STAT 341: Assignment 3

DUE: Monday March 16 by 11:59pm EST

QUESTION 1: Data Prep and EDA [7 points]

(a) [1 point] Load the data and save it in a variable called `race`.

Solution:

```
race <- read.csv("/Users/nstevens/Dropbox/Teaching/STAT_341/Assignments/Assignment3/SF_Marathon_2018.csv
```

(b) [1 point] Replace the entries in the `Time` column of `race` with marathon times recorded in minutes. In other words, convert the character string `H:MM:SS` into a numeric value, e.g., `"4:11:32"` would be converted to `251.5333`. To confirm that you have done this correctly print out the first 6 rows of the dataframe `race`. **Note:** You might find the `hour`, `minute`, `second`, and `hms` functions from the `lubridate` package helpful.

Solution:

```
library(lubridate)
tempTime <- hms(race$Time)
race$Time <- 60 * hour(hms(race$Time)) + minute(hms(race$Time)) + second(hms(race$Time))/60
head(race)
```

```
##   Place              Name Bib     Time Pace Sex Age.Division
## 1     1   Jorge Maravilla   1 147.9333 5:39   M        40-44
## 2     2 Jonathan Briskman  61 148.1333 5:39   M        25-29
## 3     3     Benjamin Heck   5 150.5667 5:45   M        25-29
## 4     4       Alex Varner  68 153.6167 5:52   M        30-34
## 5     5       Doug Howard  71 153.6167 5:52   M        35-39
## 6     6    Angel Alcantar  59 156.1167 5:58   M        20-24
```

(c) [1 point] Remove the two runners for whom `Age.Division = "NoAge"`. Be sure to state which rows were eliminated. **Note:** For the rest of the assignment, the remaining $N = 5262$ runners will be treated as the population $\mathcal{P}$ of interest.

Solution:

```
which(race$Age.Division == "NoAge")
```

```
## [1] 2353 4290
```

```
race <- race[-c(2353, 4290), ]
```

(d) [1 point] Calculate and state the average marathon time. **Note:** For the Questions 2 and 3, we will treat this value as $a(\mathcal{P})$, the attribute of interest calculated on the entire population.

Solution:

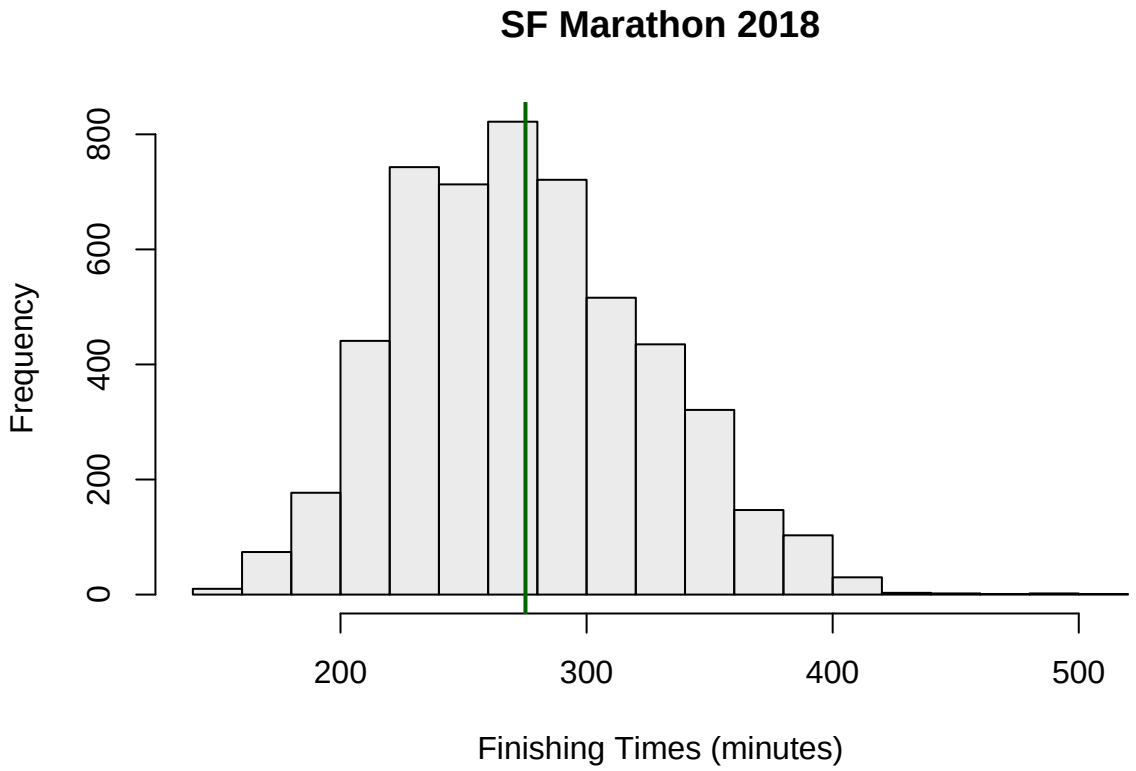
```
aveTime <- mean(race$Time)
print(aveTime)
```

```
## [1] 275.1708
```

(e) [2 points] Construct a histogram of the marathon finishing times and, with a vertical line, indicate the average finishing time. Be sure to include a relevant title and axis labels.

