Math525-A2-Coding

Jiahang Wang

```r
library(readr)
library(ggplot2)
```

Question 1

```r
# load data
data <- read.csv("data.csv")

head(data)
```

```
##   X.1 X CODE_N      COMMUNE BVQ_N POPSDC99 LOG LOGVAC STRATLOG  surf_m2
## 1   1 1  31001      AGASSAC 31239      121  65      9        1  9511618
## 2   2 2  31002       AIGNES 31033      193  99      7        1 22017258
## 3   3 3  31003 AIGREFEUILLE 31044      577 179      1        2  4608949
## 4   4 4  31004  AYGUESVIVES 31582     1815 656     15        3 13328942
## 5   5 5  31005         ALAN 31028      299 163      9        2 11439648
## 6   6 6  31006       ALBIAC 31106      159  60      3        1  4778496
##   lat_centre lon_centre
## 1   43.37249  0.8861555
## 2   43.33456  1.5866078
## 3   43.56919  1.5851581
## 4   43.42864  1.5943056
## 5   43.21998  0.9270756
## 6   43.55600  1.7832118
```

meaning of the covariates:

- Unnamed: 0: An index or identifier for each row.
- X: Appears to be another form of index or identifier.
- CODE_N: Municipality code.
- COMMUNE: Name of the municipality.
- BVQ_N: Code for the Daily Life Basin the municipality belongs to.
- POPSDC99: Number of inhabitants.

- LOG: Number of dwellings.
- LOGVAC: Number of vacant dwellings.
- STRATLOG: A classification based on the number of dwellings, with four categories reflecting different ranges of dwelling counts.
- surf_m2: Surface area in square meters.
- lat_centre: Geographical latitude of the center of the municipality.
- lon_centre: Geographical longitude of the center of the municipality.

main features of the covariates:

The first part is the data type with some example data of each covariates, and the second part is some basic statistical quantity of each covariates.

```r
# View the structure of the dataset
str(data)
```

```
## 'data.frame':    500 obs. of  12 variables:
##  $ X.1       : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ X         : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ CODE_N    : int  31001 31002 31003 31004 31005 31006 31007 31008 31009 31010 ...
##  $ COMMUNE   : chr  "AGASSAC" "AIGNES" "AIGREFEUILLE" "AYGUESVIVES" ...
##  $ BVQ_N     : int  31239 31033 31044 31582 31028 31106 31239 31239 31483 31042 ...
##  $ POPSDC99  : int  121 193 577 1815 299 159 72 230 84 100 ...
##  $ LOG       : int  65 99 179 656 163 60 37 120 77 76 ...
##  $ LOGVAC    : int  9 7 1 15 9 3 4 5 3 0 ...
##  $ STRATLOG  : int  1 1 2 3 2 1 1 2 1 1 ...
##  $ surf_m2   : int  9511618 22017258 4608949 13328942 11439648 4778496 5921731 13561175 4450033 5837
##  $ lat_centre: num  43.4 43.3 43.6 43.4 43.2 ...
##  $ lon_centre: num  0.886 1.587 1.585 1.594 0.927 ...
```

```r
# Get summary statistics
summary(data)
```

```
##       X.1              X              CODE_N        COMMUNE         
##  Min.   :  1.0   Min.   :  1.0   Min.   :31001   Length:500        
##  1st Qu.:125.8   1st Qu.:125.8   1st Qu.:31138   Class :character  
##  Median :250.5   Median :250.5   Median :31271   Mode  :character  
##  Mean   :250.5   Mean   :250.5   Mean   :31270                     
##  3rd Qu.:375.2   3rd Qu.:375.2   3rd Qu.:31403                     
##  Max.   :500.0   Max.   :500.0   Max.   :31535                     
##      BVQ_N          POPSDC99           LOG             LOGVAC      
##  Min.   :31020   Min.   :   8.0   Min.   :  13.0   Min.   :  0.00  
##  1st Qu.:31106   1st Qu.: 108.8   1st Qu.:  66.0   1st Qu.:  4.00  
##  Median :31239   Median : 265.0   Median : 140.0   Median :  8.00  
##  Mean   :31294   Mean   : 822.6   Mean   : 353.1   Mean   : 19.47  
##  3rd Qu.:31483   3rd Qu.: 815.0   3rd Qu.: 354.8   3rd Qu.: 20.00  
##  Max.   :31584   Max.   :8733.0   Max.   :4490.0   Max.   :350.00  
##     STRATLOG        surf_m2           lat_centre      lon_centre    
##  Min.   :1.000   Min.   : 1241201   Min.   :42.73   Min.   :0.4710  
##  1st Qu.:1.000   1st Qu.: 4919457   1st Qu.:43.14   1st Qu.:0.7786  
##  Median :2.000   Median : 7586664   Median :43.34   Median :1.0900  
##  Mean   :1.992   Mean   :10459298   Mean   :43.33   Mean   :1.1244  
##  3rd Qu.:3.000   3rd Qu.:12686390   3rd Qu.:43.51   3rd Qu.:1.4671  
##  Max.   :4.000   Max.   :59911212   Max.   :43.90   Max.   :1.9967
```

