Week_7

Hu Tianao

2023-11-15

1.1 Q1

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(Stat2Data)
data("Hawks")
RedTailedDf <- Hawks %>%
  filter(Species == 'RT') %>%
  select(Weight, Tail, Wing)
head(RedTailedDf)
```

```
##   Weight Tail Wing
## 1    920  219  385
## 2    930  221  376
## 3    990  235  381
## 4   1090  230  412
## 5    960  212  370
## 6    855  243  375
```

1.1 Q2

```
library(dplyr)
data <- RedTailedDf$Tail
mu_hat_mle <- mean(data)
sigma2_hat_mle <- mean((data - mu_hat_mle)^2)
mu_hat_mle
```

```
## [1] 222.149
```

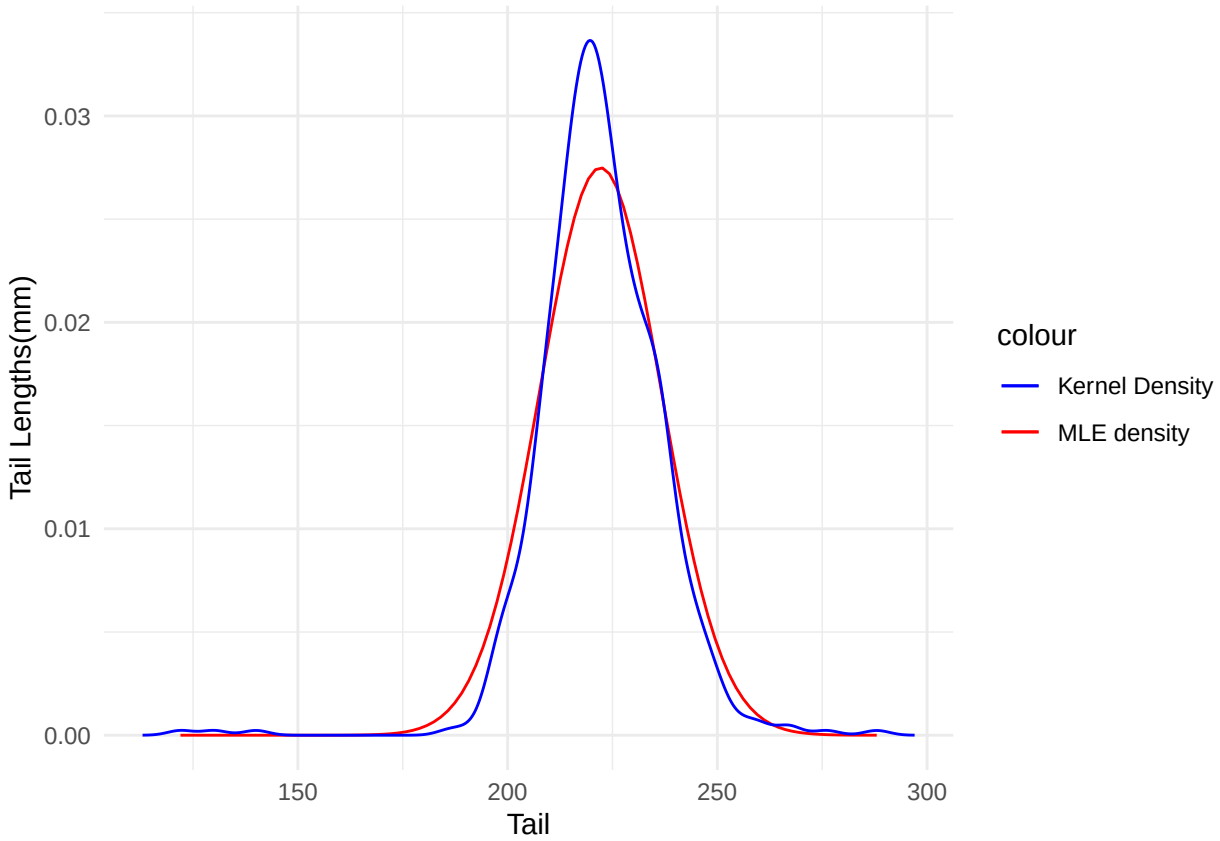
```
sigma2_hat_mle
```

