UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Faculty of Engineering, Built Environment and Information Technology

Fakulteit Ingenieurswese, Bou-omgewing en Inligtingtegnologie / Lefapha la Boetšenere, Tikologo ya Kago le Theknolotši ya Tshedimošo

DEPARTMENT OF CIVIL ENGINEERING

SHC 798

APPLIED STATISTICAL METHODS AND OPTIMISATION

RICHARD LUBEGA

Full names

25585089

Student number

1

Assignment

DECLARATION

1. I understand what plagiarism is and am aware of the University's policy in this regard.
2. I declare that this submission is my original work. Wherever other people's work has been used (either from a printed source, the internet or any other source) thishas been properly acknowledged and referenced in accordance with departmental requirements.
3. I declare that I did not use ChatGPT or similar AI-based tools to prepare this report.
4. I have not used another student's current or past written work to hand in as myown.
5. I have not allowed and will not allow anyone to copy my work to pass it off as his or her work.

Signature: __________________

Date: 14-06-2025

SHC 798 Assignment 1, 2025

Richard Lubega

2025-07-14

SHC 798 Assignment 1, 2025

Part 1: Data Analysis with R

```
# Getting Started with the Dataset:
pacman::p_load(ggplot2)
pacman::p_load(tidymodels)

head(mpg) # View first few rows of the dataset
```

```
## # A tibble: 6 x 11
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(l5)   f        18    29 p     compa~
## 2 audi         a4      1.8  1999     4 manual(m5) f        21    29 p     compa~
## 3 audi         a4      2    2008     4 manual(m6) f        20    31 p     compa~
## 4 audi         a4      2    2008     4 auto(av)   f        21    30 p     compa~
## 5 audi         a4      2.8  1999     6 auto(l5)   f        16    26 p     compa~
## 6 audi         a4      2.8  1999     6 manual(m5) f        18    26 p     compa~
```

```
summary(mpg) # Get an overview of the dataset
```

```
##  manufacturer          model               displ            year     
##  Length:234         Length:234         Min.   :1.600   Min.   :1999  
##  Class :character   Class :character   1st Qu.:2.400   1st Qu.:1999  
##  Mode  :character   Mode  :character   Median :3.300   Median :2004  
##                                        Mean   :3.472   Mean   :2004  
##                                        3rd Qu.:4.600   3rd Qu.:2008  
##                                        Max.   :7.000   Max.   :2008  
##       cyl           trans               drv                 cty       
##  Min.   :4.000   Length:234         Length:234         Min.   : 9.00  
##  1st Qu.:4.000   Class :character   Class :character   1st Qu.:14.00  
##  Median :6.000   Mode  :character   Mode  :character   Median :17.00  
##  Mean   :5.889                                         Mean   :16.86  
##  3rd Qu.:8.000                                         3rd Qu.:19.00  
##  Max.   :8.000                                         Max.   :35.00  
##       hwy             fl               class          
##  Min.   :12.00   Length:234         Length:234        
```

```
##  1st Qu.:18.00   Class :character   Class :character
##  Median :24.00   Mode  :character   Mode  :character
##  Mean   :23.44
##  3rd Qu.:27.00
##  Max.   :44.00
```

```r
#(a)
# average city and highway fuel economy across all vehicle classes
cat("=== Average city and highway fuel economy, afe, across all vehicle classes ===\n")
```

```
## === Average city and highway fuel economy, afe, across all vehicle classes ===
```

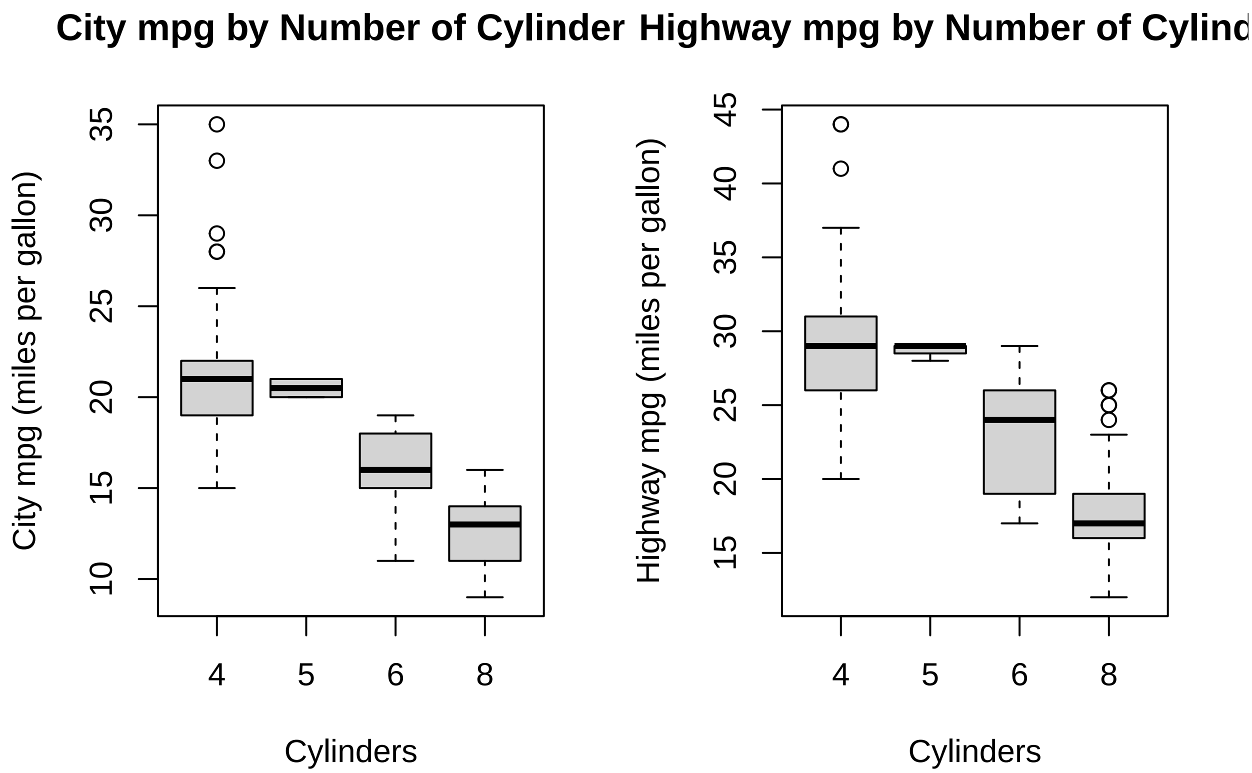
```r
afe <- aggregate(cbind(cty, hwy) ~ class, data = mpg, FUN = mean)
afe
```

```
##        class      cty      hwy
## 1    2seater 15.40000 24.80000
## 2    compact 20.12766 28.29787
## 3    midsize 18.75610 27.29268
## 4    minivan 15.81818 22.36364
## 5     pickup 13.00000 16.87879
## 6 subcompact 20.37143 28.14286
## 7        suv 13.50000 18.12903
```

```r
#(b)
# Compare the fuel efficiency (cty and hwy)
cat("=== Comparing fuel efficiency for cty and hwy economies ===\n")
```

```
## === Comparing fuel efficiency for cty and hwy economies ===
```

