Assignment07_Template

Vishal

2022-11-16

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(Stat2Data)
data("Hawks")
```

1. Maximum likelihood estimates

1.1 (Q1)

```
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)

```r
RedTailedDfVector <- RedTailedDf$Tail
n <- length(RedTailedDfVector)
mu_mle <- mean(RedTailedDfVector)
sigma_mle <- sd(RedTailedDfVector) * sqrt((n - 1) / n)

mu_mle
```

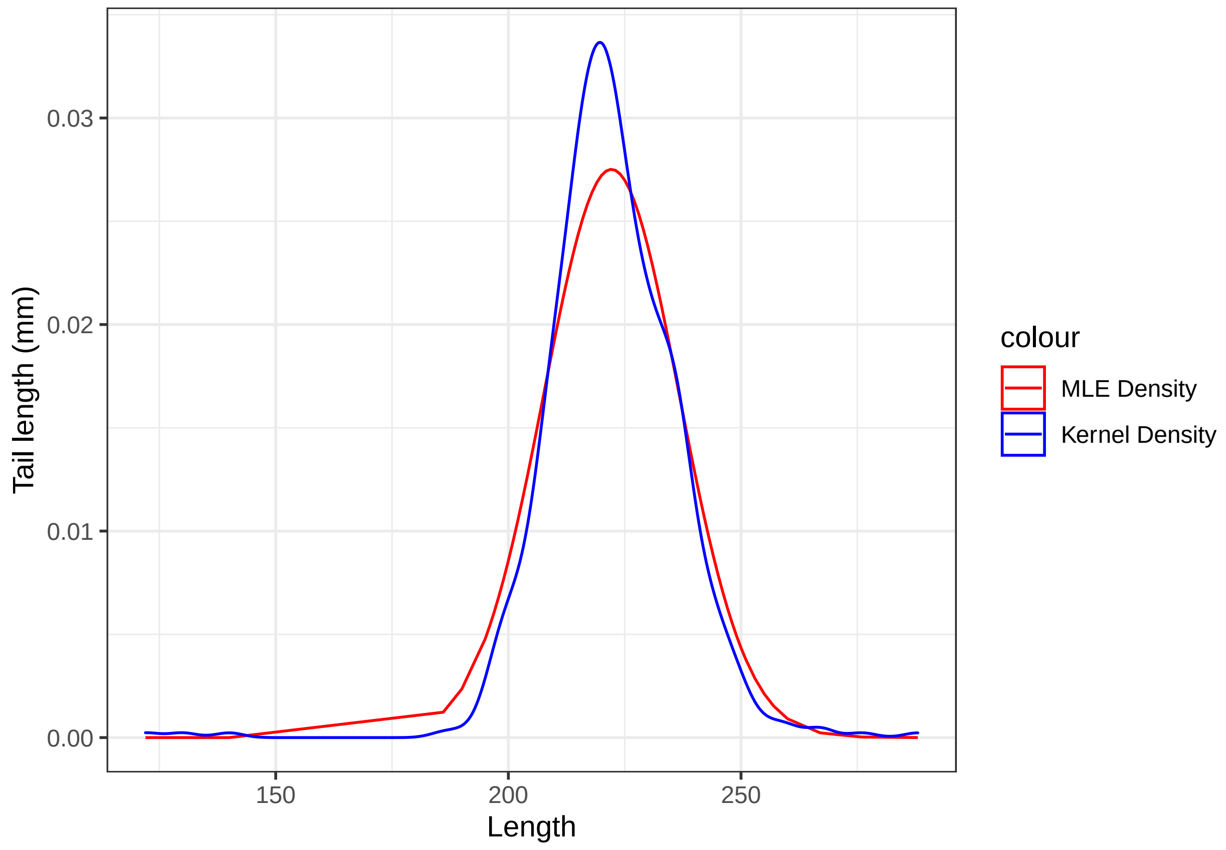
```
## [1] 222.149
```

```r
sigma_mle
```

```
## [1] 14.49838
```

1.1 (Q3)

```r
density_df <- data.frame(Length = RedTailedDfVector) %>%
  mutate(pdf = map_dbl(.x = Length, .f = ~dnorm(.x, mean = mu_mle, sd = sigma_mle)))


colors <- c("MLE Density" = "red", "Kernel Density" = "blue")

density_df %>%
  ggplot() +
  geom_line(aes(x = Length, y = pdf, color = "MLE Density")) +
  geom_density(aes(x = Length, color = "Kernel Density")) +
  scale_color_manual(values = colors) +
  labs(y = "Tail length (mm)") +
  theme_bw()
```