Solution:

```
hist(race$Time, main = "SF Marathon 2018", xlab = "Finishing Times (minutes)",
    col = adjustcolor("grey", 0.3))
abline(v = aveTime, col = "darkgreen", lwd = 2)
```



(f) [1 point] How many minutes did it take `Nathaniel Stevens` to finish the marathon? Please round your answer to 4 decimal places.

Solution:

```
round(race[which(race$Name == "Nathaniel Stevens"), "Time"], 4)
```

```
## [1] 251.5333
```

QUESTION 2: Horvitz-Thompson Estimation – SRSWOR [12 points]

(a) [5 points] Using the following code, take a *simple random sample without replacement* of size $n = 500$ from the population. (This is not worth points)

```
N <- 5262
n <- 500
srsSampIndex <- read.table("srsSampIndex.txt")$V1
srsSamp <- race[srsSampIndex, ]
```

i. [2 points] Calculate the Horvitz-Thompson estimate of the average finishing time.

Solution:

```
y_u <- srsSamp$Time/N
incl.prob <- rep(n/N, N)
pi_u <- incl.prob[srsSampIndex]
aveTimeHT.srs <- sum(y_u/pi_u)
```

```
print(aveTimeHT.srs)
```

```
## [1] 273.265
```

ii. [2 points] Calculate the standard error for this estimate. You may find the following function useful:

```
estVarHT <- function(y_u, pi_u, pi_uv) {
    ## y_u = an n element array containing the variate values for the sample

    ## pi_u = an n element array containing the (marginal) inclusion
    ## probabilities for the sample

    ## pi_uv = an nxn matrix containing the joint inclusion probabilities
    ## for the sample

    delta <- pi_uv - outer(pi_u, pi_u)
    estimateVar <- sum((delta/pi_uv) * outer(y_u/pi_u, y_u/pi_u))
    return(abs(estimateVar))
}
```

Solution:

```
joint.incl.prob <- matrix((n * (n - 1))/(N * (N - 1)), nrow = N, ncol = N)
diag(joint.incl.prob) <- incl.prob
pi_uv = joint.incl.prob[srsSampIndex, srsSampIndex]

SEaveTimeHT.srs <- sqrt(estVarHT(y_u, pi_u, pi_uv))
print(SEaveTimeHT.srs)
```

```
## [1] 2.176127
```

iii. [1 point] Calculate an approximate 95% confidence interval for the average finishing time.

Solution:

```
aveTimeHT.srs + 2 * c(-1, 1) * SEaveTimeHT.srs
```

```
## [1] 268.9127 277.6173
```

(b) [7 points] In this question you will explore the dependency of the Horvitz-Thompson estimator's sampling distribution on sample size. Consider the sample sizes $n \in \{100, 200, \ldots, 1000\}$.

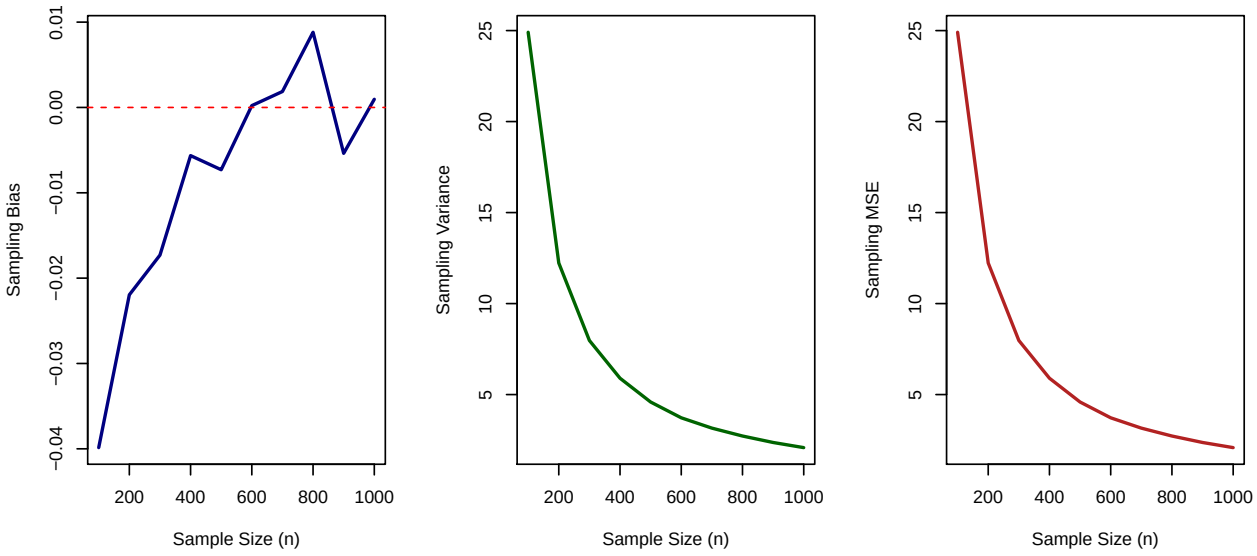
i. [4 points]

- For each sample size n take 50,000 SRSWOR samples from the population.
- For each of the 50,000 samples of a given size, calculate the Horvitz-Thompson estimate of the average finishing time.
- For each sample size n use the 50,000 HT estimates to estimate the sampling bias, sampling variance, and sampling MSE.
- Construct three line-plots (layed out in a 1×3 grid) of bias vs. n, variance vs. n and MSE vs. n. As always, make sure your plots are informatively labelled.