```
## [1] 210.2031
```

1.1 Q3

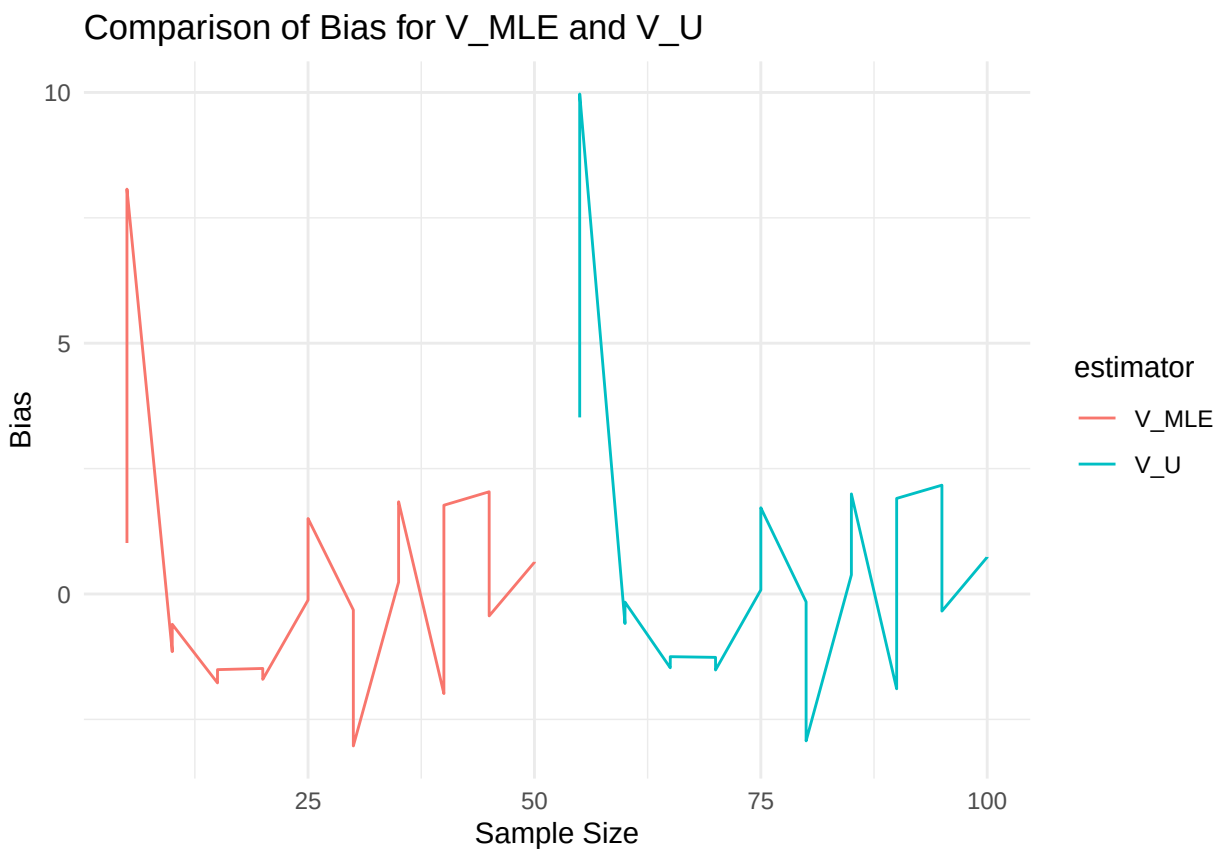
```r
library(ggplot2)

mu_hat_mle <- mean(data)
sigma2_hat_mle <- var(data)

x_vals <- seq(min(data), max(data), length.out = 100)
pdf_vals <- dnorm(x_vals, mean = mu_hat_mle, sd = sqrt(sigma2_hat_mle))

density_vals <- density(data)

ggplot() +
  geom_line(aes(x = x_vals, y = pdf_vals, color = "MLE density"), size = 0.5) +
  geom_line(aes(x = density_vals$x, y = density_vals$y, color = "Kernel Density"), size = 0.5) +
  scale_color_manual(values = c("MLE density" = "red", "Kernel Density" = "blue")) +
  xlab("Tail") +
  ylab("Tail Lengths(mm)") +
  theme_minimal()
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