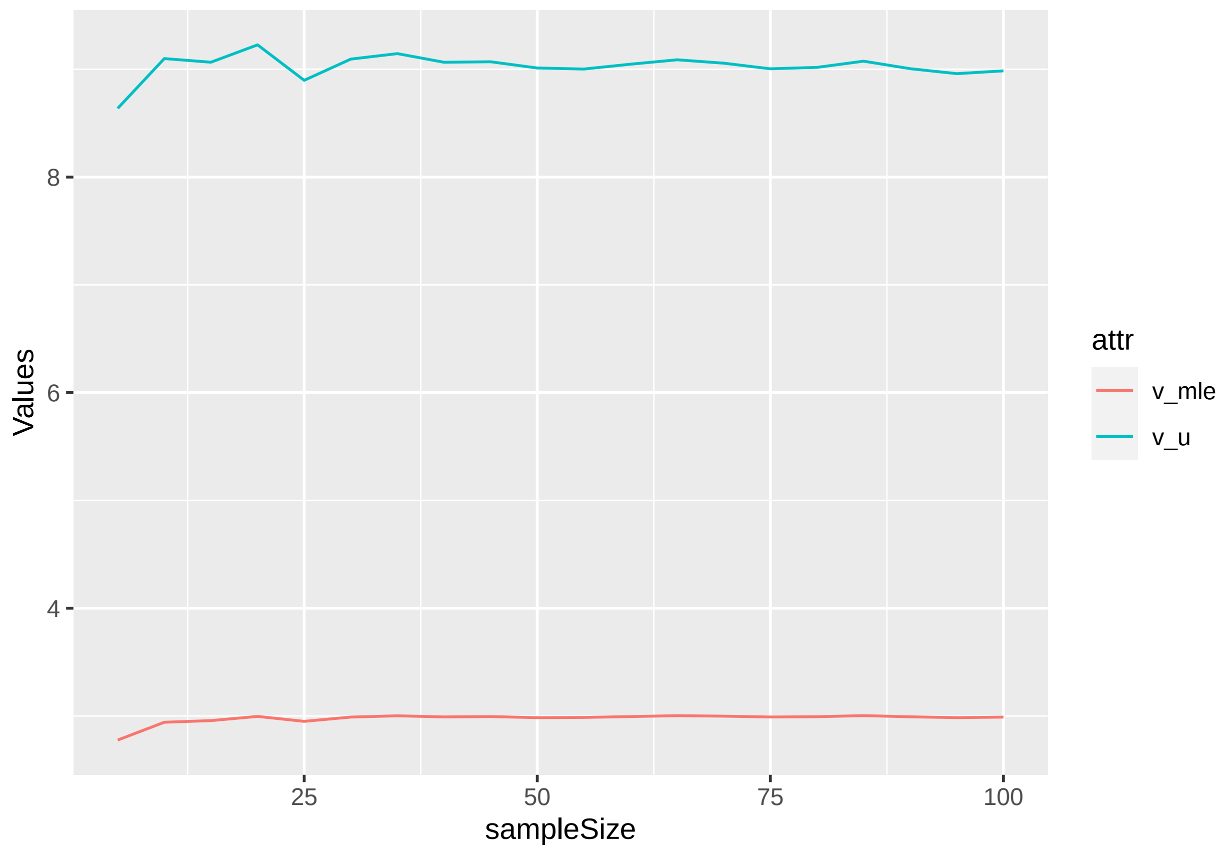


1.2 (Q1)

```
df <- data.frame(sampleSize = rep(seq(5, 100, 5), each = 1000)) %>%
  mutate(sample = map(.x = sampleSize, ~(rnorm(.x, 1, 3))),
         mu = map_dbl(.x = sample, .f = mean),
         v_mle = map_dbl(.x = sample, .f = sd),
         v_u = map_dbl(.x = sample, .f = var))

d <- df %>%
  group_by(sampleSize) %>%
  summarise(across(c(v_mle, v_u), mean)) %>%
  pivot_longer(!sampleSize, names_to = "attr", values_to = "Values")

head(d)
```

```
## # A tibble: 6 x 3
##   sampleSize attr  Values
##        <dbl> <chr>  <dbl>
## 1          5 v_mle   2.78
## 2          5 v_u     8.64
## 3         10 v_mle   2.94
## 4         10 v_u     9.10
## 5         15 v_mle   2.96
## 6         15 v_u     9.07
```

```
ggplot(d, aes(x = sampleSize, y = Values, color = attr)) +
  geom_line()
```

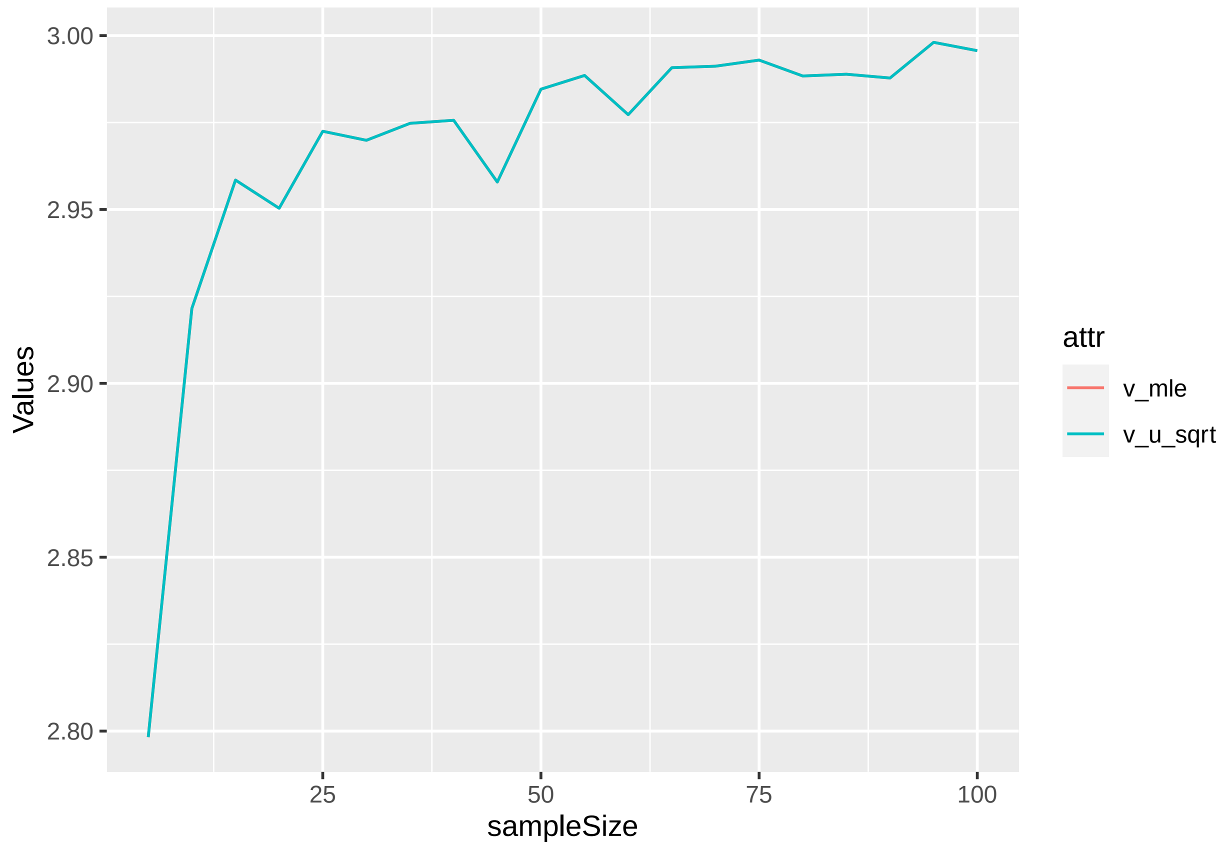


# 1.2 (Q2)

```
df <- data.frame(sampleSize = rep(seq(5, 100, 5), each = 1000)) %>%
  mutate(sample = map(.x = sampleSize, ~(rnorm(.x, 1, 3))),
         mu = map_dbl(.x = sample, .f = mean),
         v_mle = map_dbl(.x = sample, .f = sd),
         v_u = map_dbl(.x = sample, .f = var),
         v_u_sqrt = map_dbl(.x = v_u, .f = sqrt))

d <- df %>%
  group_by(sampleSize) %>%
  summarise(across(c(v_mle, v_u_sqrt), mean)) %>%
  pivot_longer(!sampleSize, names_to = "attr", values_to = "Values")

head(d)
```

```
## # A tibble: 6 x 3
##   sampleSize attr     Values
##        <dbl> <chr>     <dbl>
## 1          5 v_mle      2.80
## 2          5 v_u_sqrt   2.80
## 3         10 v_mle      2.92
## 4         10 v_u_sqrt   2.92
## 5          15 v_mle      2.96
## 6          15 v_u_sqrt   2.96
```

```
ggplot(d, aes(x = sampleSize, y = Values, color = attr)) +
  geom_line()
```



As it can be seen in the above graph, $\sqrt{Vu}$ is not an unbiased estimator of for σ_0. Calculating the sample variance treats the sample mean as the true mean, where there's an uncertainty about the true mean. This is why square root of variance is not a good estimator.