Solution:

```
N <- dim(race)[1]
n <- seq(100, 1000, 100)
bias.srs <- rep(0, length(n))
variance.srs <- rep(0, length(n))
mse.srs <- rep(0, length(n))
for (i in 1:length(n)) {
```

```
    pi.vec <- rep(n[i]/N, N)
    pi.mat <- matrix((n[i] * (n[i] - 1))/(N * (N - 1)), nrow = N, ncol = N)
    avgTimeHT.srs <- rep(0, 50000)
    for (j in 1:50000) {
        srsSampIndex <- sample(N, n[i])
        y_u <- race$Time[srsSampIndex]/N
        pi_u <- pi.vec[srsSampIndex]
        avgTimeHT.srs[j] <- sum(y_u/pi_u)
    }
    bias.srs[i] <- mean(avgTimeHT.srs - aveTime)
    variance.srs[i] <- var(avgTimeHT.srs)
    mse.srs[i] <- mean((avgTimeHT.srs - aveTime)^2)
}
par(mfrow = c(1, 3))
plot(n, bias.srs, type = "l", xlab = "Sample Size (n)", ylab = "Sampling Bias",
    col = "navyblue", lwd = 2)
abline(h = 0, col = "red", lty = 2)
plot(n, variance.srs, type = "l", xlab = "Sample Size (n)", ylab = "Sampling Variance",
    col = "darkgreen", lwd = 2)
plot(n, mse.srs, type = "l", xlab = "Sample Size (n)", ylab = "Sampling MSE",
    col = "firebrick", lwd = 2)
```



ii. [3 points] Comment on the relationship between bias, variance, MSE and n for the Horvitz-Thompson Estimator.

Solution:

- We see that regardless of sample size, the estimated bias of the estimator appears to randomly fluctuate around zero. This is unsurprising since the HT estimator is unbiased.
- We see clearly that as sample size increase, the estimated variance of the estimator decreases and approaches zero.
- We see clearly that as sample size increase, the estimated MSE of the estimator decreases and approaches zero. We also notice that the MSE and variance curves are essentially identical which is also unsurprising since $MSE[\widetilde{a_{HT}}(\mathcal{P})] = Var[\widetilde{a_{HT}}(\mathcal{P})]$

Together these results suggest that as n increases, HT estimates are both accurate and precise.

QUESTION 3: Horvitz-Thompson Estimation – Stratified Random Sampling [24 points]

Stratified randomly sampling is a probabilistic sampling mechanism that is applicable when a population $\mathcal{P}$ can be partitioned into H strata (i.e., sub-populations) $\{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_H\}$ such that

$$\mathcal{P} = \mathcal{P}_1 \cup \mathcal{P}_2 \cup \ldots \cup \mathcal{P}_H$$

and

$$N = N_1 + N_2 + \cdots + N_H$$

where N_h is the size of strata $h = 1, 2, \ldots, H$.

In this setting, a sample $\mathcal{S}$ of size n from $\mathcal{P}$ is obtained by taking a *simple random sample without replacement* from *each* of the H strata. Thus, sample $\mathcal{S}_1$ of size n_1 is taken from strata 1, sample $\mathcal{S}_2$ of size n_2 is taken from strata 2, and so on. Then

$$\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2 \cup \ldots \cup \mathcal{S}_H$$

where $n = n_1 + n_2 + \cdots + n_H$ and each $\mathcal{S}_h$ is a SRSWOR.

The function `stratRS` defined below may be used to take a stratified randomly sample $\mathcal{S}$ of size n from a population $\mathcal{P}$ of size N. The inputs are:

- `stratLabel`: an N element array containing the integer labels $\{1, 2, \ldots, H\}$ indicating which strata each unit belongs to.
- `stratSampSize`: an H element array containing $\{n_1, n_2, \ldots, n_H\}$, the sample sizes to be drawn from each strata.

```
stratRS <- function(stratLabel, stratSampSize) {
    H <- length(stratSampSize)
    sampIndex <- list()
    for (h in 1:H) {
        sampIndex[[h]] <- sample(which(stratLabel == h), size = stratSampSize[h],
            replace = FALSE)
    }
    return(unlist(sampIndex))
}
```

(a) Marginal Inclusion Probabilities

i. [2 points] Show that the (marginal) inclusion probability π_u for stratified random sampling is

$$\pi_u = \frac{n_h}{N_h} \qquad \text{if } u \in \mathcal{P}_h$$

Solution:

For unit $u \in \mathcal{P}_h$ to be selected into the sample it means that they must be selected when strata h is being sampled. Since strata h has N_h units and n_h units are being selected at random and without replacement, the probability of this happening is

$$\pi_u = Pr(u \in \mathcal{S}) = Pr(u \in \mathcal{S}_h) = \frac{\binom{N_h - 1}{n_h - 1}}{\binom{N_h}{n_h}} = \frac{n_h}{N_h}$$

ii. [2 points] Write a function called `getInclusionProbStrat` which outputs an N element array containing the inclusion probabilities for each unit in the population $\mathcal{P}$, and which takes as inputs `stratLabel` and `stratSampSize` as defined above.