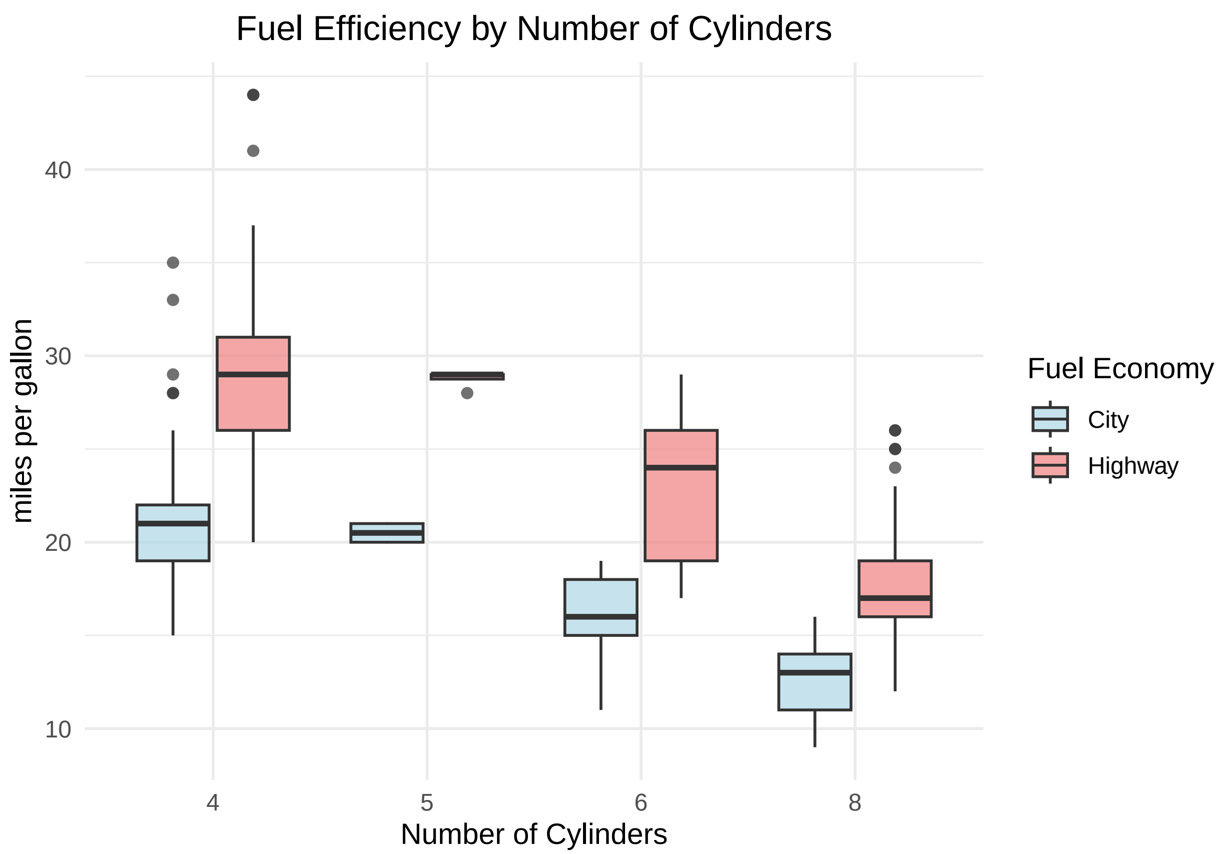
```r
par(mfrow = c(1, 2)) # Set up a 1x2 plot layout for side-by-side boxplots

# Boxplot for city mpg by cylinders
boxplot(cty ~ cyl, data = mpg,
        main = "City mpg by Number of Cylinders",
        xlab = "Cylinders",
        ylab = "City mpg (miles per gallon)")

# Boxplot for highway mpg by cylinders
boxplot(hwy ~ cyl, data = mpg,
        main = "Highway mpg by Number of Cylinders",
        xlab = "Cylinders",
        ylab = "Highway mpg (miles per gallon)")
```




```
par(mfrow = c(1, 1)) # Reset plot layout to default

# Combine plots by faceting
cat("=== Combining the box plots for comparison ===\n")
```

```
## === Combining the box plots for comparison ===
```

```
mpg_comb <- mpg %>%
  select(cyl, cty, hwy) %>%
  pivot_longer(cols = c(cty, hwy), names_to = "fuel_econ", values_to = "mpg")

ggplot(mpg_comb, aes(x = factor(cyl), y = mpg, fill = fuel_econ)) +
  geom_boxplot(alpha = 0.7) +
  labs(title = "Fuel Efficiency by Number of Cylinders",
       x = "Number of Cylinders",
       y = "miles per gallon",
       fill = "Fuel Economy") +
  scale_fill_manual(values = c("cty" = "lightblue", "hwy" = "lightcoral"),
                    labels = c("City", "Highway")) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```




```r
#---------  Median Values by cylinder count -------------
cat("=== Median Values by cylinder count ===\n")
```

```
## === Median Values by cylinder count ===
```

```r
mpg %>%
  group_by(cyl) %>%
  summarise(
    median_cty = median(cty),
    median_hwy = median(hwy),
    .groups = 'drop'
  )
```

```
## # A tibble: 4 x 3
##     cyl median_cty median_hwy
##   <int>      <dbl>      <dbl>
## 1     4       21           29
## 2     5       20.5         29
## 3     6       16           24
## 4     8       13           17
```

```r
cat("=== Trend Analysis ===\n")
```

```
## === Trend Analysis ===
```

Commenting on the Trend This analysis clearly demonstrates that engine size (cylinder count) is a major predictor of fuel efficiency, with smaller engines being visibly more fuel-efficient than larger ones. Some *outliers* exist (may be due to high-efficiency hybrids or low-efficiency compact cars).

- **Inverse relationship:** Based on the boxplots (where more cylinders = lower mpg), there's a clear *negative* correlation between the number of cylinders and fuel efficiency (mpg). As cylinder count increases, both city and highway mpg decrease.
- **Highway vs City efficiency:** Highway mpg is consistently higher than city mpg across all cylinder counts (as seen from the combined plot), which may be explained by the more efficient cruising speeds on highways. Generally, the **fuel efficiency difference** between city and highway driving becomes more pronounced in vehicles with fewer cylinders.
- **4-cylinder cars** are the most fuel-efficient, with median values of 21 mpg (for city) and 29 mpg (for highway). The rest in each category have lower values. **8-cylinder cars** are the least fuel-efficient, with median values of 13 mpg (for city) and 17 mpg (for highway).
- **5-cylinder cars** are the least common (narrower range) in both categories. This may be due to fewer models of these cars. **6-cylinder cars** have the most broad range compared to the others
- There is also **variability within cylinder** groups, and is most pronounced in **6-cylinder cars**, whch suggests that factors beyond cylinder count (including vehicle weight, engine technology, etc.) also influence fuel efficiency.

