Advanced Bioinformatics 2025 assessment

m2407447

Task 1

In this task, I will use the `sum()` function and the `:` operator in R to calculate the sum of all integers from 5 to 55.
The `:` operator generates a sequence of numbers (in this case, 5 to 55), and `sum()` adds them together. In this case the sum is 1530

```r
sum(5:55)
```

```
## [1] 1530
```

Task 2

In this task, I define a function called `sumfun()` that takes a single input parameter `n`, and calculates the sum of all integers from 5 to `n` using the `sum()` function and the `:` operator. I then use the function to compute the sum for `n = 10`, `n = 20`, and `n = 100`.

```r
# Define the function
sumfun <- function(n) {
  sum(5:n)
}
```

```r
# Use the function with different values of n
sumfun(10)
```

```
## [1] 45
```

```r
sumfun(20)
```

```
## [1] 200
```

```r
sumfun(100)
```

```
## [1] 5040
```

Task 3

In this task, I generate the first 12 numbers in the Fibonacci sequence using a `for` loop in R. The Fibonacci sequence is defined such that each number is the sum of the two preceding ones, starting from 1 and 1.

```r
# Create a numeric vector to store Fibonacci numbers
fib <- numeric(12)

# Set the first two numbers
fib[1] <- 1
fib[2] <- 1

# Use a for loop to calculate the next numbers
for (i in 3:12) {
  fib[i] <- fib[i - 1] + fib[i - 2]
}

# Print the result
fib
```

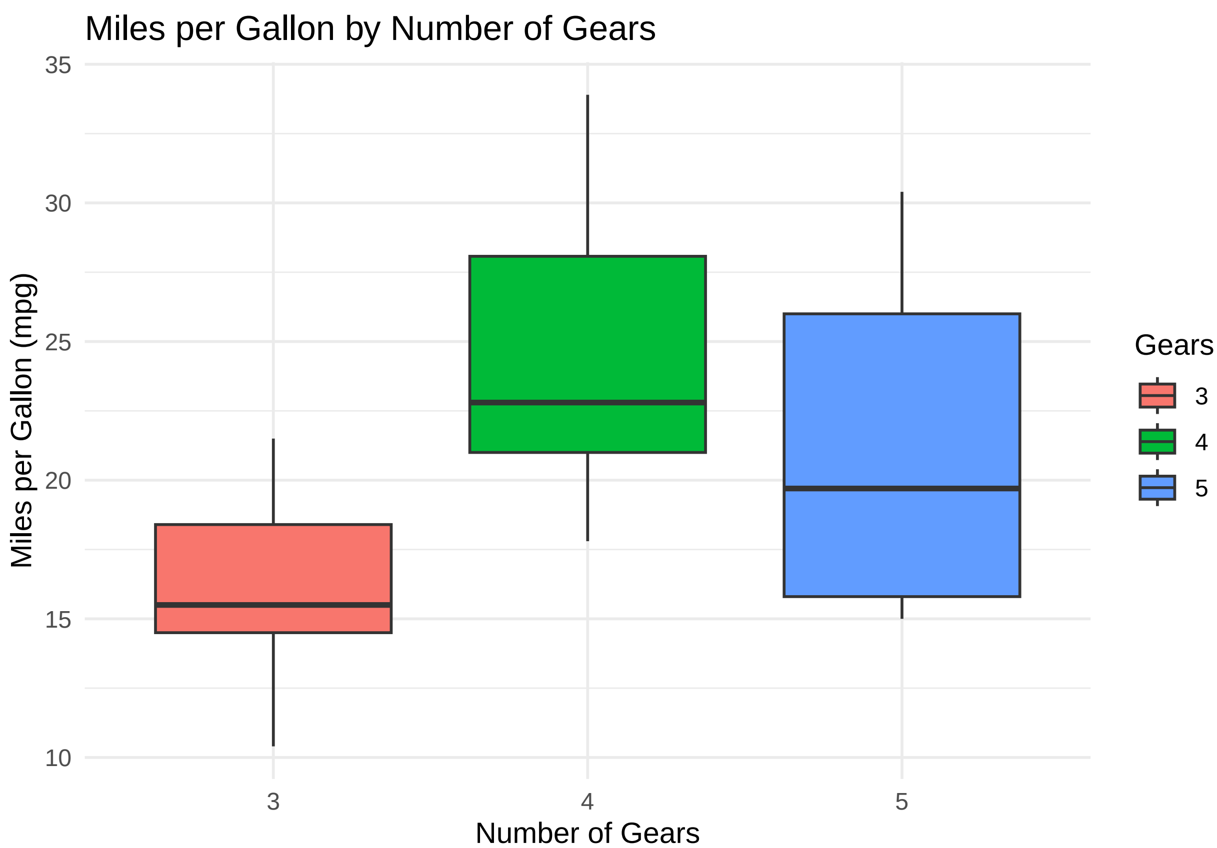
```
##  [1]   1   1   2   3   5   8  13  21  34  55  89 144
```