1.3 Maximum likelihood estimation with the Poisson distribution

1.3 (Q1)

$$f(x) = \frac{\lambda^x e^{-\lambda}}{x!} x = 0, 1, 2,$$

$$L(\lambda) = \prod_{i=1}^{n} \frac{\lambda^{x_i} e^{-\lambda}}{x_i!} = e^{-n\lambda} \frac{\lambda^{\sum_1^n x_i}}{\prod_{i=1}^n x_i}$$

$$lnL(\lambda) = -n\lambda + \sum_{1}^{n} x_i ln(\lambda) - ln\left(\prod_{i=1}^{n} x_i\right)$$

$$\frac{dlnL(\lambda)}{dp} = -n + \sum_{1}^{n} x_i \frac{1}{\lambda}$$

$$\hat{\lambda} = \frac{\sum_{i=1}^{n} x_i}{n}$$

1.3 (Q2)

Suppose that $X = (X_1, X_2, ..., X_n)$ are iid observations from a Poisson distribution with unknown parameter λ. The likelihood function is

$$L(\lambda) = \prod_{i=1}^{n} f(x_i; \lambda) = \prod_{i=1}^{n} \frac{\lambda^{x_i} e^{-\lambda}}{x_i!} = \frac{\lambda^{\sum_i x_i} e^{-n\lambda}}{x_1! x_2! \cdots x_n!}$$

The corresponding loglikelihood function is

$$\sum_{i=1}^{n} x_i \log \lambda - n\lambda - \sum_{i=1}^{n} x_i!$$

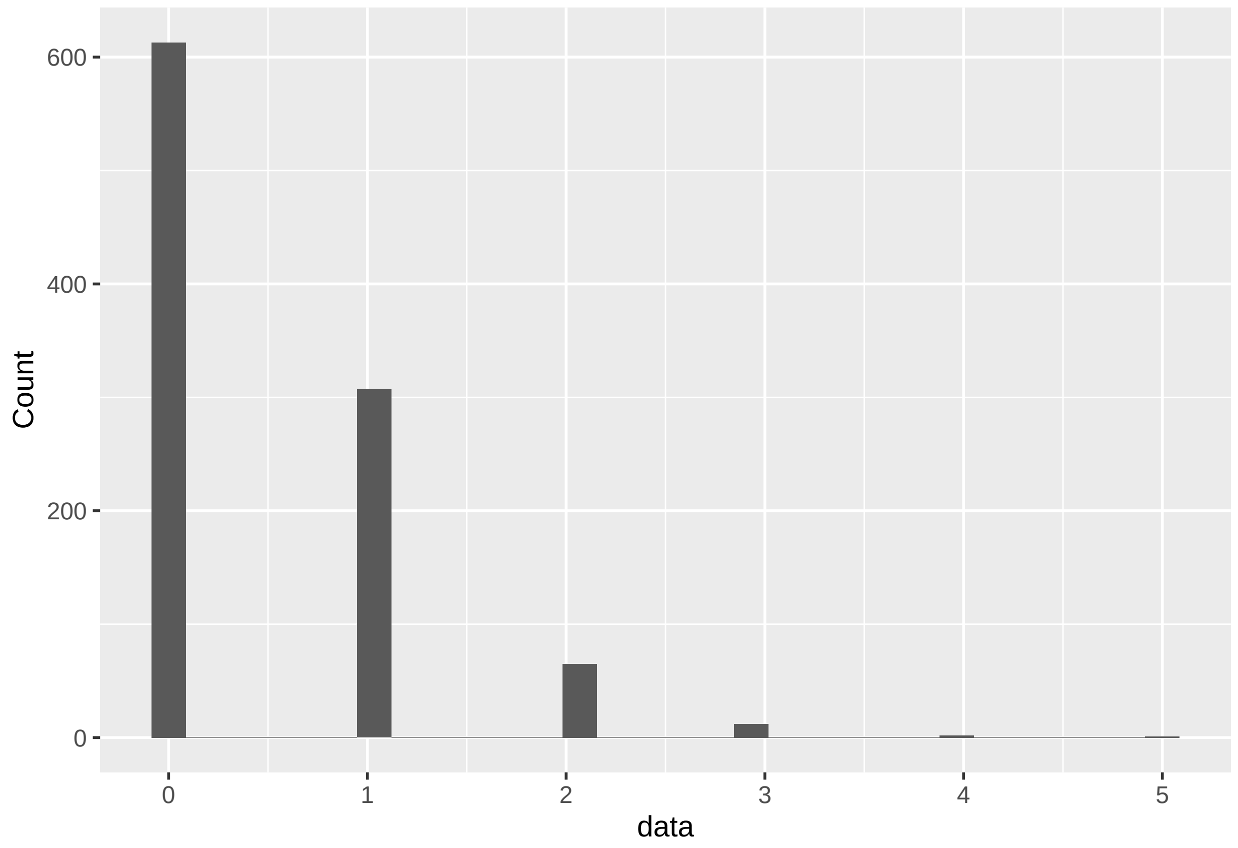
And the MLE for λ can then be found by maximizing either of these with respect to λ. Setting the first derivative equal to 0 gives the solution:

$$\hat{\lambda} = \sum_{i=1}^{n} \frac{x_i}{n}$$

Thus, for a Poisson sample, the MLE for λ is just the sample mean.

1.3 (Q3)

```r
dfPoisson <- data.frame(data = rpois(1000, 0.5))

dfPoisson %>%
  ggplot(aes(x = data)) +
  geom_histogram(bins = 30) +
  labs(y = "Count", x = "data")
```