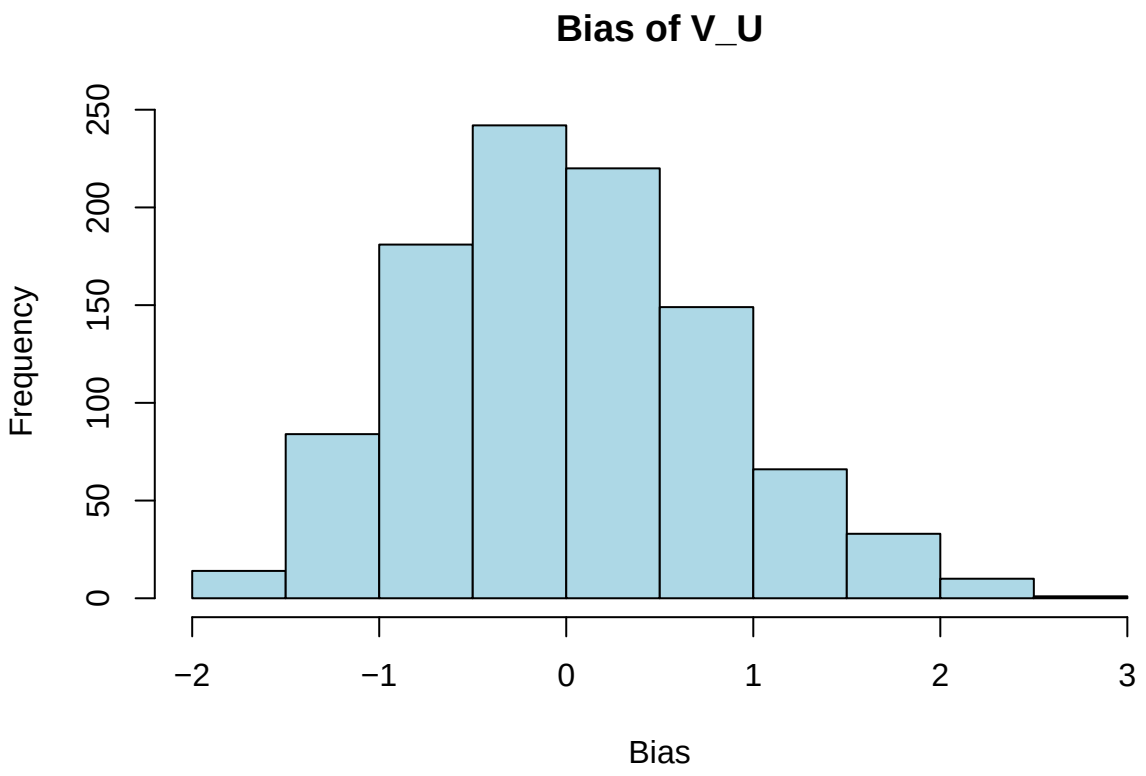
1.2 Q1

```
size <- seq(5, 100, by = 5)
VMLE <- vector()
VU <- vector()

for (i in size) {
  l <- rnorm(i, mean = 1, sd = 3)
  V_MLE <- var(l)
  V_U <- V_MLE * i / (i - 1)
  VMLE <- append(VMLE, V_MLE)
  VU <- append(VU, V_U)
}

# Calculate biases
bias_V_MLE <- VMLE - 3^2
bias_V_U <- VU - 3^2

# Create a data frame for plotting
results <- data.frame(sample_size = rep(size, each = 2),
                      bias = c(bias_V_MLE, bias_V_U),
                      estimator = rep(c("V_MLE", "V_U"), each = length(size)))

# Plot
library(ggplot2)

ggplot(results, aes(x = sample_size, y = bias, color = estimator)) +
  geom_line() +
  ggtitle("Comparison of Bias for V_MLE and V_U") +
  xlab("Sample Size") +
  ylab("Bias") +
  theme_minimal()
```



1.2 Q2

```r
set.seed(123)
n <- 50
mu <- 0
sigma <- 2

num_simulations <- 1000
bias_V_U <- numeric(num_simulations)

for (i in 1:num_simulations) {
  sample_data <- rnorm(n, mean = mu, sd = sigma)

  V_U <- sum((sample_data - mean(sample_data))^2) / (n - 1)
  true_variance <- sigma^2

  bias_V_U[i] <- V_U - true_variance
}

hist(bias_V_U, main = "Bias of V_U", xlab = "Bias", col = "lightblue", border = "black")
```



