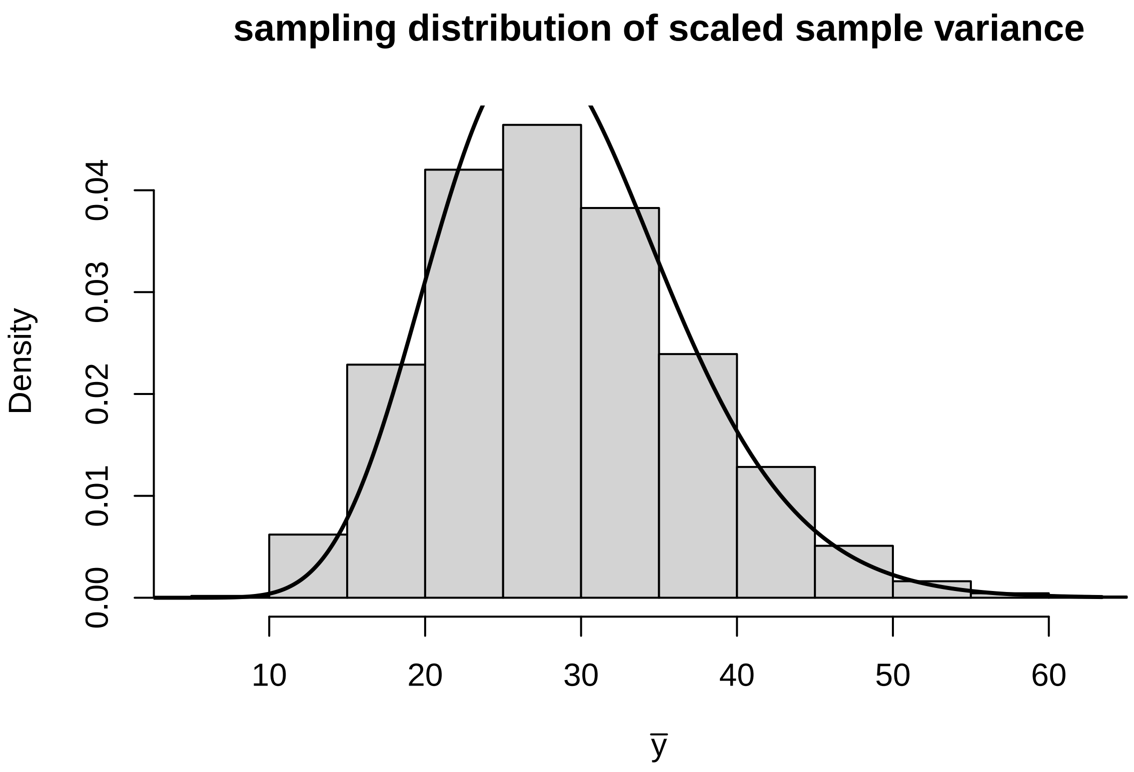
```
## theoretical values
##  ==================
##  mean = 142.367143, var = 59.638923
##  percentiles:
##  131.1555 137.1225 140.765 147.035 159.8852
##
## result of sample means
##  ======================
##  mean = 142.354664, var = 1.984513
##  percentiles:
##  139.7113 141.3853 142.3157 143.2964 145.1947
##
## result of scaled sample var
##  ===========================
##  mean = 28.747533, var = 69.091112
##  percentiles:
##  14.39847 22.80234 28.02941 34.012 46.61924
```

(b) (ii)

```r
hist(yBar,
  probability = TRUE,
  main="sampling distribution of yBar",
  xlab="yBar",
)
x1 = seq(min(yBar), max(yBar), length.out = 500)
lines(x1, dnorm(x1, mean=mu, sd=sqrt(sigma2/n)), lwd=2)
```



```
hist(stats,
  probability=TRUE,
  main="sampling distribution of scaled sample variance",
  xlab=expression(bar(y)),
)
x2 = seq(0, max(stats), length.out = 500)
lines(x2, dchisq(x2, df=n - 1), lwd=2)
```