```
# (c)
# Correlation: Engine Displacement vs Highway Fuel Economy
cat("Correlation: engine displacement (displ) and highway fuel economy (hwy) \n")
```

```
## Correlation: engine displacement (displ) and highway fuel economy (hwy)
```

```
# Calculate correlation coefficient
correlation_pearson <- cor(mpg$displ, mpg$hwy)
correlation_spearman <- cor(mpg$displ, mpg$hwy, method = "spearman")

cat("Pearson correlation coefficient:", round(correlation_pearson, 4), "\n")
```

```
## Pearson correlation coefficient: -0.766
```

```
cat("Spearman correlation coefficient:", round(correlation_spearman, 4), "\n")
```

```
## Spearman correlation coefficient: -0.8267
```

```
# Interpretation of correlation strength
interpret_correlation <- function(r) {
  abs_r <- abs(r)
  if (abs_r >= 0.7) return("Strong")
  else if (abs_r >= 0.3) return("Moderate")
  else return("Weak")
}

cat("Correlation strength:", interpret_correlation(correlation_pearson), "\n")
```

```
## Correlation strength: Strong
```

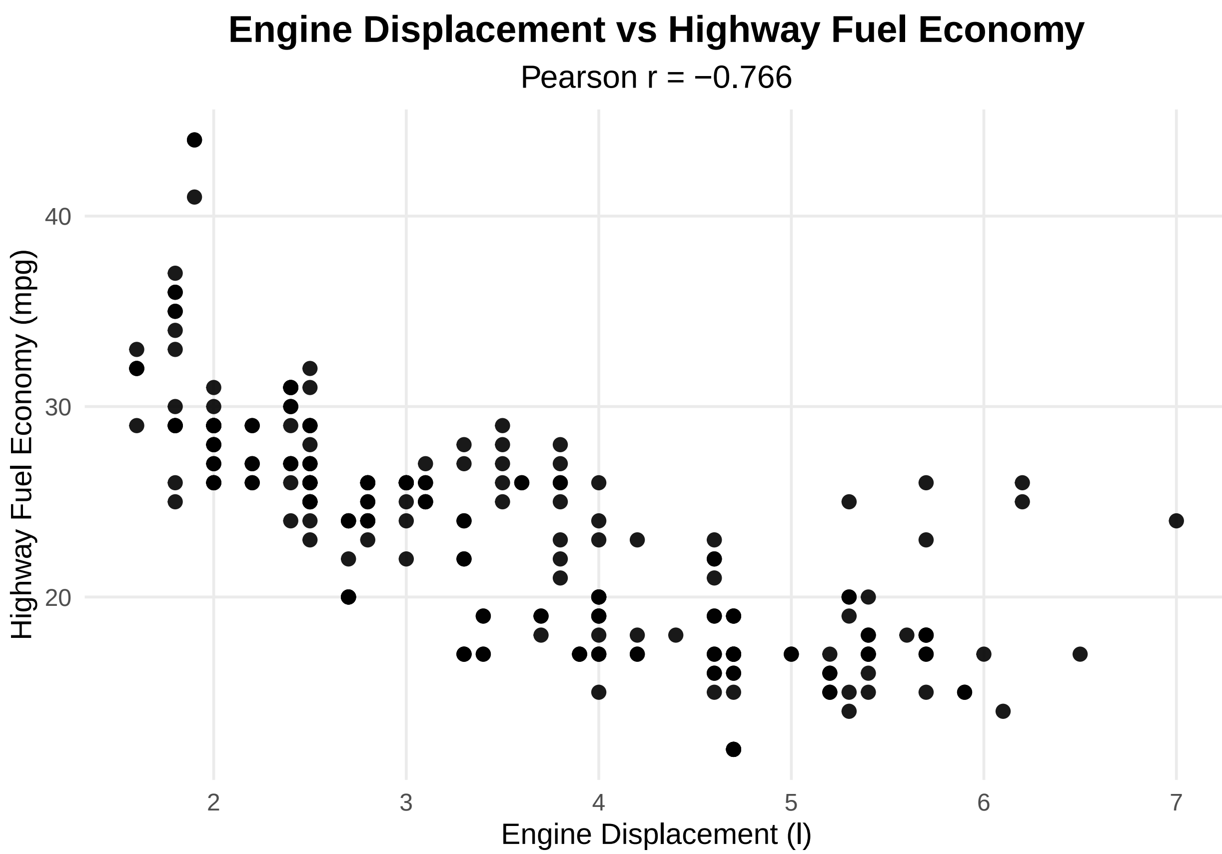
```r
cat("Direction:", ifelse(correlation_pearson > 0, "Positive", "Negative"), "\n")
```

```
## Direction: Negative
```

```r
# Create basic scatter plot
cat("=== Creating a Basic Scatter Plot ===\n")
```

```
## === Creating a Basic Scatter Plot ===
```

```r
# Basic scatter plot
plot_dh <- ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point(alpha = 0.9, size = 2, color = "black") +
    labs(
    title = "Engine Displacement vs Highway Fuel Economy",
    subtitle = paste("Pearson r =", round(correlation_pearson, 3)),
    x = "Engine Displacement (l)",
    y = "Highway Fuel Economy (mpg)",
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 14, face = "bold", hjust = 0.5),
    plot.subtitle = element_text(size = 12, hjust = 0.5),
    axis.title = element_text(size = 11),
    panel.grid.minor = element_blank()
  )

print(plot_dh)
```




```r
cat("Test the significance of the correlation \n")
```

```
## Test the significance of the correlation
```

```r
cor_test <- cor.test(mpg$displ, mpg$hwy, method = "pearson")
cat("Pearson correlation test:\n")
```

```
## Pearson correlation test:
```

```r
print(cor_test)
```

```
##
##  Pearson's product-moment correlation
##
## data:  mpg$displ and mpg$hwy
## t = -18.151, df = 232, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.8142727 -0.7072539
## sample estimates:
##      cor
## -0.76602
```

```r
cat("Significance level: ", ifelse(cor_test$p.value < 0.001, "p < 0.001 (highly significant)",
                                   ifelse(cor_test$p.value < 0.01, "p < 0.01 (significant)",
                                          ifelse(cor_test$p.value < 0.05, "p < 0.05 (significant)", "no
```

```
## Significance level:  p < 0.001 (highly significant)
```

Therefore, based on the analysis, a **strong negative**, **highly** statistically **significant** correlation exists between engine displacement (displ) and highway fuel economy (hwy).

The scatter plot reinforces this because as displacement increases, highway mpg decreases.

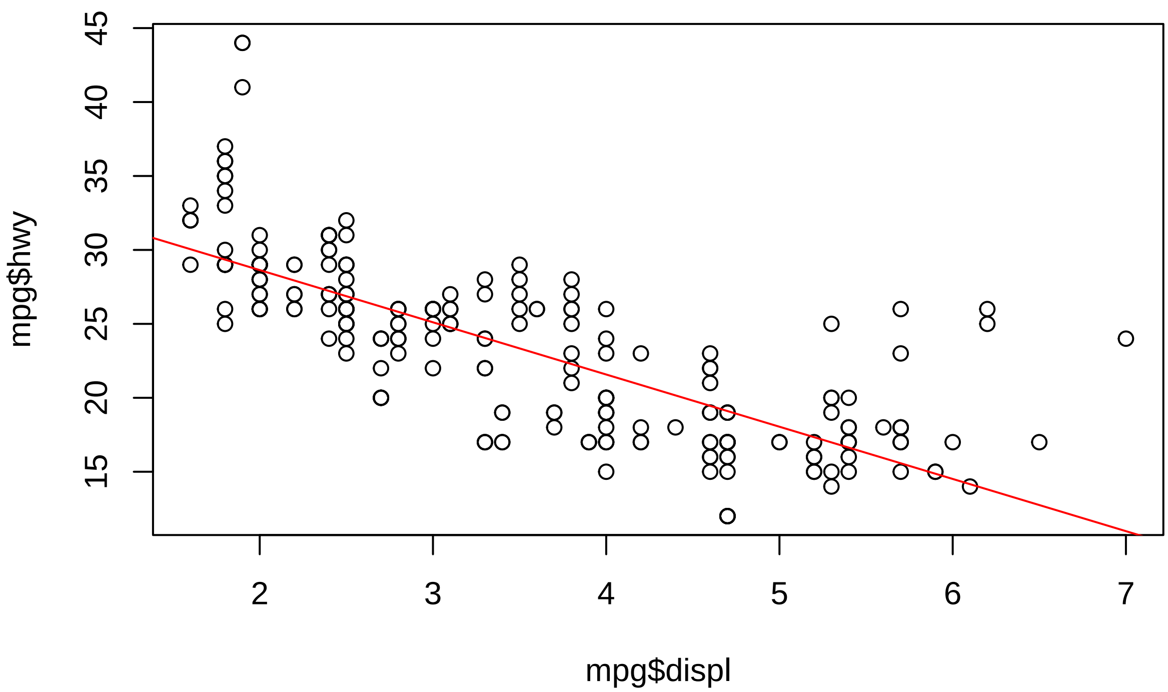
```r
# Linear regression
cat("Linear Regression Model \n")
```