# 1.3 (Q4)

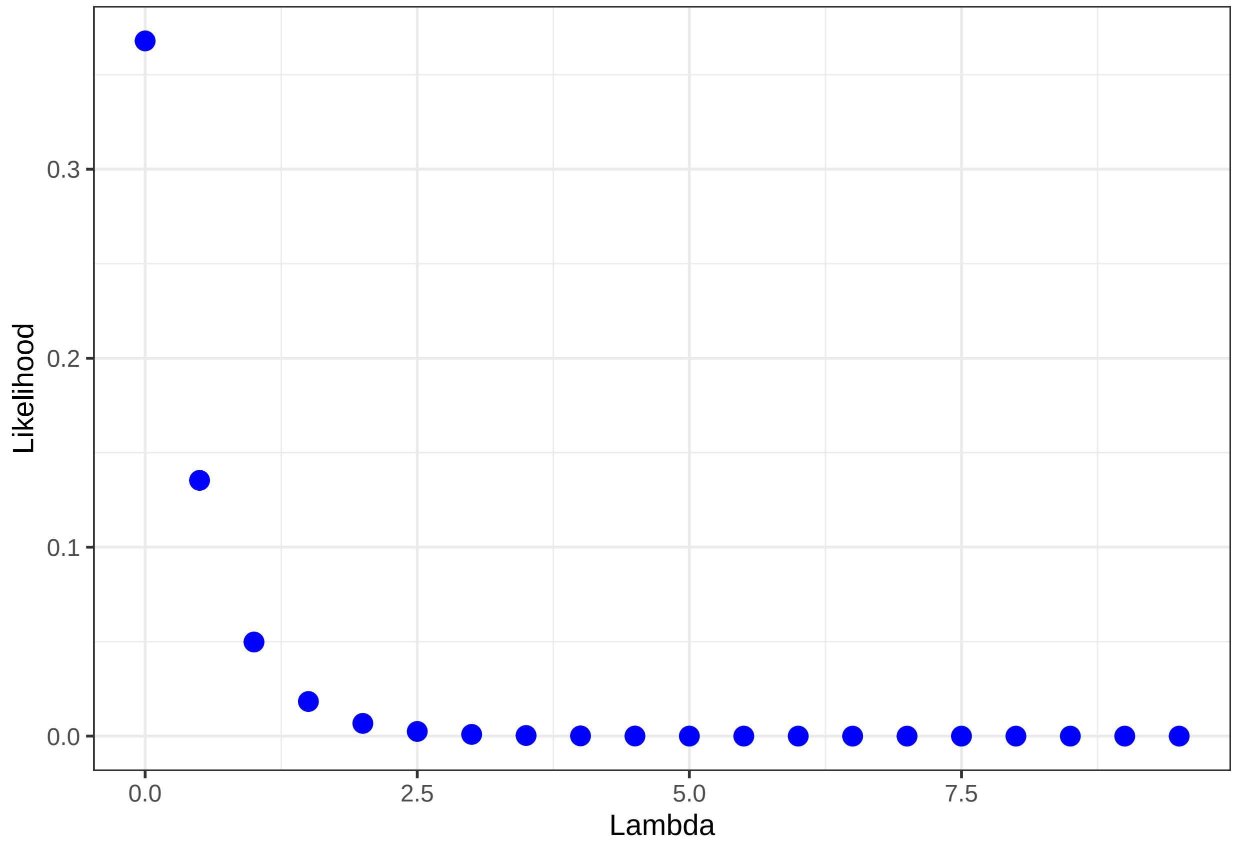
```
df_VonBortkiewicz <- read.csv("VonBortkiewicz.csv")
head(df_VonBortkiewicz)
```

```
##   fatalities year corps fisher
## 1          0 1875     G     no
## 2          0 1875     I     no
## 3          0 1875    II    yes
## 4          0 1875   III    yes
## 5          0 1875    IV    yes
## 6          0 1875     V    yes
```

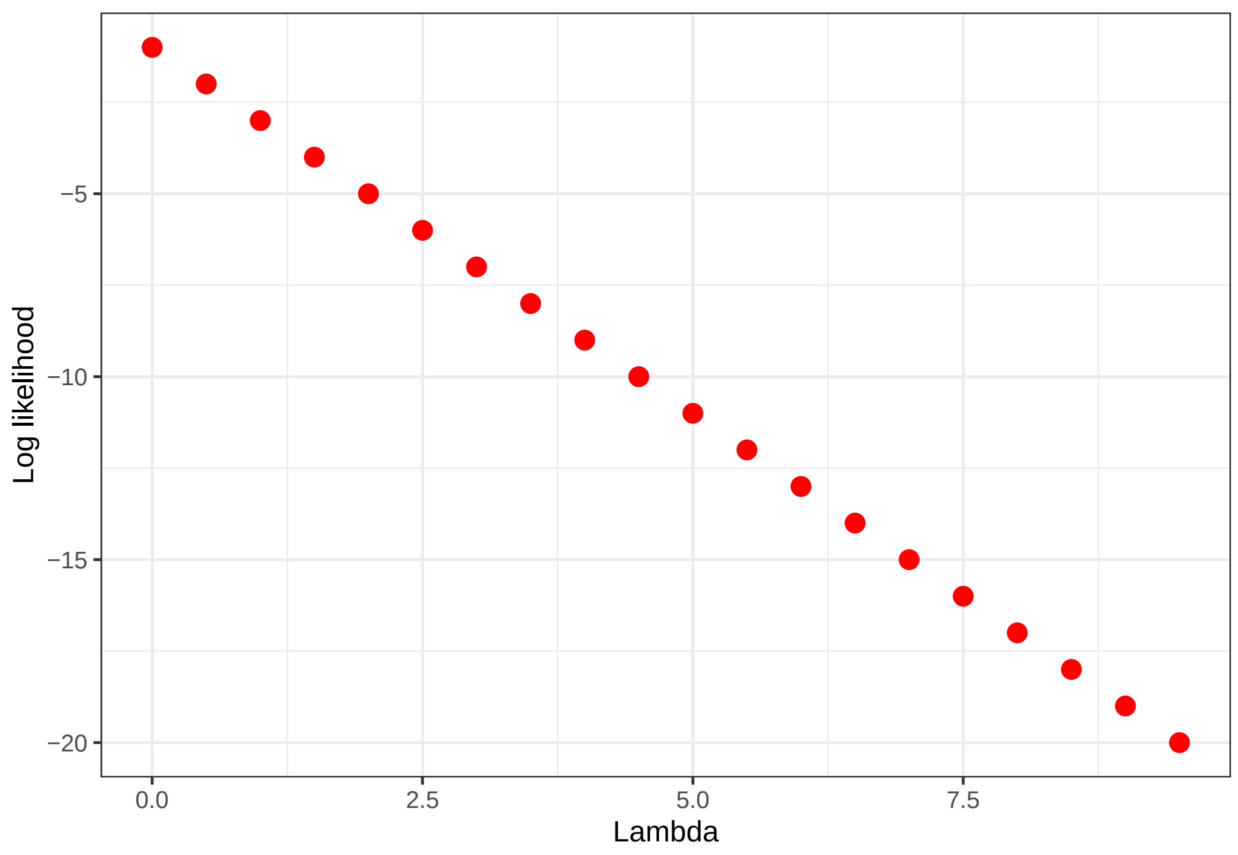
```
# likelihood of single data point
#likelihood <- data.frame(data = dpois(df_VonBortkiewicz$fatalities[1], seq(20)))

likelihood <- dpois(df_VonBortkiewicz$fatalities[1], seq(20))

lh_single <- data.frame(x = seq(0, 9.5, 0.5), likelihood = likelihood)

head(lh_single)
```

```
##     x  likelihood
## 1 0.0 0.367879441
## 2 0.5 0.135335283
## 3 1.0 0.049787068
## 4 1.5 0.018315639
## 5 2.0 0.006737947
## 6 2.5 0.002478752
```

```
lh_single %>%
  ggplot(aes(x = x, y = likelihood)) +
  geom_point(size = 3, color = "blue") +
  labs(x = "Lambda", y = "Likelihood") +
  theme_bw()
```



```
#ggplot(horseFatilities, aes(x = data)) +
#  geom_histogram(bins = 100)

## Log likelihood of single data point
log_likelihood <- dpois(df_VonBortkiewicz$fatalities[1], seq(20), log = T)
llh_single <- data.frame(x = seq(0, 9.5, 0.5), log_like = log_likelihood)

llh_single %>%
  ggplot(aes(x = x, y = log_like)) +
  geom_point(size = 3, color = "red") +
  labs(x = "Lambda", y = "Log likelihood") +
  theme_bw()
```