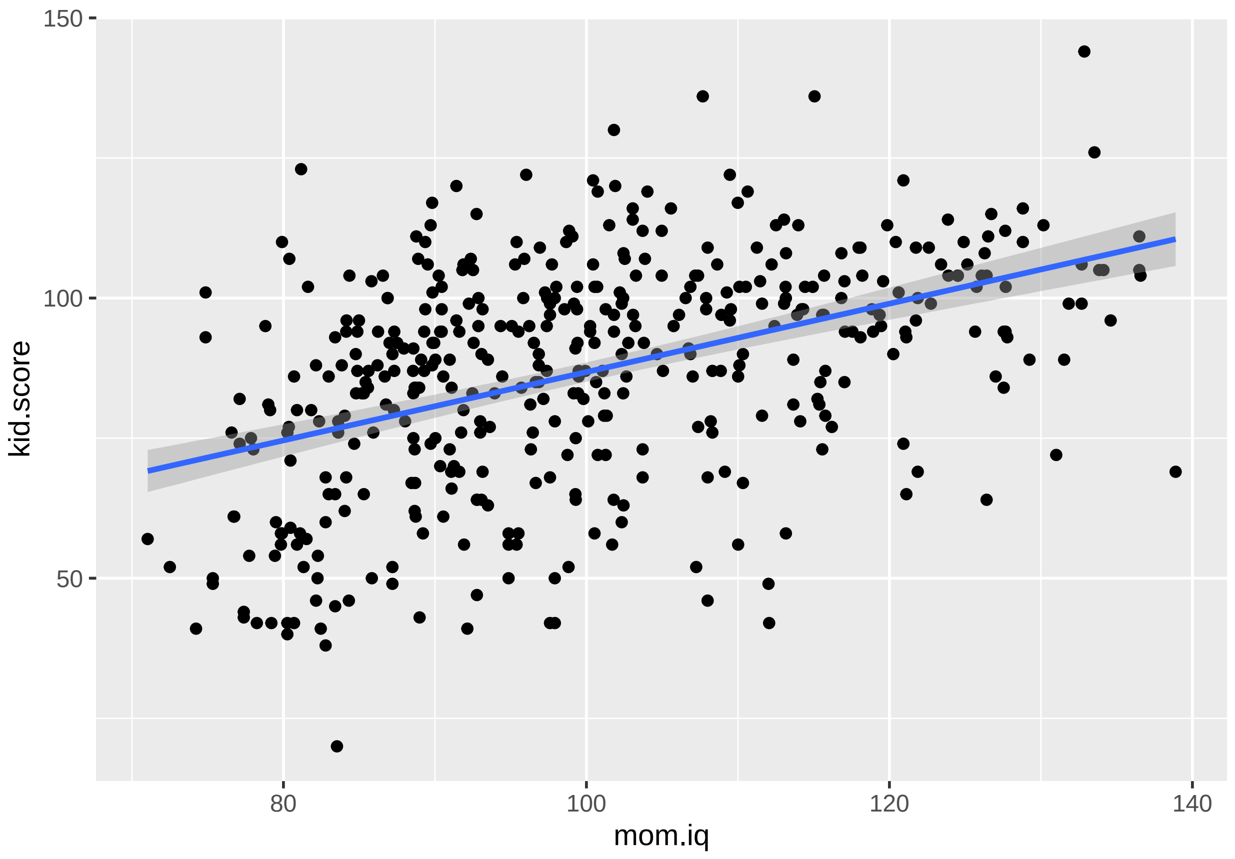
Q7

(a)

```r
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.5.2
```

```r
csvFile = "kidiq.csv"
df = read.csv(csvFile)
ggplot(
  df,
  aes(
    x = mom.iq,
    y = kid.score,
  )
) + geom_point() + geom_smooth(formula = y~x, method = "lm")
```