```
## Linear Regression Model
```

```r
lm_model <- lm(hwy ~ displ, data = mpg)
summary(lm_model)
```

```
##
## Call:
## lm(formula = hwy ~ displ, data = mpg)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.1039 -2.1646 -0.2242  2.0589 15.0105
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  35.6977     0.7204   49.55   <2e-16 ***
## displ        -3.5306     0.1945  -18.15   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.836 on 232 degrees of freedom
## Multiple R-squared:  0.5868, Adjusted R-squared:  0.585
## F-statistic: 329.5 on 1 and 232 DF,  p-value: < 2.2e-16
```

```r
plot(mpg$displ, mpg$hwy)+
  abline(lm_model, col = "red")
```

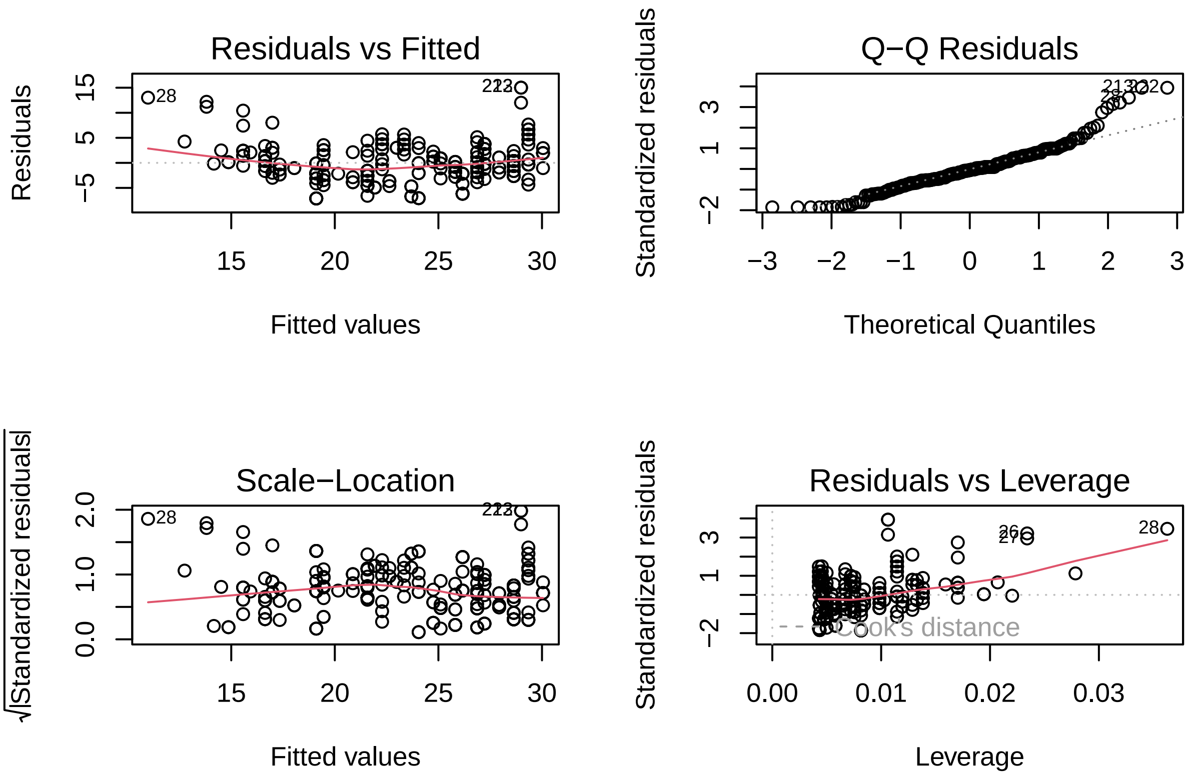



```
## integer(0)
```

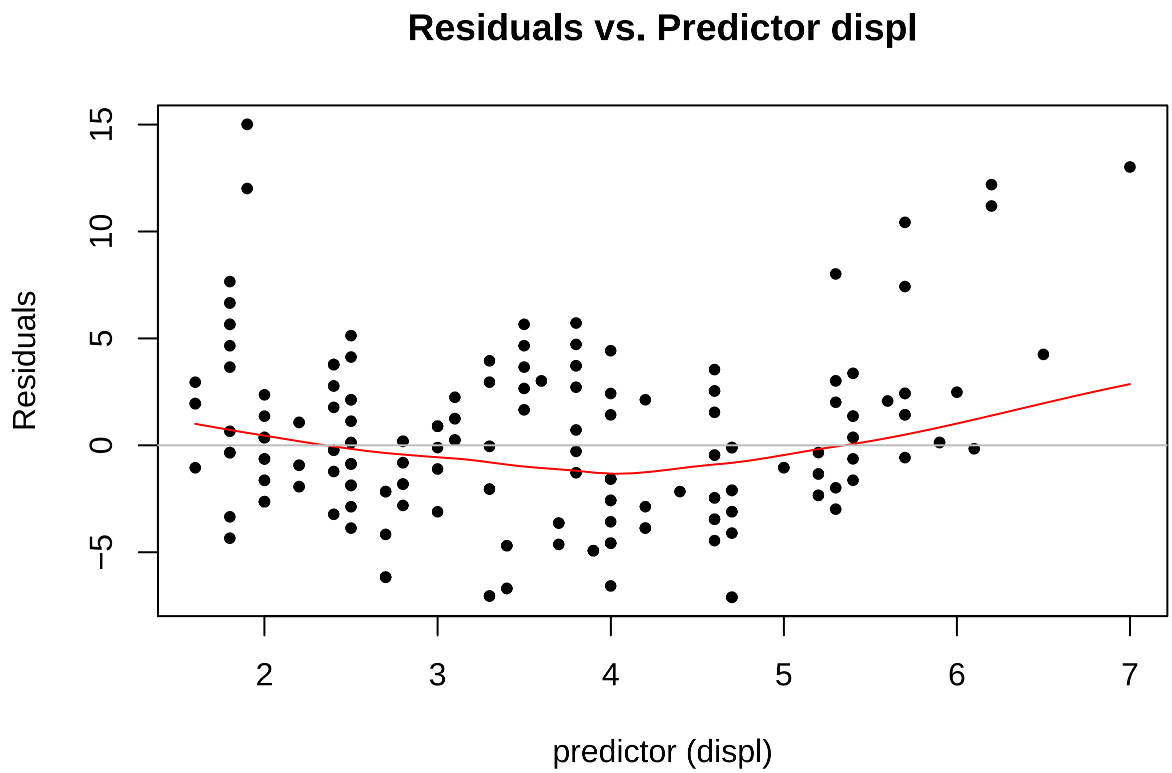
```r
cat("Model Diagnostics \n")
```