1.3

Q1

```
likelihood_function <- function(lambda, n, X) {
  exp(-n * lambda) * lambda^(n * X) / factorial(n)
}

log_likelihood_function <- function(lambda, n, X) {
  -n * lambda + n * X * log(lambda) - sum(log(1:n))
}

derivative_log_likelihood <- function(lambda, n, X) {
  -n + n * X / lambda
}

#
lambda_val <- 2.5
n_val <- 10
X_val <- 3.5

likelihood_value <- likelihood_function(lambda_val, n_val, X_val)
cat("Likelihood Value:", likelihood_value, "\n")
```

```
## Likelihood Value: 0.0003241718
```

```
log_likelihood_value <- log_likelihood_function(lambda_val, n_val, X_val)
cat("Log-Likelihood Value:", log_likelihood_value, "\n")
```

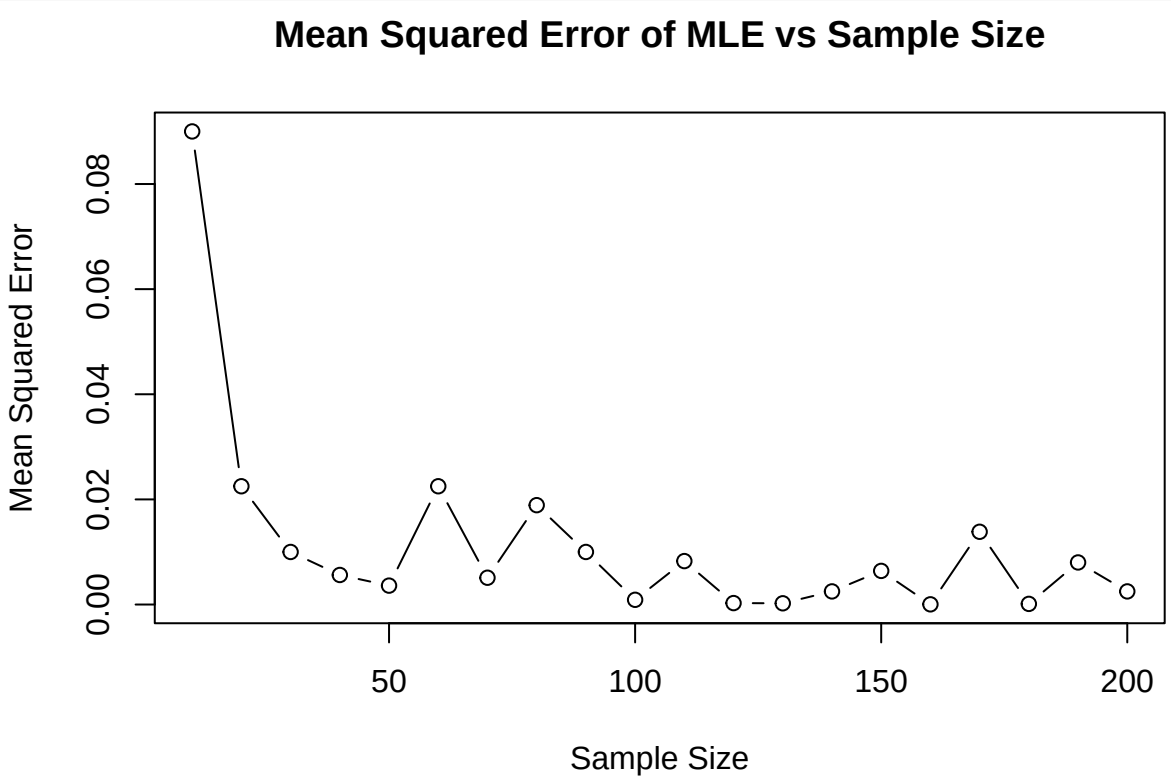
```
## Log-Likelihood Value: -8.034237
```

```
derivative_value <- derivative_log_likelihood(lambda_val, n_val, X_val)
cat("Derivative of Log-Likelihood Value:", derivative_value, "\n")
```

```
## Derivative of Log-Likelihood Value: 4
```

1.3 Q3

```r
simulate_and_calculate_mle <- function(lambda_true, sample_size) {

  data <- rpois(sample_size, lambda_true)

  mle <- mean(data)

  mse <- mean((mle - lambda_true)^2)

  return(mse)
}

lambda_true <- 0.5
sample_sizes <- seq(10, 200, by = 10)

mse_values <- sapply(sample_sizes, function(size) simulate_and_calculate_mle(lambda_true, size))
plot(sample_sizes, mse_values, type = "b",
     main = "Mean Squared Error of MLE vs Sample Size",
     xlab = "Sample Size", ylab = "Mean Squared Error")
```