Check missing data

```r
# Count missing values in each column
sapply(data, function(x) sum(is.na(x)))
```

```
##        X.1          X     CODE_N    COMMUNE      BVQ_N   POPSDC99        LOG
##          0          0          0          0          0          0          0
##     LOGVAC   STRATLOG    surf_m2 lat_centre lon_centre
##          0          0          0          0          0
```

As we can see, there are no missing data in the file

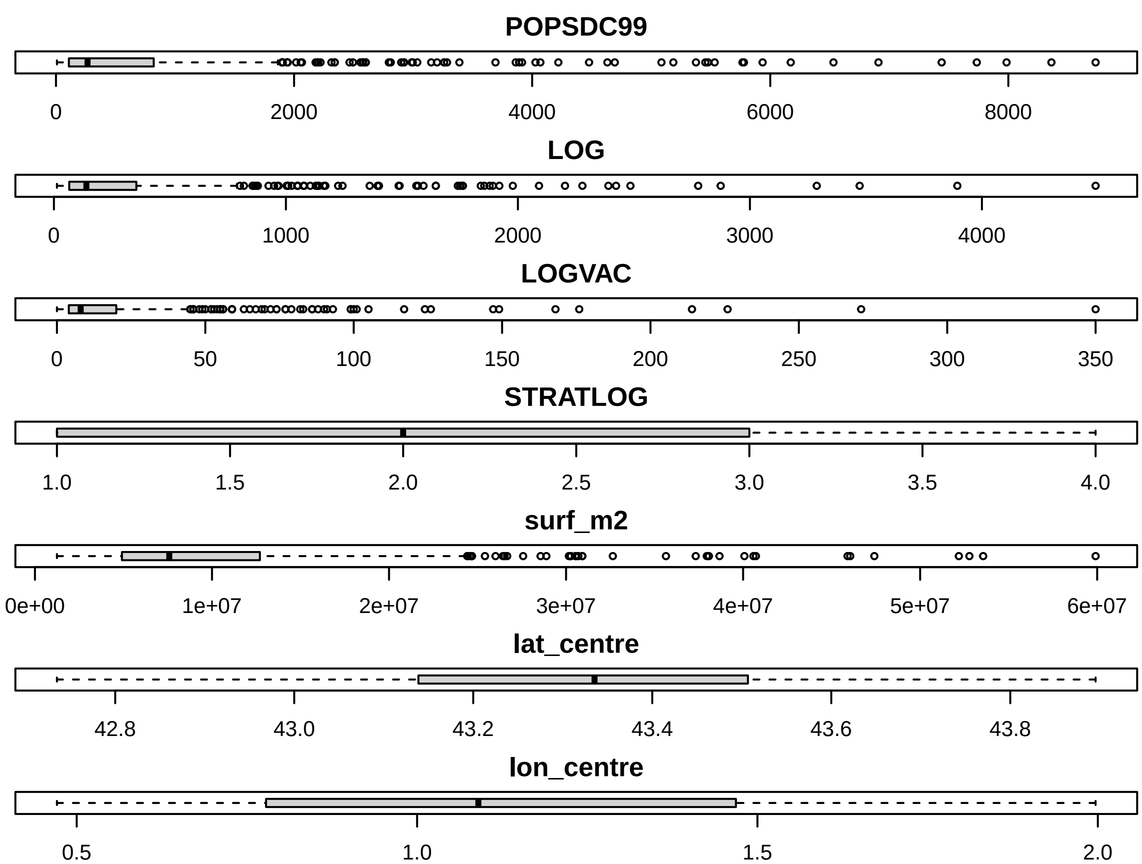
Check outliers

```r
# List of columns to plot
columns_to_plot <- c("POPSDC99", "LOG", "LOGVAC", "STRATLOG", "surf_m2", "lat_centre", "lon_centre")

par(mfrow=c(7, 1), mar=c(2, 4, 2, 1), las=1)


for (col in columns_to_plot) {

  boxplot(data[[col]], main=paste(col), horizontal=TRUE, las=1, xlab="")
}
```




```r
par(mfrow=c(1, 1), mar=c(5, 4, 4, 2) + 0.1)
```