```
## Model Diagnostics
```

```r
# Diagnostics plots
par(mfrow = c(2,2))
plot(lm_model)
```

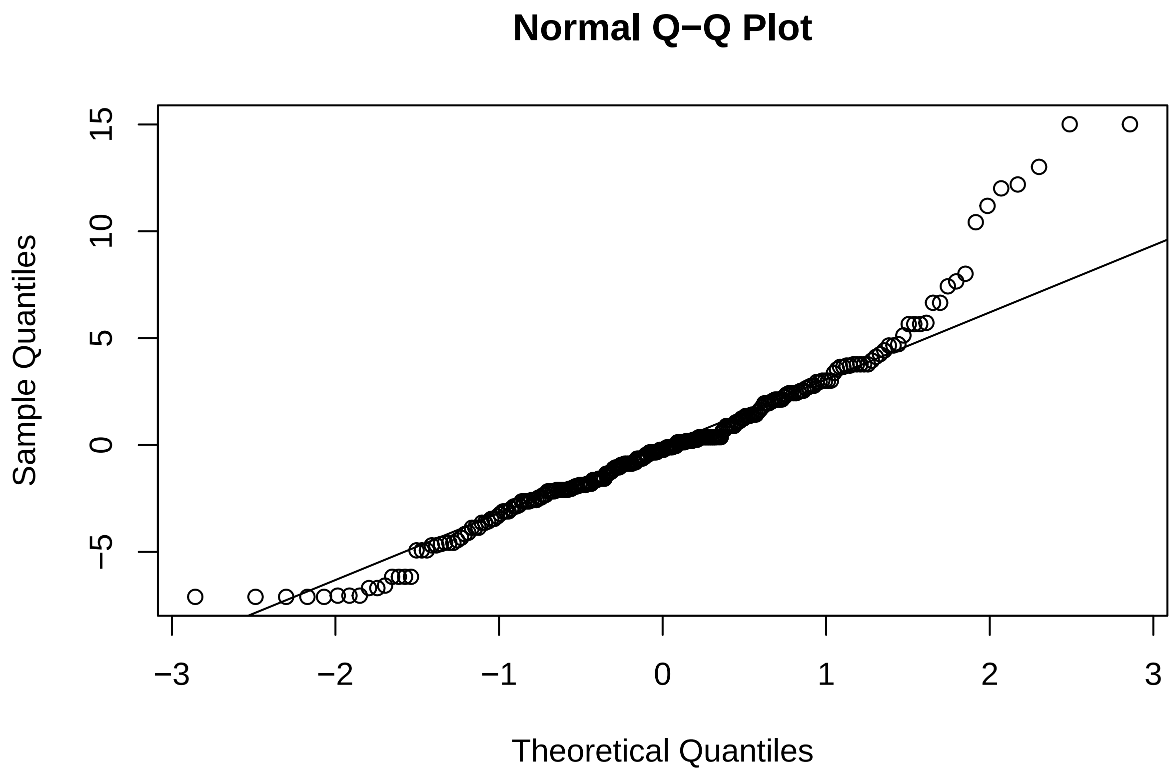



```
par(mfrow = c(1,1))
# Tukey-Anscombe Plot
# plot(lm_model$fitted.values, lm_model$residuals, xlab="Fitted", ylab="Residuals", pch=20) +
#   title("Residuals vs. Fitted Values") +
#   lines(loess.smooth(lm_model$fitted.values, lm_model$residuals),col="red") +
#   abline(h=0, col="grey")

# Residuals vs. Predictor Plot
plot(mpg$displ, lm_model$residuals, xlab="predictor (displ)", ylab="Residuals", pch=20) +
  title("Residuals vs. Predictor displ") +
  lines(loess.smooth(mpg$displ, lm_model$residuals),col="red") +
  abline(h=0, col="grey")
```