##

1.3 Q4

```r
data <- read.csv("./VonBortkiewicz.csv")

sample_data <- data$fatalities

lambda_mle <- mean(sample_data)
```

```r
cat("Maximum Likelihood Estimate (MLE):", lambda_mle, "\n")
```

```
## Maximum Likelihood Estimate (MLE): 0.7
```

```r
prob_zero_fatalities <- dpois(0, lambda_mle)
cat("Probability of zero fatalities in a single year:", prob_zero_fatalities, "\n")
```

```
## Probability of zero fatalities in a single year: 0.4965853
```

1.4 Q2

```r
customer_data <- read.csv("./CustomerPurchase.csv")
customer_data$time_diffs <- c(diff(customer_data$PurchaseTime), NA)
head(customer_data)
```

```
##   Time Purchase time_diffs
## 1  564     3.25         NA
## 2  571   504.85         NA
## 3  578     7.60         NA
## 4  600    43.45         NA
## 5  745     9.30         NA
## 6  806   352.80         NA
```

1.4 Q3

```r
lambda_mle <- 1 / mean(customer_data$time_diffs, na.rm = TRUE)
cat("Maximum Likelihood Estimate (MLE) of rate parameter:", lambda_mle, "\n")
```

```
## Maximum Likelihood Estimate (MLE) of rate parameter: NaN
```

1.4 Q4

```r
prob_exceed_one_minute <- 1 - pexp(60, rate = lambda_mle)
cat("Probability of an arrival time in excess of one minute:", prob_exceed_one_minute, "\n")
```

```
## Probability of an arrival time in excess of one minute: NaN
```

2.1 Q2

```r
red_tailed_weights <- Hawks %>%
  filter(Species == 'RT') %>%
  pull(Weight) %>%
  na.omit()
alpha <- 0.01
sample_size_rt <- length(red_tailed_weights)
sample_mean_rt <- mean(red_tailed_weights)
sample_sd_rt <- sd(red_tailed_weights)
t_rt <- qt(1 - alpha / 2, df = sample_size_rt - 1)

confidence_interval_l_rt <- sample_mean_rt - t_rt * sample_sd_rt / sqrt(sample_size_rt)
confidence_interval_u_rt <- sample_mean_rt + t_rt * sample_sd_rt / sqrt(sample_size_rt)
confidence_interval_rt <- c(confidence_interval_l_rt, confidence_interval_u_rt)
```

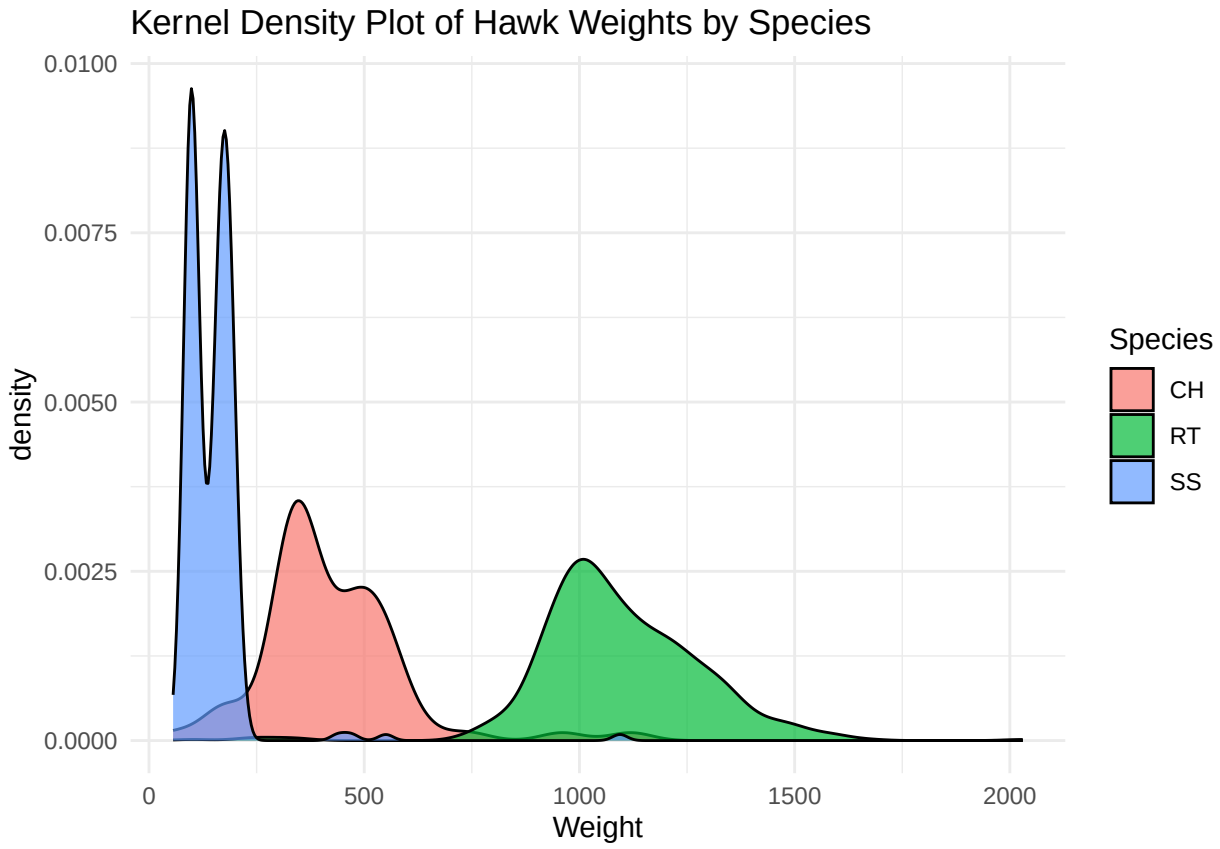
```
confidence_interval_rt
```

```
## [1] 1073.984 1114.877
```