Here I made the boxplot for the 7 numerical columns(excluding the code or index columns) to check their outliers. It shows that there are some outliers in columns [POPSDC99, LOG, LOGVAC, surf_m2] which are much higher than their mean value.

Question 2

```r
# Calculate 'Vacant_rate' as LOGVAC divided by LOG
data$Vacant_rate <- data$LOGVAC / data$LOG
data$more_than_10 <- ifelse(data$Vacant_rate > 0.1, 1, 0)

head(data$Vacant_rate)
```

```
## [1] 0.138461538 0.070707071 0.005586592 0.022865854 0.055214724 0.050000000
```

```r
head(data$more_than_10)
```

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

2.a

```r
# true mean
true_mean = mean(data$more_than_10)

set.seed(123) # For reproducibility
N <- length(data$more_than_10) # Population size
n <- 100 # Desired sample size
p <- n / N
num_simulations <- 10000

# Initialize vectors to store
mean_estimates_Srswor <- numeric(num_simulations)
mean_estimates_Bernoulli <- numeric(num_simulations)
mean_estimates_Systematic <- numeric(num_simulations)

square_error_Srswor <- numeric(num_simulations)
square_error_Bernoulli <- numeric(num_simulations)
square_error_Systematic <- numeric(num_simulations)

for (i in 1:num_simulations) {
  # SRSWOR Sampling
  sample_Srswor <- sample(data$more_than_10, n, replace = FALSE)
  mean_estimates_Srswor[i] <- mean(sample_Srswor)
  square_error_Srswor[i] <- (mean(sample_Srswor) - true_mean)^2

  # Bernoulli Sampling
  indices_Bernoulli <- rbinom(N, 1, p) == 1
  sample_Bernoulli <- data$more_than_10[indices_Bernoulli]
  ht_estimate_Bernoulli <- sum(sample_Bernoulli / p) / N
  mean_estimates_Bernoulli[i] <- ht_estimate_Bernoulli
  square_error_Bernoulli[i] <- (mean(ht_estimate_Bernoulli) - true_mean)^2

  # Systematic Sampling
  start <- sample(1:(N/n), 1)
```

```r
  indices_Systematic <- seq(from = start, by = N/n, length.out = n)
  sample_Systematic <- data$more_than_10[indices_Systematic]
  mean_estimates_Systematic[i] <- mean(sample_Systematic)
  square_error_Systematic[i] <- (mean(sample_Systematic) - true_mean)^2
}


mean_HT_estimator_Srswor <- mean(mean_estimates_Srswor)
mean_HT_estimator_Bernoulli <- mean(mean_estimates_Bernoulli)
mean_HT_estimator_Systematic <- mean(mean_estimates_Systematic)

MSE_Srswor <- mean(square_error_Srswor)
MSE_Bernoulli <- mean(square_error_Bernoulli)
MSE_Systematic <- mean(square_error_Systematic)

# cat("population mean:", true_mean, "\n\n")
# cat("HT for SRSWOR:", mean_HT_estimator_Srswor, "\n")
# cat("HT for Bernoulli:", mean_HT_estimator_Bernoulli, "\n")
# cat("HT for Systematic:", mean_HT_estimator_Systematic, "\n\n")

# cat("bias for SRSWOR:", true_mean - mean_HT_estimator_Srswor, "\n")
# cat("bias for Bernoulli:", true_mean - mean_HT_estimator_Bernoulli, "\n")
# cat("bias for Systematic:", true_mean - mean_HT_estimator_Systematic, "\n\n")

cat("MSE for SRSWOR:", MSE_Srswor, "\n")
```

```
## MSE for SRSWOR: 0.00129213
```

```r
cat("MSE for Bernoulli:", MSE_Bernoulli, "\n")
```

```
## MSE for Bernoulli: 0.00165195
```

```r
cat("MSE for Systematic:", MSE_Systematic, "\n")
```

```
## MSE for Systematic: 0.001394086
```