Solution:

```
getInclusionProbStrat <- function(stratLabel, stratSampSize) {
    H <- length(stratSampSize)
    stratSize <- as.numeric(table(stratLabel))
    N <- sum(stratSize)
    pi_u <- rep(0, N)
    for (h in 1:H) {
        pi_u[which(stratLabel == h)] <- stratSampSize[h]/stratSize[h]
    }
    return(pi_u)
}
```

iii. [1 point] Call `getInclusionProbStrat` using `stratLabel = c(1,1,1,2,2,2,2)` and `stratSampSize = c(2,3)` and output the result.

Solution:

```
getInclusionProbStrat(stratLabel = c(1, 1, 1, 2, 2, 2, 2), stratSampSize = c(2,
    3))
```

```
## [1] 0.6666667 0.6666667 0.6666667 0.7500000 0.7500000 0.7500000 0.7500000
```

(b) Joint Inclusion Probabilities

i. [4 points] Show that the joint inclusion probability π_{uv} for stratified random sampling is

$$\pi_{uv} = \begin{cases} \frac{n_h(n_h-1)}{N_h(N_h-1)} & \text{if } u, v \in \mathcal{P}_h \\ \frac{n_h n_k}{N_h N_k} & \text{if } u \in \mathcal{P}_h, v \in \mathcal{P}_k \end{cases}$$

Solution:

In the case that units u and v are both in strata h (i.e., $u, v \in \mathcal{P}_h$), units u and v will both be included in the sample $\mathcal{S}$ only if they are both selected when $\mathcal{P}_h$ is sampled. Since strata h has N_h units and n_h units are being selected at random and without replacement, the probability of this happening is

$$\pi_{uv} = Pr(u \in \mathcal{S}, v \in \mathcal{S}) = Pr(u \in \mathcal{S}_h, v \in \mathcal{S}_h) = \frac{\binom{N_h-2}{n_h-2}}{\binom{N_h}{n_h}} = \frac{n_h(n_h-1)}{N_h(N_h-1)}$$

In the case that unit u is in strata h and unit v is in strata k then units u and v will both be included in the sample $\mathcal{S}$ only if u is selected when $\mathcal{P}_h$ is sampled and v is selected when $\mathcal{P}_k$ is sampled. Since each of the strata is sampled independently of the other, and sampling is done randomly and without replacement, the joint inclusion probability in this case is

$$\pi_{uv} = Pr(u \in \mathcal{S}, v \in \mathcal{S}) = Pr(u \in \mathcal{S}_h, v \in \mathcal{S}_k) = Pr(u \in \mathcal{S}_h)Pr(v \in \mathcal{S}_k) = \frac{n_h n_k}{N_h N_K}$$

ii. [2 points] Write a function called `getJointInclusionProbStrat` which outputs an $N \times N$ matrix containing the joint inclusion probabilities for each unit in the population $\mathcal{P}$, and which takes as inputs `stratLabel` and `stratSampSize` as defined above.

Solution:

```
getJointInclusionProbStrat <- function(stratLabel, stratSampSize) {
    H <- length(stratSampSize)
    stratSize <- as.numeric(table(stratLabel))
    N <- sum(stratSize)
    pi_uv <- matrix(0, N, N)
```

```
    for (u in 1:N) {
        for (v in 1:N) {
            if (u == v) {
                pi_uv[u, v] <- stratSampSize[stratLabel[u]]/stratSize[stratLabel[u]]
            } else {
                if (stratLabel[u] == stratLabel[v]) {
                  pi_uv[u, v] <- (stratSampSize[stratLabel[u]] * (stratSampSize[stratLabel[u]] -
                    1))/(stratSize[stratLabel[u]] * (stratSize[stratLabel[u]] -
                    1))
                } else {
                  pi_uv[u, v] <- (stratSampSize[stratLabel[u]]/stratSize[stratLabel[u]]) *
                    (stratSampSize[stratLabel[v]]/stratSize[stratLabel[v]])
                }
            }
        }
    }
    return(pi_uv)
}
```

iii. [1 point] Call `getJointInclusionProbStrat` using `stratLabel = c(1,1,1,2,2,2,2)` and `stratSampSize = c(2,3)` and output the result.

Solution:

```
getJointInclusionProbStrat(stratLabel = c(1, 1, 1, 2, 2, 2, 2), stratSampSize = c(2,
    3))
```

```
##           [,1]      [,2]      [,3] [,4] [,5] [,6] [,7]
## [1,] 0.6666667 0.3333333 0.3333333 0.50 0.50 0.50 0.50
## [2,] 0.3333333 0.6666667 0.3333333 0.50 0.50 0.50 0.50
## [3,] 0.3333333 0.3333333 0.6666667 0.50 0.50 0.50 0.50
## [4,] 0.5000000 0.5000000 0.5000000 0.75 0.50 0.50 0.50
## [5,] 0.5000000 0.5000000 0.5000000 0.50 0.75 0.50 0.50
## [6,] 0.5000000 0.5000000 0.5000000 0.50 0.50 0.75 0.50
## [7,] 0.5000000 0.5000000 0.5000000 0.50 0.50 0.50 0.75
```

(c) [2 points] Add a new column to the `race` dataframe called `stratLabel` which assigns a numeric label to each age division. In particular, when `Age.Division = "Under20"` then `stratLabel = 1`; when `Age.Division = "20-24"` then `stratLabel = 2`; when `Age.Division = "25-29"` then `stratLabel = 3`; … when `Age.Division = "65+"` then `stratLabel = 11`. Once you have done this, present the output from the command `table(race$stratLabel)`.