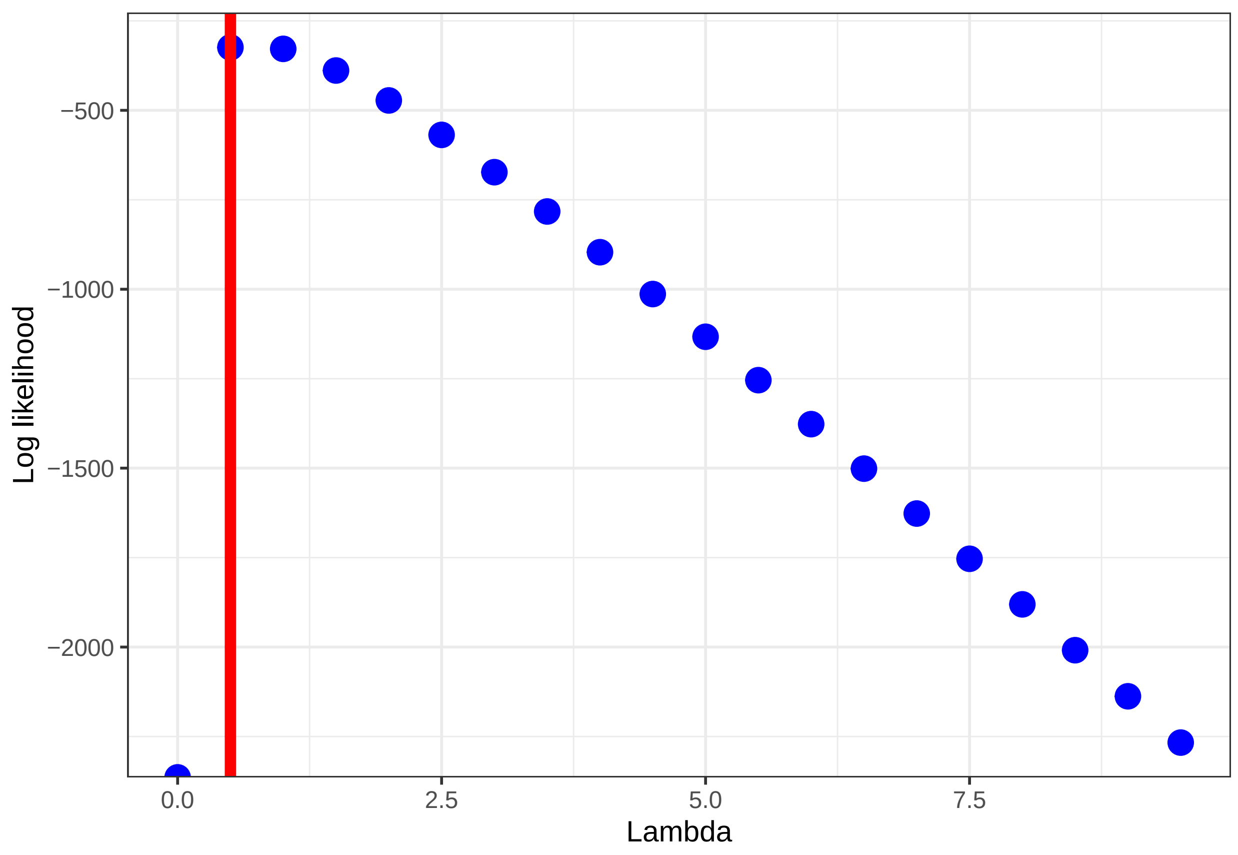


```r
llh_poisson <- function(lambda, y) {
  llh <- sum(dpois(y, lambda, log = T))
  return(llh)
}
```

```r
lambdas <- seq(0, 9.5, 0.5)
```

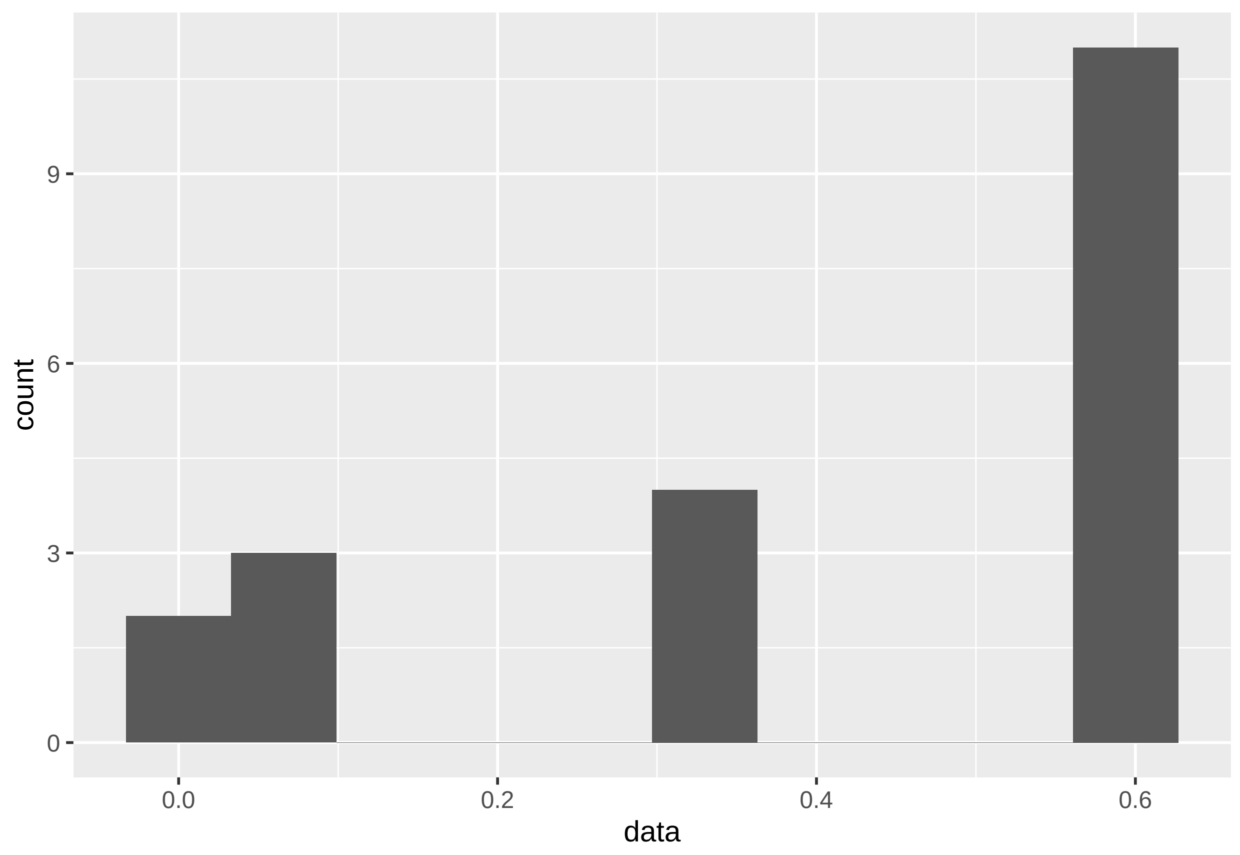
```r
ll <- sapply(lambdas, function(x) {llh_poisson(x, df_VonBortkiewicz$fatalities)})

df <- data.frame(ll = ll, lambda = lambdas)
```

```r
df %>%
  ggplot(aes(x = lambda, y = ll)) +
  geom_point(size = 4, color = "blue") +
  labs(x = "Lambda", y = "Log likelihood") +
  geom_vline(xintercept = lambdas[which.max(ll)], color = "red", size = 2) +
  theme_bw()
```