```
## integer(0)
```

```
# Quantile-Quantile Plot
qqnorm(lm_model$residuals) #Quantile-Quantile Plot
qqline(lm_model$residuals) # adds the diagonal line
```



From the **model diagnostics (Tuskey-Anscombe plot)**, the red LOESS line is slightly curved which indicates non-linearity. Nonetheless, the expectation of the residuals can be considered zero. The variance of the errors increases with fitted values and homoskedasticity is violated.

The **Q-Q plot** indicates that the bulk of the residuals (in the central region) are approximately Gaussian distributed. The data exhibits heavy tails (skewness) and has outliers at the extremes. The noticeable presence of extreme positive residuals suggests a right-skewed distribution (departure from normality), the assumption of Gaussian errors is violated by the model.

To improve the model, variable transformation is required (to stabilize the spread and ensure error normality).

Comment on Model Outputs: The regression model, **lm_model** predicts highway fuel economy (hwy, in mpg) as a function of engine displacement (displ, in litres).

- **Regression Coefficients**:
  - **Intercept** (35.6977) implies that when engine displacement is theoretically 0 litres, the predicted highway fuel economy is approximately 35.7 mpg. It's p-value (< 2e-16) is very small and indicates that it is statistically significant.
  - **Slope** (-3.5306): For each 1-litre increase in engine displacement, highway fuel economy decreases by approximately 3.53 mpg, on average. The t-value (-15.07) and p-value (< 2e-16) indicate this coefficient is highly significant, confirming a strong negative relationship.

- **Statistical Significance**:

The p-value for displ is very small (< 2.2e-16), meaning the relationship is statistically significant. Engine size is a strong/meaningful predictor of fuel efficiency.

- **Model Fit**:
  - The **multiple R-squared** (0.5868) and **adjusted R-squared** (0.585) indicate that approximately 58.68% of the variability in highway fuel economy is explained by engine displacement. This suggests a moderately strong negative relationship, but other factors (e.g., vehicle weight, transmission type) may also play a role.

Implication of Model Outputs on the Relationship

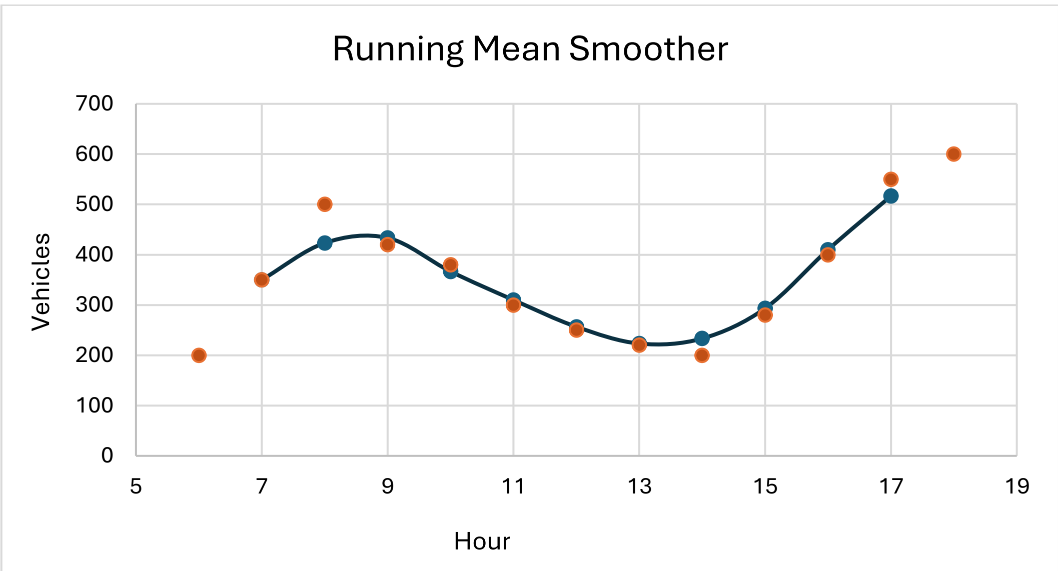
- The **negative coefficient** for *displ* (-3.5306) supports the belief that cars with smaller engines have better fuel efficiency. As engine size increases, highway fuel economy decreases significantly, with a 1-liter increase in displacement leading to a 3.53 mpg reduction in fuel efficiency, on average. The highly significant **p-values** for both the displ coefficient and the overall model ($< 2e\text{-}16$) confirm that the negative relationship between engine size and fuel efficiency is *not due to random chance*. This strengthens the conclusion that engine size is a reliable predictor of fuel efficiency.
- The **R-squared value** (0.5868) indicates that engine size alone doesn't explain all the variability in fuel efficiency, so the remaining 41.32% of variability implies other factors (like, vehicle weight, car transmission type, or car drive type) also influence fuel efficiency.

PART 2: DATA SMOOTHING

(a-1) Compute a running mean smoother by hand

Running Mean Smoother			
Window width		3 hours	
Time	Hour	Ave.Veh	Ave.Veh
06:00	6	-	N/A
07:00	7	(200+350+500)/3	350.000
08:00	8	(350+500+420)/3	423.333
09:00	9	(500+420+380)/3	433.333
10:00	10	(420+380+300)/3	366.667
11:00	11	(380+300+250)/3	310.000
12:00	12	(300+250+220)/3	256.667
13:00	13	(250+220+200)/3	223.333
14:00	14	(220+200+280)/3	233.333
15:00	15	(200+280+400)/3	293.333
16:00	16	(280+400+550)/3	410.000
17:00	17	(400+550+600)/3	516.667
18:00	18	-	N/A

(a-2) Draw your solution on a scatter plot of the data



(a-3) Validate your solution in R

------ (Code and Output shown in the following section) -----

(b) Use R to compute a running mean smoother using ksmooth()

------ (Code and Output shown in the following section) -----

(c-1) Create a Gaussian kernel smoother in Excel

σ	2
λ = 2*σ^2	8
Weights	wi

Gaussian Kernel Smoother in Excel			
Standard deviation is two hours			
Time	Hour	Vehicles	Gaussian Kernel Values
06:00	6	200	338.0665
07:00	7	350	360.0905
08:00	8	500	372.6262
09:00	9	420	369.2072
10:00	10	380	348.6222
11:00	11	300	318.1419
12:00	12	250	291.1620
13:00	13	220	281.2593
14:00	14	200	296.4157
15:00	15	280	335.1786
16:00	16	400	387.3647
17:00	17	550	440.2893
18:00	18	600	485.3281

(c-2) Validate your solution by hand

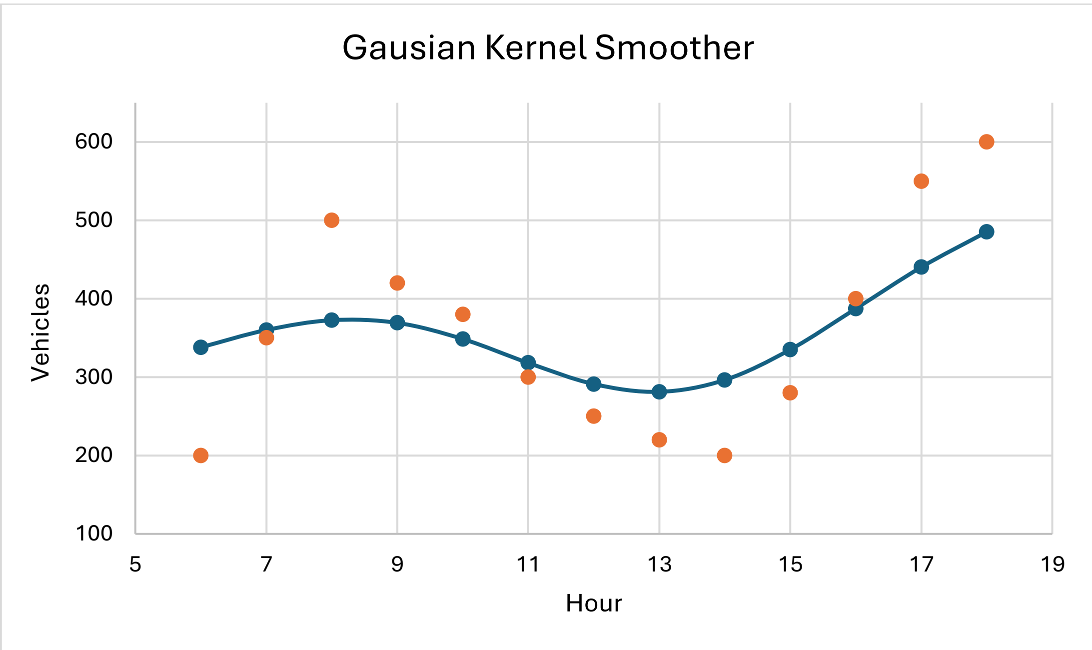
The estimator for f(·), denoted as $\hat{f}_\lambda(\cdot)$, is defined as follows:

$$\hat{f}_\lambda(\cdot) = \frac{\sum_{i=1}^{n} w_i y_i}{\sum_{j=1}^{n} w_i}$$

The weights are defined as $w_i = exp\left(-\frac{(x-x_i)^2}{\lambda}\right)$, i.e. the window is infinitely wide, but distant observations obtain little weight.

			06:00		07:00		08:00		09:00		10:00	
Time	Hour (xi)	Vehicles (y,)	wi (xi =6)	w*y	wi (xi =7)	w*y	wi (xi =8)	w*y	wi (xi =9)	w*y	wi (xi =10)	w*y
)6:00	6	200	1.0000	200.0000	0.8825	176.4994	0.6065	121.3061	0.3247	64.9305	0.1353	27.0671
)7:00	7	350	0.8825	308.8739	1.0000	350.0000	0.8825	308.8739	0.6065	212.2857	0.3247	113.6284
)8:00	8	500	0.6065	303.2653	0.8825	441.2485	1.0000	500.0000	0.8825	441.2485	0.6065	303.2653
)9:00	9	420	0.3247	136.3540	0.6065	254.7429	0.8825	370.6487	1.0000	420.0000	0.8825	370.6487
.0:00	10	380	0.1353	51.4274	0.3247	123.3679	0.6065	230.4817	0.8825	335.3488	1.0000	380.0000
.1:00	11	300	0.0439	13.1811	0.1353	40.6006	0.3247	97.3957	0.6065	181.9592	0.8825	264.7491
.2:00	12	250	0.0111	2.7772	0.0439	10.9842	0.1353	33.8338	0.3247	81.1631	0.6065	151.6327
.3:00	13	220	0.0022	0.4812	0.0111	2.4440	0.0439	9.6661	0.1353	29.7738	0.3247	71.4235
.4:00	14	200	0.0003	0.0671	0.0022	0.4375	0.0111	2.2218	0.0439	8.7874	0.1353	27.0671
.5:00	15	280	0.0000	0.0112	0.0003	0.0939	0.0022	0.6125	0.0111	3.1105	0.0439	12.3023
.6:00	16	400	0.0000	0.0015	0.0000	0.0160	0.0003	0.1342	0.0022	0.8750	0.0111	4.4436
.7:00	17	550	0.0000	0.0001	0.0000	0.0020	0.0000	0.0220	0.0003	0.1845	0.0022	1.2031
.8:00	18	600	0.0000	0.0000	0.0000	0.0002	0.0000	0.0022	0.0000	0.0240	0.0003	0.2013
Sum			3.0066	1016.4402	3.8891	1400.4371	4.4957	1675.1988	4.8203	1779.6910	4.9556	1727.6321
Smoothed Value			338.0665		360.0905		372.6262		369.2072		348.6222	

(c-3) Draw your solution on a scatter plot of the data



(c-4) Validate your solution in R

------ (Code and Output shown in the following section) -----

(d) Create the Gaussian kernel smoother with R, using the function ksmooth()

------ (Code and Output shown in the following section) -----

(e) Use a LOESS smoother for this data set

------ (Code and Output shown in the following section) -----

SHC 798 Assignment 1, 2025

Richard Lubega

2025-07-14

SHC 798 Assignment 1, 2025

Part 2: Data Smoothing

Traffic Flow Data Analysis

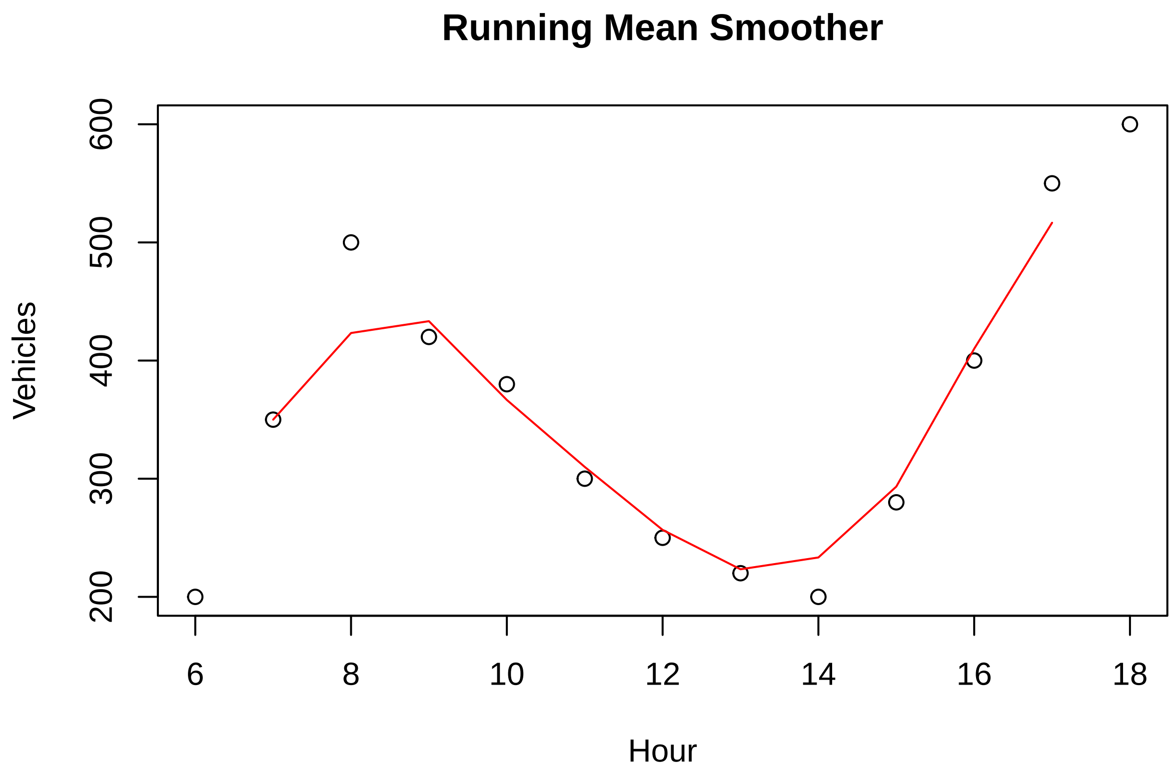
```r
# Traffic flow data
hour <- 6:18
vehicles <- c(200, 350, 500, 420, 380, 300, 250, 220, 200, 280, 400, 550, 600)
traffic <- data.frame(hour, vehicles)
```

(a) Validating in R