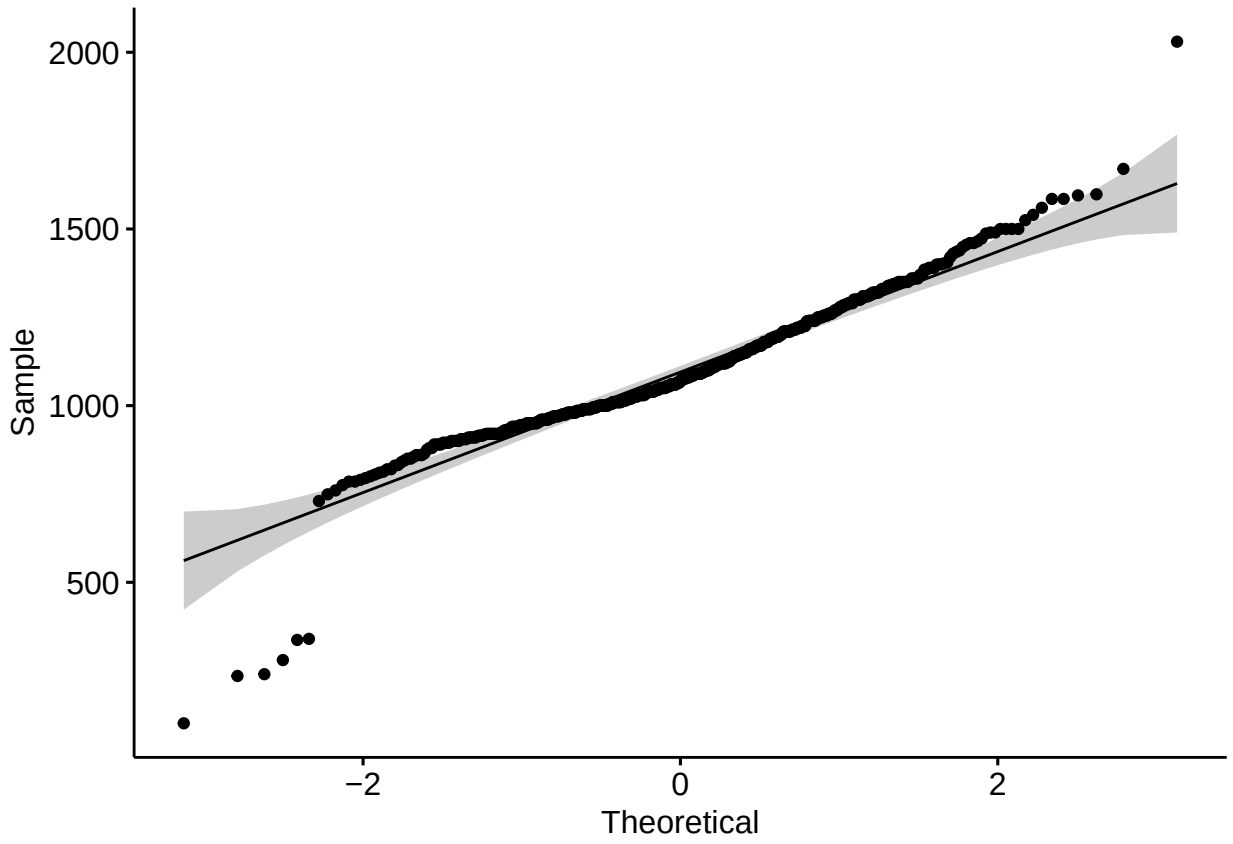
2.1 Q3

```
library(ggplot2)
library(ggpubr)

ggplot(data = Hawks, aes(x = Weight, fill = Species)) +
  geom_density(alpha = 0.7) +
  labs(title = "Kernel Density Plot of Hawk Weights by Species", x = "Weight") +
  theme_minimal()
```

```
## Warning: Removed 10 rows containing non-finite values (`stat_density()`).
```




```
qq_plot <- ggqqplot(red_tailed_weights, distribution = "norm")
print(qq_plot)
```