From above graph, we can see that $\lambda = 0.5$.

```
df_VonBortkiewicz_I <- df_VonBortkiewicz %>%
  filter(corps == 'I')

noFatilities <- data.frame(data = dpois(df_VonBortkiewicz_I$fatalities, 0.5))
noFatilities %>%
  ggplot(aes(x = data)) +
  geom_histogram(bins = 10)
```



1.4 Maximum likelihood estimation for the exponential distribution

1.4 (Q1)

$$\mathcal{L}(\lambda, x_1, \ldots, x_n) = \prod_{i=1}^{n} f(x_i, \lambda) = \prod_{i=1}^{n} \lambda e^{-\lambda x} = \lambda^n e^{-\lambda \sum_{i=1}^{n} x_i}$$

Maximum likelihood estimator:

$$\frac{d \ln (\mathcal{L}(\lambda, x_1, \ldots, x_n))}{d\lambda} \overset{!}{=} 0$$

$$\frac{d \ln (\mathcal{L}(\lambda, x_1, \ldots, x_n))}{d\lambda} = \frac{d \ln \left(\lambda^n e^{-\lambda \sum_{i=1}^{n} x_i}\right)}{d\lambda} = \frac{d \ln (n \ln(\lambda) - \lambda \sum_{i=1}^{n} x_i)}{d\lambda} = \frac{n}{\lambda} - \sum_{i=1}^{n} x_i$$

Which equals to,

$$\lambda = \frac{n}{\sum\limits_{i=1}^{n} x_i}$$

## 1.4 (Q2)

```
df_CustomerPurchases <- read.csv("CustomerPurchase.csv")

df_CustomerPurchases$lead <- lead(df_CustomerPurchases$Time)
```

```
df_CustomerPurchases$time_diffs <- df_CustomerPurchases$lead - df_CustomerPurchases$Time
df_CustomerPurchases <- select(df_CustomerPurchases, -lead)

head(df_CustomerPurchases)
```

```
##   Time Purchase time_diffs
## 1  564     3.25          7
## 2  571   504.85          7
## 3  578     7.60         22
## 4  600    43.45        145
## 5  745     9.30         61
## 6  806   352.80         27
```

1.4 (Q3)

```
exp_mle <- 1 / mean(df_CustomerPurchases$time_diffs, na.rm = T)
exp_mle
```

```
## [1] 0.02007792
```

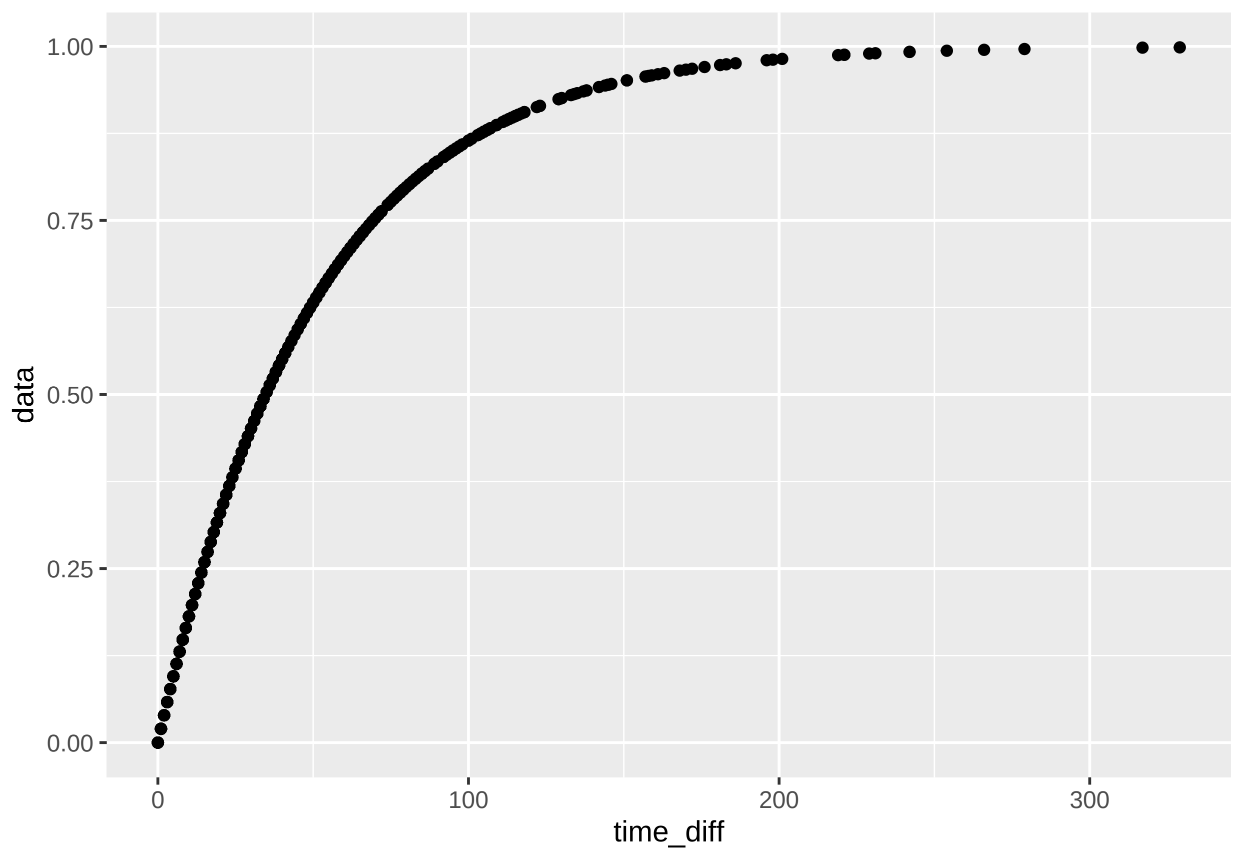
1.4 (Q4)

```
moreThanMin <- filter(df_CustomerPurchases, time_diffs > 60)
head(moreThanMin)
```

```
##   Time Purchase time_diffs
## 1  600    43.45        145
## 2  745     9.30         61
## 3  888    65.35         70
## 4  996   471.30         62
## 5 1058    76.30        221
## 6 1384   406.50         94
```

```
dist <- data.frame(data = pexp(df_CustomerPurchases$time_diffs, rate = 0.02)) %>%
  mutate(time_diff = df_CustomerPurchases$time_diffs)

ggplot(dist, aes(x = time_diff, y = data)) +
  geom_point()
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```



2. Confidence intervals

2.1 Student's t-confidence intervals

2.1 (Q1)

1. If the sample mean increases, width of confidence interval will not change.
2. If the sample standard deviation was higher, the width of confidence interval would have increased.
3. The width of confidence interval decreases when the sample size increases.

2.1 (Q2)