```r
# Compute running mean using window width of 3
traffic$smoothed <- stats::filter(traffic$vehicles, rep(1/3, 3), sides = 2)
print(traffic)
```

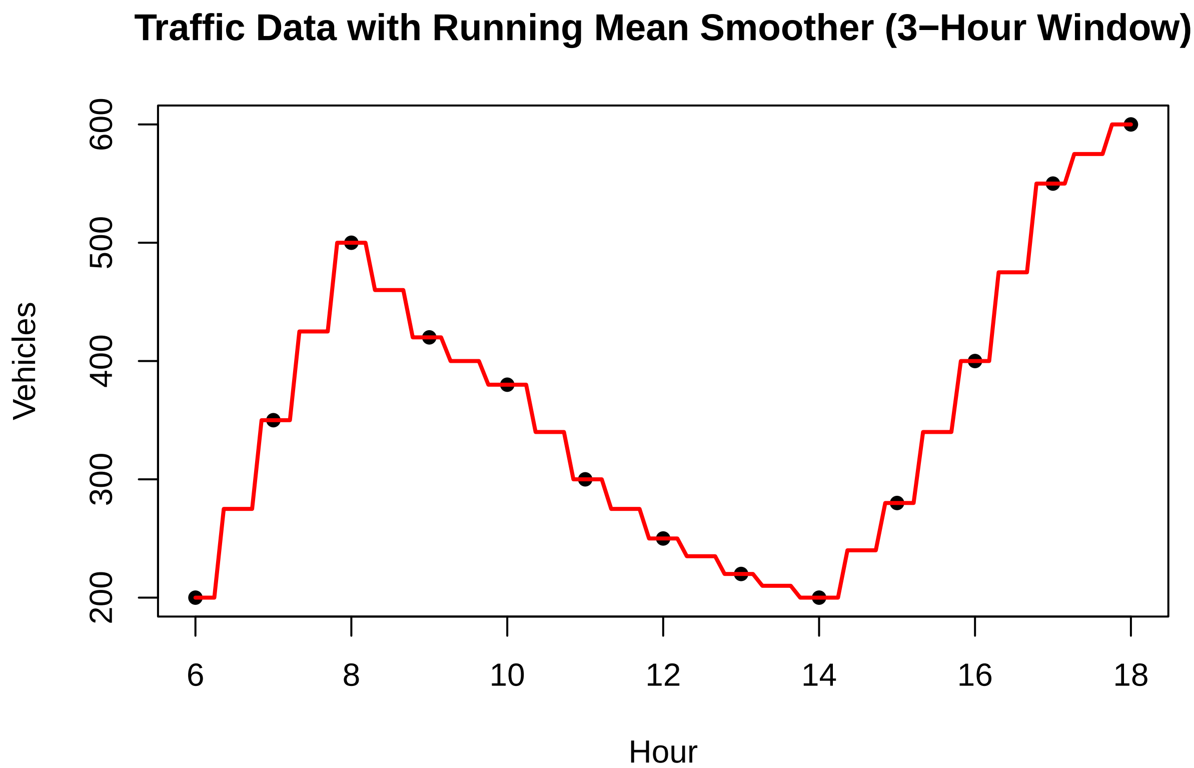
```
##    hour vehicles smoothed
## 1     6      200       NA
## 2     7      350 350.0000
## 3     8      500 423.3333
## 4     9      420 433.3333
## 5    10      380 366.6667
## 6    11      300 310.0000
## 7    12      250 256.6667
## 8    13      220 223.3333
## 9    14      200 233.3333
## 10   15      280 293.3333
## 11   16      400 410.0000
## 12   17      550 516.6667
## 13   18      600       NA
```