The MSE of each sampling method is calculated as:

$\text{MSE} = \frac{1}{R}\sum_{i=1}^{R}(\hat{\mu}_{\pi,i} - \mu)^2$

As shown by the MSE result from the Monte-Carlo simulation. The Simple Random Sampling Without Replacement (SRSWOR) strategy is the best among the three in this population, as it has the lowest mean square error (MSE) of 0.00129213, indicating the highest estimation accuracy among the three sampling approach.

2.b

We implement the SRSWOR approach as suggested above to randomly select a sample, and use the HT estimator to estimate the proportion of cities having more than 10% of vacant wellings which is approximately 20%. This is very close to the true population proportion 20.4%

```r
# true mean
true_mean = mean(data$more_than_10)

set.seed(123) # Ensure reproducibility

# SRSWOR
sample_Srswor <- sample(data$more_than_10, size = 100, replace = FALSE)

HT_estimate = mean(sample_Srswor)

cat("population proportion:", true_mean, "\n")
```

```
## population proportion: 0.204
```

```r
cat("estimated proportion:", HT_estimate)
```

```
## estimated proportion: 0.2
```

Question 3

3.a

```r
# true mean
true_mean = mean(data$LOGVAC)

set.seed(123) # For reproducibility
N <- length(data$LOGVAC) # Population size
n <- 100 # Desired sample size
p <- n / N
num_simulations <- 10000

# Initialize vectors to store
mean_estimates_Srswor <- numeric(num_simulations)
mean_estimates_Bernoulli <- numeric(num_simulations)
mean_estimates_Systematic <- numeric(num_simulations)

square_error_Srswor <- numeric(num_simulations)
square_error_Bernoulli <- numeric(num_simulations)
square_error_Systematic <- numeric(num_simulations)

for (i in 1:num_simulations) {
  # SRSWOR Sampling
  sample_Srswor <- sample(data$LOGVAC, n, replace = FALSE)
  mean_estimates_Srswor[i] <- mean(sample_Srswor)
  square_error_Srswor[i] <- (mean(sample_Srswor) - true_mean)^2

  # Bernoulli Sampling
  indices_Bernoulli <- rbinom(N, 1, p) == 1
  sample_Bernoulli <- data$LOGVAC[indices_Bernoulli]
```

```r
  ht_estimate_Bernoulli <- sum(sample_Bernoulli / p) / N
  mean_estimates_Bernoulli[i] <- ht_estimate_Bernoulli
  square_error_Bernoulli[i] <- (mean(ht_estimate_Bernoulli) - true_mean)^2

  # Systematic Sampling
  start <- sample(1:(N/n), 1)
  indices_Systematic <- seq(from = start, by = N/n, length.out = n)
  sample_Systematic <- data$LOGVAC[indices_Systematic]
  mean_estimates_Systematic[i] <- mean(sample_Systematic)
  square_error_Systematic[i] <- (mean(sample_Systematic) - true_mean)^2
}


mean_HT_estimator_Srswor <- mean(mean_estimates_Srswor)
mean_HT_estimator_Bernoulli <- mean(mean_estimates_Bernoulli)
mean_HT_estimator_Systematic <- mean(mean_estimates_Systematic)

MSE_Srswor <- mean(square_error_Srswor)
MSE_Bernoulli <- mean(square_error_Bernoulli)
MSE_Systematic <- mean(square_error_Systematic)

# cat("population mean:", true_mean, "\n\n")
# cat("HT for SRSWOR:", mean_HT_estimator_Srswor, "\n")
# cat("HT for Bernoulli:", mean_HT_estimator_Bernoulli, "\n")
# cat("HT for Systematic:", mean_HT_estimator_Systematic, "\n\n")
#
# cat("bias for SRSWOR:", true_mean - mean_HT_estimator_Srswor, "\n")
# cat("bias for Bernoulli:", true_mean - mean_HT_estimator_Bernoulli, "\n")
# cat("bias for Systematic:", true_mean - mean_HT_estimator_Systematic, "\n\n")

cat("MSE for SRSWOR:", MSE_Srswor, "\n")
```

```
## MSE for SRSWOR: 9.00325
```

```r
cat("MSE for Bernoulli:", MSE_Bernoulli, "\n")
```

```
## MSE for Bernoulli: 12.15968
```

```r
cat("MSE for Systematic:", MSE_Systematic, "\n")
```

```
## MSE for Systematic: 11.26292
```

The MSE of each sampling method is calculated as:

$\text{MSE} = \frac{1}{R}\sum_{i=1}^{R}(\hat{\mu}_{\pi,i} - \mu)^2$

As shown by the MSE result from the Monte-Carlo simulation. The Simple Random Sampling Without Replacement (SRSWOR) strategy is the best among the three in this population, as it has the lowest mean square error (MSE) of 9.00325, indicating the highest estimation accuracy among the three sampling approach.