```r
HawksVector <- Hawks %>%
  filter(Species == "RT") %>%
  pull(Weight)

HawksVector <- HawksVector[!is.na(HawksVector)]
head(HawksVector)
```

```
## [1]  920  930  990 1090  960  855
```

```r
alpha <- 0.01
sample_size <- length(HawksVector)
sample_mean <- mean(HawksVector)
sample_sd <- sd(HawksVector)
t <- qt(1 - alpha / 2, df = sample_size - 1)
# confidence interval
confidence_interval_l <- sample_mean - t * sample_sd / sqrt(sample_size)
confidence_interval_u <- sample_mean + t * sample_sd / sqrt(sample_size)
confidence_interval <- c(confidence_interval_l, confidence_interval_u)
confidence_interval
```

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

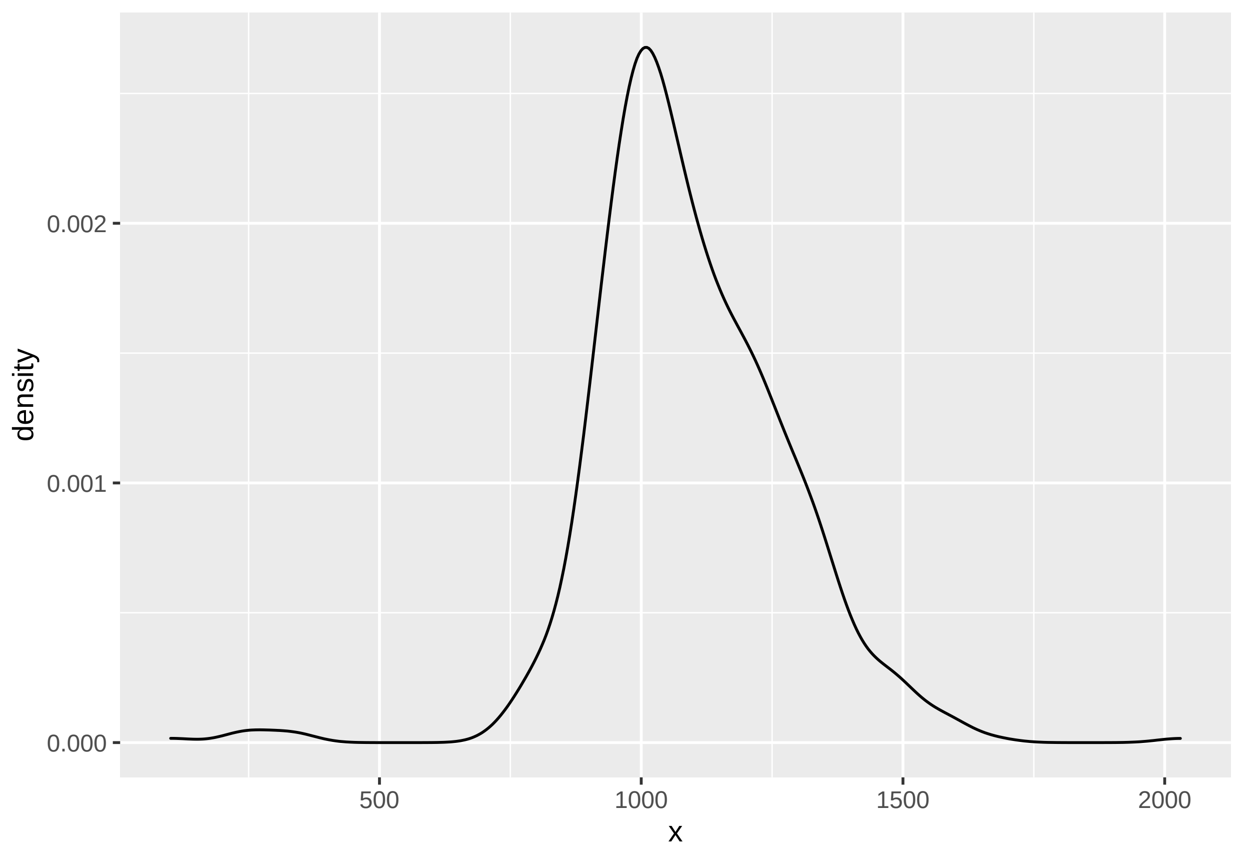
```r
DF <- length(HawksVector) - 1
alpha <- 0.01
t = qt(1 - alpha / 2, DF)
t
```