Solution:

```
temp <- rep(0, N)
temp[race$Age.Division == "Under20"] <- 1
temp[race$Age.Division == "20-24"] <- 2
temp[race$Age.Division == "25-29"] <- 3
temp[race$Age.Division == "30-34"] <- 4
temp[race$Age.Division == "35-39"] <- 5
temp[race$Age.Division == "40-44"] <- 6
temp[race$Age.Division == "45-49"] <- 7
temp[race$Age.Division == "50-54"] <- 8
temp[race$Age.Division == "55-59"] <- 9
temp[race$Age.Division == "60-64"] <- 10
temp[race$Age.Division == "65+"] <- 11
```

```
race$stratLabel <- temp
table(race$stratLabel)
```

```
##
##    1    2    3    4    5    6    7    8    9   10   11
##   73  486 1016  881  742  603  561  419  253  141   87
```

(d) [5 points] Using the following code, take a *stratifed random sample* of size $n = 500$ from the population. (This is not worth points)

```
stratSampIndex <- read.table("stratSampIndex.txt")$V1
stratSamp <- race[stratSampIndex, ]
```

i. [2 points] Calculate the Horvitz-Thompson estimate of the average finishing time. It will be useful to use your `getInclusionProbStrat` function.

Solution:

```
y_u <- stratSamp$Time/N
incl.prob <- getInclusionProbStrat(stratLabel = race$stratLabel, stratSampSize = c(7,
    46, 97, 84, 71, 57, 53, 40, 24, 13, 8))
pi_u <- incl.prob[stratSampIndex]
aveTimeHT.strat <- sum(y_u/pi_u)
print(aveTimeHT.strat)
```

```
## [1] 274.4291
```

ii. [2 points] Calculate the standard error for this estimate. Feel free to use the `estVarHT` function. It will also be useful to use your `getJointInclusionProbStrat` function.

Solution:

```
joint.incl.prob <- getJointInclusionProbStrat(stratLabel = race$stratLabel,
    stratSampSize = c(7, 46, 97, 84, 71, 57, 53, 40, 24, 13, 8))
pi_uv = joint.incl.prob[stratSampIndex, stratSampIndex]

SEaveTimeHT.strat <- sqrt(estVarHT(y_u, pi_u, pi_uv))
print(SEaveTimeHT.strat)
```

```
## [1] 2.194157
```

iii. [1 point] Calculate an approximate 95% confidence interval for the average finishing time.

Solution:

```
aveTimeHT.strat + 2 * c(-1, 1) * SEaveTimeHT.strat
```

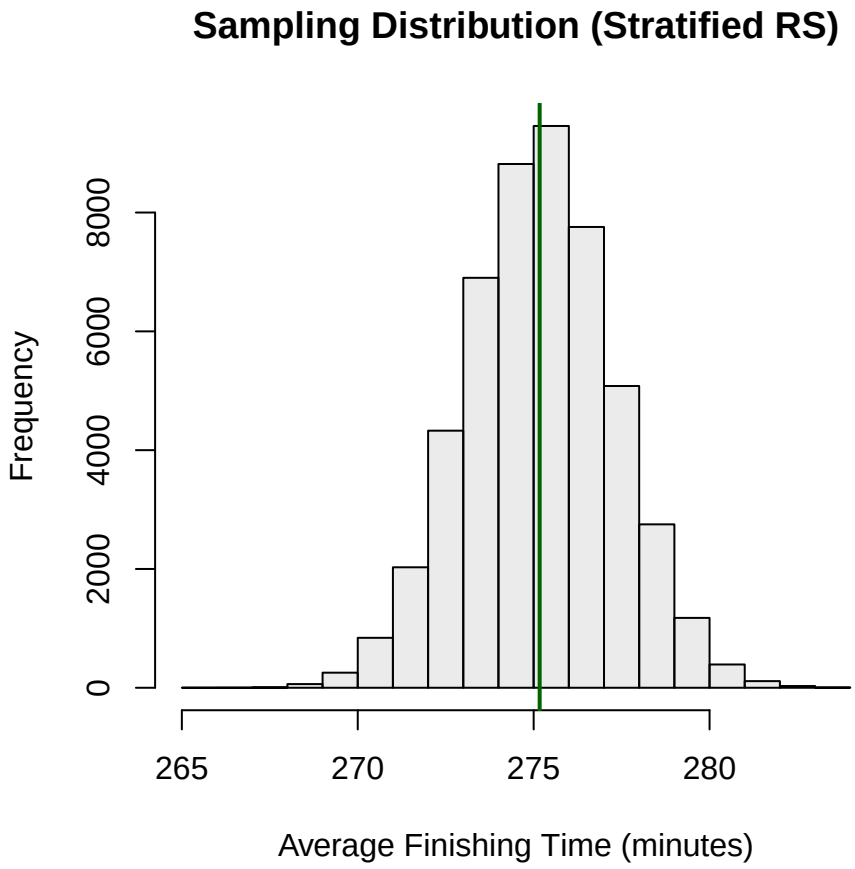
```
## [1] 270.0408 278.8174
```

(e) [3 points] Take 50,000 stratified random samples of size $n = 500$ from the population with $\{n_1 = 7, n_2 = 46, n_3 = 97, n_4 = 84, n_5 = 71, n_6 = 57, n_7 = 53, n_8 = 40, n_9 = 24, n_{10} = 13, n_{11} = 8\}$. For each sample calculate the Horvitz-Thompson estimate of the average finishing time. Construct a histogram of these estimates and overlay a vertical line indicating $a(\mathcal{P})$ the true average finishing time.