3.b

```r
# sort the data
data_new <- data[order(data$LOG), ]

# View
head(data_new)
```

```
##     X.1   X CODE_N     COMMUNE BVQ_N POPSDC99 LOG LOGVAC STRATLOG surf_m2
## 42   42  42  31046       BAREN 31042        8  13      1        1 2937107
## 116 116 116  31127      CAUBOUS 31042       12  16      0        1 3964224
## 288 288 288  31312   MAILHOLAS 31107       38  16      2        1 3018467
## 58   58  58  31062 BELLESSERRE 31232       50  17      1        1 3446412
## 204 204 204  31223      GOUDEX 31239       43  19      1        1 2634934
## 326 326 326  31353       MONES 31454       50  21      0        1 2498757
##     lat_centre lon_centre Vacant_rate more_than_10
## 42    42.86707  0.6345736  0.07692308            0
## 116   42.84701  0.5224892  0.00000000            0
## 288   43.24572  1.2504575  0.12500000            1
## 58    43.78745  1.1019859  0.05882353            0
## 204   43.37897  0.9570148  0.05263158            0
## 326   43.41813  1.0320877  0.00000000            0
```

```r
# true mean
true_mean = mean(data_new$LOGVAC)

set.seed(123) # For reproducibility
N <- length(data_new$LOGVAC) # Population size
n <- 100 # Expected sample size
p <- n / N
num_simulations <- 10000

# Initialize vectors to store
mean_estimates_Srswor <- numeric(num_simulations)
mean_estimates_Bernoulli <- numeric(num_simulations)
mean_estimates_Systematic <- numeric(num_simulations)

square_error_Srswor <- numeric(num_simulations)
square_error_Bernoulli <- numeric(num_simulations)
square_error_Systematic <- numeric(num_simulations)

for (i in 1:num_simulations) {
  # SRSWOR Sampling
  sample_Srswor <- sample(data_new$LOGVAC, n, replace = FALSE)
  mean_estimates_Srswor[i] <- mean(sample_Srswor)
  square_error_Srswor[i] <- (mean(sample_Srswor) - true_mean)^2

  # Bernoulli Sampling
  indices_Bernoulli <- rbinom(N, 1, p) == 1
  sample_Bernoulli <- data_new$LOGVAC[indices_Bernoulli]
  ht_estimate_Bernoulli <- sum(sample_Bernoulli / p) / N
  mean_estimates_Bernoulli[i] <- ht_estimate_Bernoulli
```

```r
  square_error_Bernoulli[i] <- (mean(ht_estimate_Bernoulli) - true_mean)^2

  # Systematic Sampling
  start <- sample(1:(N/n), 1)
  indices_Systematic <- seq(from = start, by = N/n, length.out = n)
  sample_Systematic <- data_new$LOGVAC[indices_Systematic]
  mean_estimates_Systematic[i] <- mean(sample_Systematic)
  square_error_Systematic[i] <- (mean(sample_Systematic) - true_mean)^2
}