```
## [1] 2.584467
```

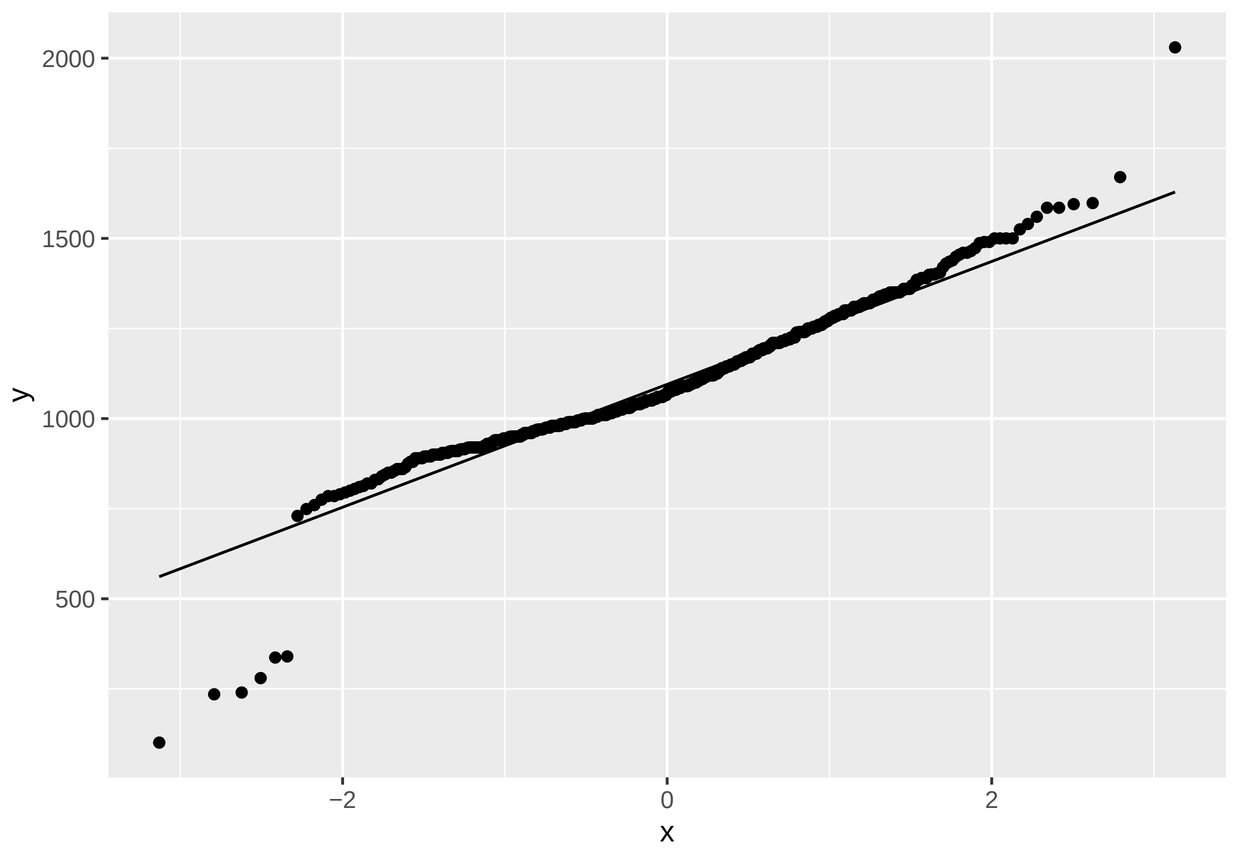
```r
t.test(HawksVector, conf.level = 0.99)
```

```
##
##  One Sample t-test
##
## data:  HawksVector
## t = 138.34, df = 571, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 99 percent confidence interval:
##  1073.984 1114.877
## sample estimates:
## mean of x
##   1094.43
```

```r
tempDf <- data.frame(x = HawksVector) %>%
  mutate(pdf = map_dbl(.x = x, ~dnorm(.x, mean(HawksVector), sd(HawksVector))))

ggplot(data = tempDf, aes(x = x)) +
  geom_density()
```



```
ggplot(tempDf, aes(sample = x)) +
  stat_qq() +
  stat_qq_line()
```



2.2 Investigating coverage for Student's t intervals

2.2 (Q1)

```r
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))
}
```

```r
num_trials <- 100000
sample_size <- 30
mu_0 <- 1
sigma_0 <- 3
alpha <- seq(0.01, 0.09, 0.01)
set.seed(0) # set random seed for reproducibility
```

```r
probEstimate <- function(num_trials, sample_size, mu_0, alpha) {
  single_alpha_coverage_simulation_df <- data.frame(trial = seq(num_trials)) %>%
    mutate(sample = map(.x =  trial,.f = ~rnorm(n = sample_size, mean = mu_0, sd = sigma_0))) %>%
    mutate(ci_interval = map(.x = sample, .f = ~student_t_confidence_interval(.x, 1 - alpha)))%>%
    mutate(cover = map_lgl(.x = ci_interval, .f = ~((min(.x) <= mu_0) & (max(.x) >= mu_0))))%>%
    mutate(ci_length = map_dbl(.x = ci_interval, .f = ~(max(.x) - min(.x))))


  val <- single_alpha_coverage_simulation_df %>%
    pull(cover) %>%
    mean()
  return(val)
}

lu_vec <- c()
gamma_vec <- c()

for(i in alpha) {
  g <- probEstimate(num_trials, sample_size, mu_0, i)
  g
}
```

3. One sample hypothesis testing

3.1 One sample t-test on penguins data

3.1 (Q1)

```r
library(palmerpenguins)
```

```
## Warning: package 'palmerpenguins' was built under R version 4.2.2
```

```r
data(package = 'palmerpenguins')
head(penguins)
```

```
## # A tibble: 6 x 8
##   species island    bill_length_mm bill_depth_mm flipper_l~1 body_~2 sex    year
##   <fct>   <fct>              <dbl>         <dbl>       <int>   <int> <fct> <int>
## 1 Adelie  Torgersen           39.1          18.7         181    3750 male   2007
## 2 Adelie  Torgersen           39.5          17.4         186    3800 fema~  2007
## 3 Adelie  Torgersen           40.3          18           195    3250 fema~  2007
## 4 Adelie  Torgersen           NA            NA            NA      NA <NA>   2007
## 5 Adelie  Torgersen           36.7          19.3         193    3450 fema~  2007
## 6 Adelie  Torgersen           39.3          20.6         190    3650 male   2007
## # ... with abbreviated variable names 1: flipper_length_mm, 2: body_mass_g
```

```r
bill_adelie <- penguins %>%
  filter(species == "Adelie") %>%
  pull(bill_length_mm)

t.test(bill_adelie, mu = 40, conf.level = 0.01)
```

```
##
##  One Sample t-test
##
## data:  bill_adelie
## t = -5.5762, df = 150, p-value = 1.114e-07
## alternative hypothesis: true mean is not equal to 40
## 1 percent confidence interval:
##  38.78867 38.79411
## sample estimates:
## mean of x
##  38.79139
```

3.2 Implementing a one-sample t-test

3.2 (Q1)

```
p_val_calc <- function(x, mu) {
  mn <- mean(x, na.rm = T)
  t = (mn - mu) / (sd(x, na.rm = T) / sqrt(length(x)))
  p = 2 * (1 - pt(abs(t), df = length(x) - 1))
  return(p)
}

p_val_calc(bill_adelie, 40)
```

```
## [1] 1.011578e-07
```