Solution:

```
avgTimeHT.strat <- rep(0, 50000)
for (j in 1:50000) {
    stratSampIndex <- stratRS(stratLabel = race$stratLabel, stratSampSize = c(7,
        46, 97, 84, 71, 57, 53, 40, 24, 13, 8))
    y_u <- race$Time[stratSampIndex]/N
```

```
    pi_u <- incl.prob[stratSampIndex]
    avgTimeHT.strat[j] <- sum(y_u/pi_u)
}
hist(avgTimeHT.strat, main = "Sampling Distribution (Stratified RS)", xlab = "Average Finishing Time (m
    col = adjustcolor("grey", 0.3))
abline(v = aveTime, col = "darkgreen", lwd = 2)
```



(f) [2 points] Using the approximate sampling distribution calculated in (e), estimate the sampling bias, sampling variance and sampling mean squared error (MSE) of the stratified random sample Horvitz-Thompson estimator. Compare these results to the $n = 500$ case from Question 2(b) i. and state which sampling mechanism is to be preferred and explain why.

Solution

```
bias.strat <- mean(avgTimeHT.strat - aveTime)
variance.strat <- var(avgTimeHT.strat)
mse.strat <- mean((avgTimeHT.strat - aveTime)^2)
kable(data.frame(bias = bias.strat, variance = variance.strat, mse = mse.strat))
```

bias	variance	mse
0.0033321	4.393662	4.393586

We can compare the sampling mechanisms on the basis of their mean squared errors. As shown above, the MSE for stratified random sampling is 4.3936 and the MSE for SRSWOR when $n = 500$ was 4.5909. Because the MSE associated with stratified random sampling is smaller we would prefer to use it over SRSWOR in this case.

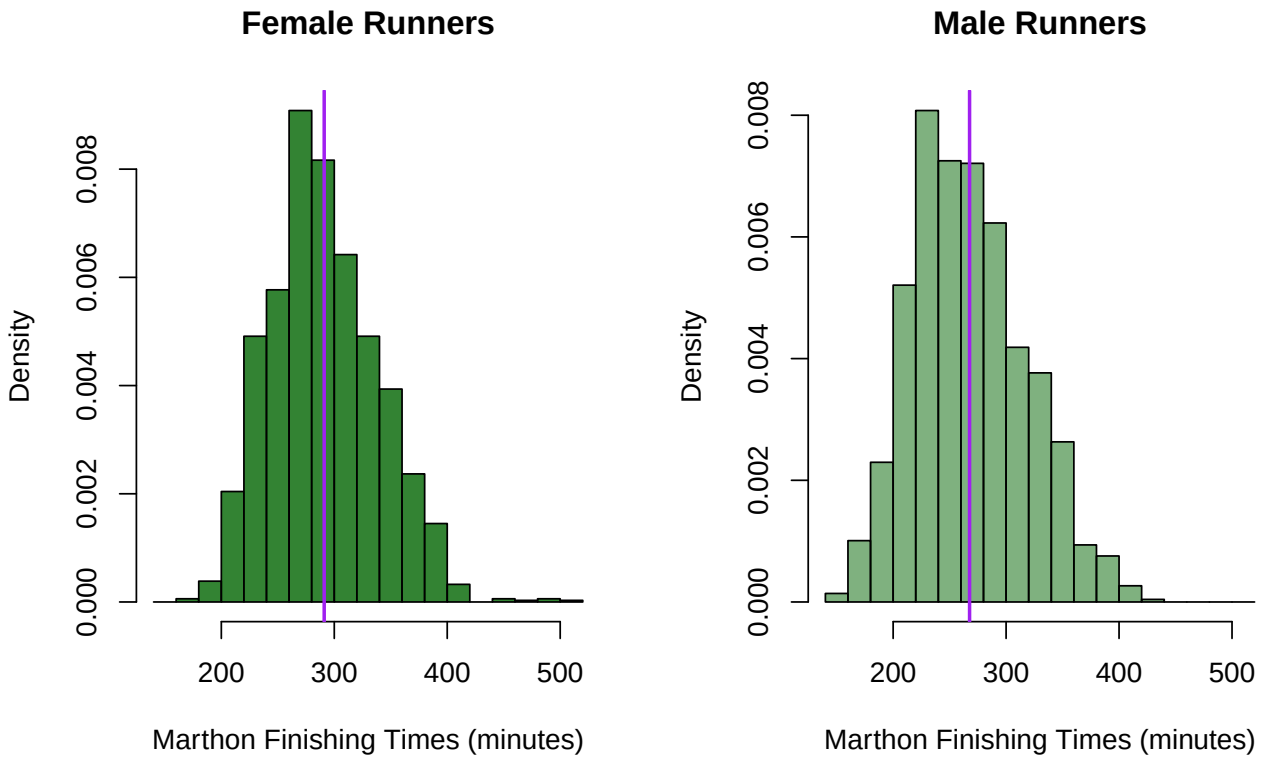
QUESTION 4: Permutation Test [12 points]

Consider the female (F) and male (M) subpopulations of runners. We may refer to them as $\mathcal{P}_F$ and $\mathcal{P}_M$ and we note that $\mathcal{P} = \mathcal{P}_F \cup \mathcal{P}_M$.