Task 4

In this task, I use the `ggplot2` package to create a boxplot of miles per gallon (`mpg`) as a function of the number of gears (`gear`) in the `mtcars` dataset. The `fill` aesthetic is used to colour the boxes based on the number of gears.

```r
# Load ggplot2
library(ggplot2)

# Create the boxplot
ggplot(mtcars, aes(x = factor(gear), y = mpg, fill = factor(gear))) +
  geom_boxplot() +
  labs(
    title = "Miles per Gallon by Number of Gears",
    x = "Number of Gears",
    y = "Miles per Gallon (mpg)",
    fill = "Gears"
  ) +
  theme_minimal()
```



Task 5

In this task, I fit a linear model to explore the relationship between a car's speed (`speed`) and its stopping distance (`dist`) using the `cars` dataset. The `lm()` function is used to compute the regression line.

Step 1: Fit the linear model

```
# Fit the linear model
model <- lm(dist ~ speed, data = cars)

# Show the model summary
summary(model)
```

```
##
## Call:
## lm(formula = dist ~ speed, data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.069  -9.525  -2.272   9.215  43.201
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -17.5791     6.7584  -2.601   0.0123 *
## speed         3.9324     0.4155   9.464 1.49e-12 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.38 on 48 degrees of freedom
## Multiple R-squared:  0.6511, Adjusted R-squared:  0.6438
## F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

Step 2: Interpret the output

- **Fitted slope (speed coefficient)**: This tells us how much the stopping distance increases for every 1 mph increase in speed.
- **Fitted intercept**: This is the estimated stopping distance when speed = 0 mph.
- **Standard errors**: These represent the variability (uncertainty) in the slope and intercept estimates.

Step 3: Units

- `speed` is measured in **miles per hour (mph)**

- `dist` is measured in **feet**

Task 6

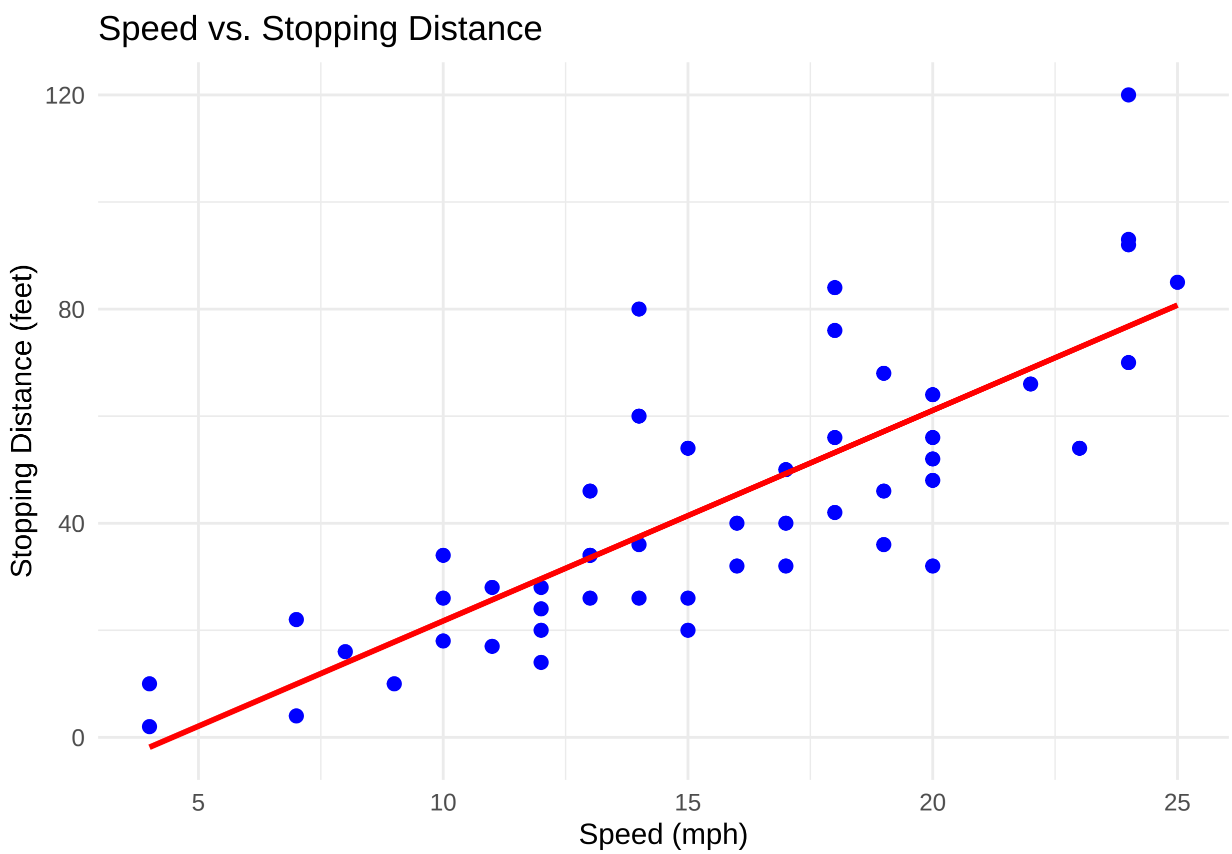
In this task, I use `ggplot2` to plot the data points from the `cars` dataset along with the fitted linear regression line from Task 5.

Step 1: Create the Plot