(b)

```
y = df[["kid.score"]]
x = df[["mom.iq"]]
fit = lm(y~x, data = df)
beta0 = fit$coefficients[["(Intercept)"]]
beta1 = fit$coefficients[["x"]]
cat(sprintf("beta0 = %f, beta1 = %f", beta0, beta1))
```

```
## beta0 = 25.799778, beta1 = 0.609975
```

(c)

```
x_bar = mean(x)
y_bar = mean(y)

SS_yy = sum((y - y_bar)^2)
SS_xy = sum((y - y_bar) * (x - x_bar))
SS_xx = sum((x - x_bar)^2)

beta1_hat = SS_xy / SS_xx
beta0_hat = y_bar - beta1_hat * x_bar

cat(
  sprintf("R built-in function\n"),
  sprintf("===================\n"),
  sprintf("intercept = %f, slope = %f\n", beta0, beta1),

  sprintf("\nComputed result\n"),
  sprintf("===============\n"),
  sprintf("beta0_hat = %f, beta1_hat = %f", beta0_hat, beta1_hat)
)
```

```
## R built-in function
##  ==================
##  intercept = 25.799778, slope = 0.609975
##
## Computed result
##  ===============
##  beta0_hat = 25.799778, beta1_hat = 0.609975
```

(c)

```
alpha = 0.05
n = length(x)

sse = SS_yy - beta1_hat * SS_xy
ss = sse / (n - 2)
se_beta1 = sqrt(ss / SS_xx)
se_beta0 = sqrt((1/n + x_bar^2/SS_xx) * ss)

# two-sided CI
tcrit = qt(1 - alpha/2, df = n - 2)

beta0_ci_upper = beta0_hat + tcrit * se_beta0
beta0_ci_lower = beta0_hat - tcrit * se_beta0

beta1_ci_upper = beta1_hat + tcrit * se_beta1
beta1_ci_lower = beta1_hat - tcrit * se_beta1

cat(
  sprintf("handcrafted CIs\n"),
  sprintf("===============\n"),
  sprintf("            2.5%%      97.5%%\n"),
  sprintf("beta0: %f, %f\n", beta0_ci_lower, beta0_ci_upper),
  sprintf("beta1: %f, %f\n", beta1_ci_lower, beta1_ci_upper),

  sprintf("\n\nbuilt-in function\n"),
  sprintf("=================\n")
)
```

```
## handcrafted CIs
##  ===============
##             2.5%      97.5%
##  beta0: 14.169279, 37.430277
##  beta1: 0.494953, 0.724996
##
##
## built-in function
##  =================
confint(fit)
```

```
##                  2.5 %     97.5 %
## (Intercept) 14.1692789 37.4302768
## x            0.4949534  0.7249957
```

(d)

```
t = beta1_hat / se_beta1
tcrit = qt(1 - alpha, df = n-2)
sprintf("test stats: %f, t-criteria: %f\n", t, tcrit)
if (t > tcrit) {
  cat(
    sprintf("Since t > tcrit, the test stats does not lie within the confidence\n"),
    sprintf("region, so we reject H0; there is no positive correlation between\n"),
    sprintf("two variables.\n")
  )
} else {
  cat(
    sprintf("Since t <= tcrit, the test stats does lie within the confidence\n"),
    sprintf("region, so we accept H0; there is a positive correlation between\n"),
    sprintf("two variables.\n")
  )
}

p_val = 2 * pt(t, df = n-2, lower.tail = FALSE)
sprintf("the p value is given by: %f", p_val)
```

```
## [1] "test stats: 10.423188, t-criteria: 1.648388\n"
## Since t > tcrit, the test stats does not lie within the confidence
##  region, so we reject H0; there is no positive correlation between
##  two variables.
## [1] "the p value is given by: 0.000000"
```

(e)

```
tt = 0.1 / se_beta1
p = 2 * (1 - pt(tt, df = n-2))
sprintf("the probability is: %f", p)
```

```
## [1] "the probability is: 0.088208"
```