(a) [3 points] Construct two *density* histograms: one of the female finishing times and the other of the male finishing times, and plot them next to each in a 1×2 grid. Be sure to include informative titles and axis labels. To enhance comparability ensure the x-axis of each histogram is the same, ensure both histograms have the same number of bins, and indicate (with a vertical line) the average marathon time in each sub-population.

Solution:

```
pop <- list(pop1 = race[race$Sex == "F", ], pop2 = race[race$Sex == "M",
    ])
par(mfrow = c(1, 2))
hist(pop[[1]]$Time, col = adjustcolor("darkgreen", 0.8), freq = FALSE,
    breaks = seq(140, 520, 20), xlab = "Marthon Finishing Times (minutes)",
    main = "Female Runners")
abline(v = mean(pop[[1]]$Time), col = "purple", lwd = 2)
hist(pop[[2]]$Time, col = adjustcolor("darkgreen", 0.5), freq = FALSE,
    breaks = seq(140, 520, 20), xlab = "Marthon Finishing Times (minutes)",
    main = "Male Runners")
abline(v = mean(pop[[2]]$Time), col = "purple", lwd = 2)
```



(b) [1 point] State the null hypothesis H_0 that is being tested when comparing these two sub-populations with a permutation test.

Solution: $H_0 : \mathcal{P}_F$ and $\mathcal{P}_M$ are drawn from the same population of marathon times.

In other words, the distribution of female and male marathon times are indistinguishable.

(c) [4 points] In this question you will test the hypothesis in (b) using the discrepancy measure

$$D(\mathcal{P}_F, \mathcal{P}_M) = |\overline{y}_F - \overline{y}_M|$$

i. [1 point] Calculate the observed discrepancy.

Solution:

```
DiscrepMeanFn <- function(variate) {
    function(pop) {
        abs(mean(pop$pop1[, variate]) - mean(pop$pop2[, variate]))
    }
}
diffAveTimes <- DiscrepMeanFn("Time")

dobs <- diffAveTimes(pop)
print(dobs)
```

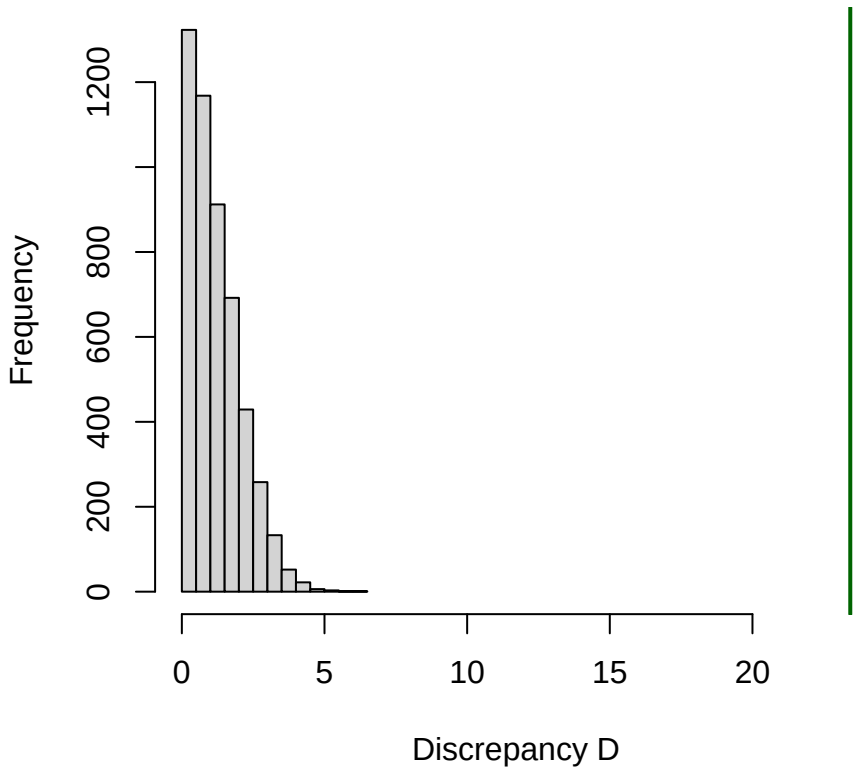
```
## [1] 23.42872
```

ii. [1 point] Randomly mix the populations $M = 5,000$ times and construct a histogram of the 5,000 $D(\mathcal{P}_F^\star, \mathcal{P}_M^\star)$ values. Indicate, with a vertical line, the observed discrepancy calculated in i. Note that you may use the `mixRandomly` function from class.