```r
# Load ggplot2
library(ggplot2)

# Create the scatter plot with the regression line
ggplot(cars, aes(x = speed, y = dist)) +
  geom_point(color = 'blue', size = 2) +   # Plot the data points
  geom_smooth(method = 'lm', color = 'red', se = FALSE) +  # Add the linear fit line
  labs(
    title = "Speed vs. Stopping Distance",
    x = "Speed (mph)",
    y = "Stopping Distance (feet)"
  ) +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Step 2: Interpretation

- The **blue points** represent the data (speed vs. stopping distance).
- The **red line** represents the fitted linear regression model, showing the relationship between speed and stopping distance.

Task 7

In this task, I estimate the average **reaction time** for a driver to start breaking using the `cars` dataset. The assumption is that once breaking starts, the breaking distance is proportional to the square of the speed.

Step 1: Fit the Linear Regression Model

First, I will fit a linear regression model to the `dist` (breaking distance) as a function of `speed^2`. The relationship is assumed to be:

$$\text{dist} = \text{reaction time} + k \times \text{speed}^2$$

```r
# Fit the linear model
model_reaction_time <- lm(dist ~ I(speed^2), data = cars)

# Show the model summary
summary(model_reaction_time)
```

```
##
## Call:
## lm(formula = dist ~ I(speed^2), data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.448  -9.211  -3.594   5.076  45.862
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  8.86005    4.08633   2.168   0.0351 *
## I(speed^2)   0.12897    0.01319   9.781  5.2e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.05 on 48 degrees of freedom
## Multiple R-squared:  0.6659, Adjusted R-squared:  0.6589
## F-statistic: 95.67 on 1 and 48 DF,  p-value: 5.2e-13
```

Step 2: Calculate Reaction Time

From the regression results, I can extract the **reaction time** by subtracting the fitted intercept (which represents the reaction time) from the regression model.

```r
# Extract the fitted intercept (reaction time)
reaction_time <- coef(model_reaction_time)[1]
reaction_time
```

```
## (Intercept)
##    8.860049
```

Step 3: Interpretation

- The **reaction time** represents the constant time it takes the driver to start breaking.
- The **slope** (coefficient of `speed^2`) represents the constant of proportionality `k` for the breaking distance.

Step 4: Plot the Data and the Fitted Relationship

I will now use `ggplot2` to visualize the data and the fitted regression line.

```r
# Load ggplot2 for plotting
library(ggplot2)

# Create the scatter plot and the fitted regression line
ggplot(cars, aes(x = speed, y = dist)) +
  geom_point(color = 'blue', size = 2) +   # Data points
  geom_smooth(method = 'lm', formula = y ~ I(x^2), color = 'red', se = FALSE) +  # Regression line
  labs(
    title = "Speed vs. Stopping Distance with Reaction Time",
    x = "Speed (mph)",
```

```
    y = "Stopping Distance (feet)"
  ) +
  theme_minimal()
```