2.2 Q1

```r
# Function to calculate Student's t confidence interval
student_t_confidence_interval <- function(sample, confidence_level) {
  sample <- sample[!is.na(sample)]  # Remove any missing values
  n <- length(sample)  # Compute sample size
  mu_est <- mean(sample)  # Compute sample mean
  sig_est <- sd(sample)  # Compute sample sd
  alpha <- 1 - confidence_level  # Alpha from gamma
  t <- qt(1 - alpha / 2, df = n - 1)  # Get Student t quantile
  l <- mu_est - (t / sqrt(n)) * sig_est  # Lower
  u <- mu_est + (t / sqrt(n)) * sig_est  # Upper
  return(c(l, u))
}
```

2.2 Q2

```r
# Simulation for coverage property
num_trials <- 100000
sample_size <- 30
mu_0 <- 1
sigma_0 <- 3

# Generate random Gaussian samples
simulation_df <- data.frame(trial = seq(num_trials)) %>%
  mutate(sample = map(.x = trial, .f = ~rnorm(n = sample_size, mean = mu_0, sd = sigma_0))) %>%
  # Generate confidence intervals
  mutate(ci_interval = map(.x = sample, .f = ~student_t_confidence_interval(.x, 1 - alpha))) %>%
```

```r
  # Check if interval covers mu_0
  mutate(cover = map_lgl(.x = ci_interval, .f = ~((min(.x) <= mu_0) & (max(.x) >= mu_0))))

# Estimate of coverage probability
coverage_probability <- simulation_df %>%
  pull(cover) %>%
  mean()

coverage_probability
```

```
## [1] 0.99005
```

3.1 Q1

```r
library(palmerpenguins)

bill_adelie <- subset(penguins, species == "Adelie")$bill_length_mm

test_result <- t.test(bill_adelie, mu = 40, alternative = "two.sided", conf.level = 0.99)

cat("Test Statistic:", test_result$statistic, "\n")
```

```
## Test Statistic: -5.576185
```

```r
cat("P-value:", test_result$p.value, "\n")
```

```
## P-value: 1.114322e-07
```

```r
cat("Confidence Interval:", test_result$conf.int, "\n")
```

```
## Confidence Interval: 38.2259 39.35688
```

3.2 Q1

```r
one_sample_t_test <- function(x, mu0) {
  test_result <- t.test(x, mu = mu0, alternative = "two.sided")

  return(test_result$p.value)
}

library(palmerpenguins)

bill_adelie <- subset(penguins, species == "Adelie")$bill_length_mm

custom_test_p_value <- one_sample_t_test(bill_adelie, mu0 = 40)

t_test_result <- t.test(bill_adelie, mu = 40, alternative = "two.sided")

cat("Custom Function P-value:", custom_test_p_value, "\n")
```

```
## Custom Function P-value: 1.114322e-07
```

```r
cat("t.test() Function P-value:", t_test_result$p.value, "\n")
```

```
## t.test() Function P-value: 1.114322e-07
```