mean_HT_estimator_Srswor <- mean(mean_estimates_Srswor)
mean_HT_estimator_Bernoulli <- mean(mean_estimates_Bernoulli)
mean_HT_estimator_Systematic <- mean(mean_estimates_Systematic)

MSE_Srswor <- mean(square_error_Srswor)
MSE_Bernoulli <- mean(square_error_Bernoulli)
MSE_Systematic <- mean(square_error_Systematic)

# cat("population mean:", true_mean, "\n\n")
# cat("HT for SRSWOR:", mean_HT_estimator_Srswor, "\n")
# cat("HT for Bernoulli:", mean_HT_estimator_Bernoulli, "\n")
# cat("HT for Systematic:", mean_HT_estimator_Systematic, "\n\n")
#
# cat("bias for SRSWOR:", true_mean - mean_HT_estimator_Srswor, "\n")
# cat("bias for Bernoulli:", true_mean - mean_HT_estimator_Bernoulli, "\n")
# cat("bias for Systematic:", true_mean - mean_HT_estimator_Systematic, "\n\n")

cat("MSE for SRSWOR:", MSE_Srswor, "\n")
```

```
## MSE for SRSWOR: 9.297134
```

```r
cat("MSE for Bernoulli:", MSE_Bernoulli, "\n")
```

```
## MSE for Bernoulli: 11.94096
```

```r
cat("MSE for Systematic:", MSE_Systematic, "\n")
```

```
## MSE for Systematic: 2.990881
```

As shown by the MSE result from the Monte-Carlo simulation using the sorted data this time. The Systematic Sampling strategy is the best among the three in this population, as it has the lowest mean square error (MSE) of 2.990881, indicating the highest estimation accuracy among the three sampling approach.

3.c

- Yes, the results(MSE) differ significantly after sorting the population by the Number of dwellings for Systematic sampling design.
- The efficiencies of the three strategies changed, with Systematic sampling showing a substantial improvement in accuracy, evidenced by the lowest mean square error (MSE) after sorting.

- This change highlights the impact of population order on sampling strategies, particularly benefiting Systematic sampling due to its reliance on the order for selection. Thus given the mechanics of the sampling methods, the improvement in Systematic Sampling's efficiency after sorting the population by a relevant characteristic can be seen as predictable. Sorting introduces a form of order that Systematic Sampling can exploit, leading to more representative samples and MSE.
- from the perspective of ANOVA decomposition of systematic sampling: "Total variations" = "Between group variations" + "Within group variations". Sorting increase the Within group variations. Since Total variations doesn't change. Between group variations reduces which also leads to the decrease of MSE.