Solution:

```
set.seed(341)
diffTimes <- sapply(1:5000, FUN = function(...) {
    diffAveTimes(mixRandomly(pop))
})

hist(diffTimes, breaks = 20, xlim = c(0, dobs), main = "Randomly Mixed Populations",
    xlab = "Discrepancy D", col = "lightgrey")
abline(v = dobs, col = "darkgreen", lwd = 2)
```



iii. [1 point] Calculate the p-value associated with this test.

Solution:

```
mean(diffTimes >= diffAveTimes(pop))
```

```
## [1] 0
```

iv. [1 point] Based on the p-value calculated in iii. what do you conclude about the comparability of these two populations? In other words, summarize your findings and draw a conclusion about the null hypothesis from part (b).

Solution: Our estimated p-value is zero. This means that we have very strong evidence against the null hypothesis that female and male marathon times are indistinguishable, as evaluated by a comparison of averages.

(d) [4 points] In this question you will test the hypothesis in (b) using the discrepancy measure

$$D(\mathcal{P}_F, \mathcal{P}_M) = \left| \frac{SD(\mathcal{P}_F)}{SD(\mathcal{P}_M)} - 1 \right|$$

i. [1 point] Calculate the observed discrepancy.

Solution:

```
DiscrepSDFn <- function(variate) {
    function(pop) {
        abs(sd(pop$pop1[, variate])/sd(pop$pop2[, variate]) - 1)
    }
}
ratioSDTimes <- DiscrepSDFn("Time")
```

```
dobs <- ratioSDTimes(pop)
print(dobs)
```

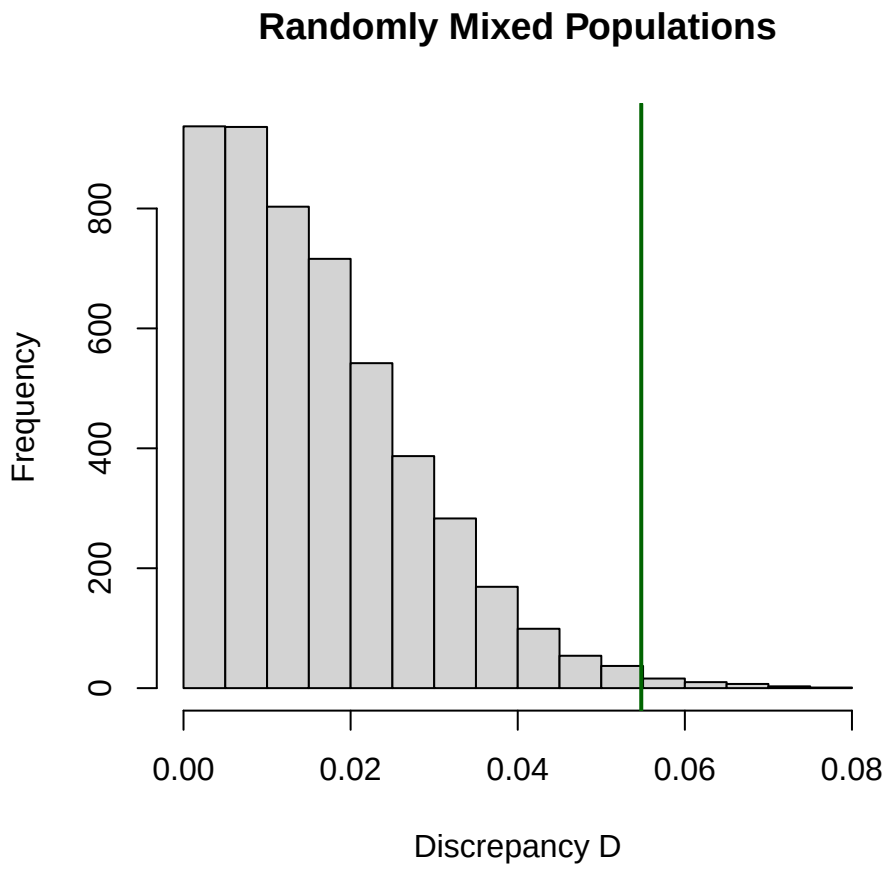
```
## [1] 0.05478395
```

ii. [1 point] Randomly mix the populations $M = 5,000$ times and construct a histogram of the 5,000 $D(\mathcal{P}_F^\star, \mathcal{P}_M^\star)$ values. Indicate, with a vertical line, the observed discrepancy calculated in i. Note that you may use the `mixRandomly` function from class.

Solution:

```
set.seed(341)
ratioTimes <- sapply(1:5000, FUN = function(...) {
    ratioSDTimes(mixRandomly(pop))
})

hist(ratioTimes, breaks = 20, main = "Randomly Mixed Populations", xlab = "Discrepancy D",
    col = "lightgrey")
abline(v = ratioSDTimes(pop), col = "darkgreen", lwd = 2)
```



iii. [1 point] Calculate the p-value associated with this test.

Solution:

```
mean(ratioTimes >= ratioSDTimes(pop))
```

```
## [1] 0.008
```

iv. [1 point] Based on the p-value calculated in iii. what do you conclude about the comparability of

these two populations? In other words, summarize your findings and draw a conclusion about the null hypothesis from part (b).

Solution: Our estimated p-value is 0.008. This means that we have strong evidence against the null hypothesis that female and male marathon times are indistinguishable, as evaluated by a comparison of standard deviations.