```r
# Scatter Plot
plot(hour, vehicles, type = "p", main = "Running Mean Smoother", xlab = "Hour", ylab = "Vehicles")
lines(traffic$hour, traffic$smoothed, type = "l", col = "red")
```



(b) Using R to compute a running mean smoother using ksmooth()

```
# Compute running mean smoother using ksmooth
smoothed <- ksmooth(traffic$hour, traffic$vehicles, kernel = "box", bandwidth = 1.5)

# Create a data frame with smoothed values
traffic_smoothed <- data.frame(hour = smoothed$x, vehicles_smoothed = smoothed$y)

# Plot scatter plot of original data and overlay the smoothed line
plot(traffic$hour, traffic$vehicles,
     xlab = "Hour", ylab = "Vehicles",
     main = "Traffic Data with Running Mean Smoother (3-Hour Window)",
     pch = 16, col = "black")  # Scatter plot
lines(traffic_smoothed$hour, traffic_smoothed$vehicles_smoothed,
      col = "red", lwd = 2)  # Smoothed line
```



(c) Validating in R

```
# Checking the Gaussian kernel smoother (does not have the normalization constant)
gaussian_kernel <- function(xi, x, y, h) {
  weights <- exp(-((x - xi)^2) / (2 * h^2))
  sum(weights * y) / sum(weights)}

# Compute values
xi_values <- 6:18
kernel_values <- sapply(xi_values, function(xi) gaussian_kernel(xi, hour, vehicles, h = 2))

# # Validating with h = 2
validation <- data.frame(xi = xi_values,   manual_values = round(kernel_values, 4))
print(validation)
```

```
##    xi manual_values
## 1   6      338.0665
## 2   7      360.0905
## 3   8      372.6262
## 4   9      369.2072
## 5  10      348.6222
## 6  11      318.1419
## 7  12      291.1620
## 8  13      281.2593
## 9  14      296.4157
```

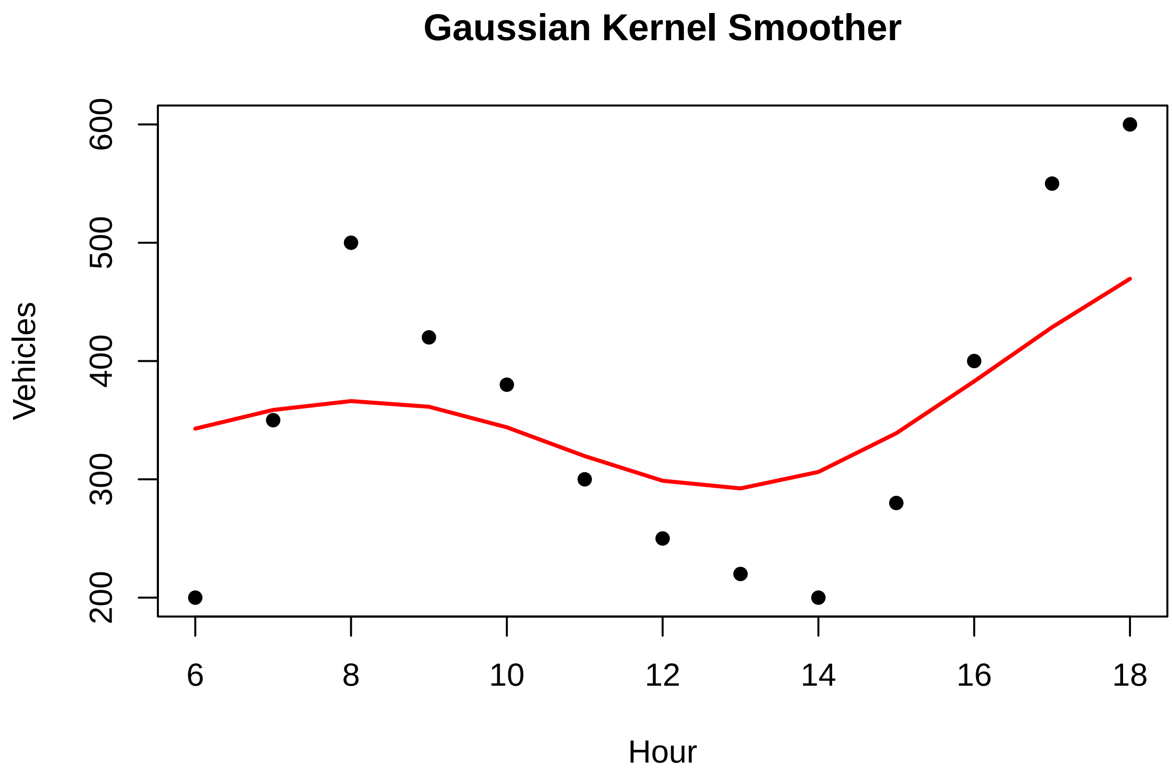
```
## 10 15      335.1786
## 11 16      387.3647
## 12 17      440.2893
## 13 18      485.3281
```

(d) Create the Gaussian kernel smoother with R, using the function ksmooth()

```r
# Gaussian kernel smoother with bandwidth = 2
smoothed <- ksmooth(traffic$hour, traffic$vehicles, kernel = "normal", bandwidth = 6, x.points = traffic
traffic_smoothed <- data.frame(hour = smoothed$x, vehicles_smoothed = smoothed$y)
print(traffic_smoothed)
```

```
##    hour vehicles_smoothed
## 1     6          342.8414
## 2     7          358.6636
## 3     8          366.1537
## 4     9          361.3546
## 5    10          343.9766
## 6    11          319.6306
## 7    12          298.7996
## 8    13          292.3215
## 9    14          306.2371
## 10   15          338.9925
## 11   16          382.8491
## 12   17          428.5728
## 13   18          469.5297
```

```r
# Plot original data and smoothed curve
plot(traffic$hour, traffic$vehicles, xlab = "Hour", ylab = "Vehicles", main = "Gaussian Kernel Smoother
     pch = 16, col = "black")
lines(traffic_smoothed$hour, traffic_smoothed$vehicles_smoothed, col = "red", lwd = 2)
```



(e) Using LOESS smoother

```
# Defining LOESS smoothers with varying degrees and spans
loess_1_03 <- loess(vehicles ~ hour, data = traffic, degree = 1, span = 0.3)
```

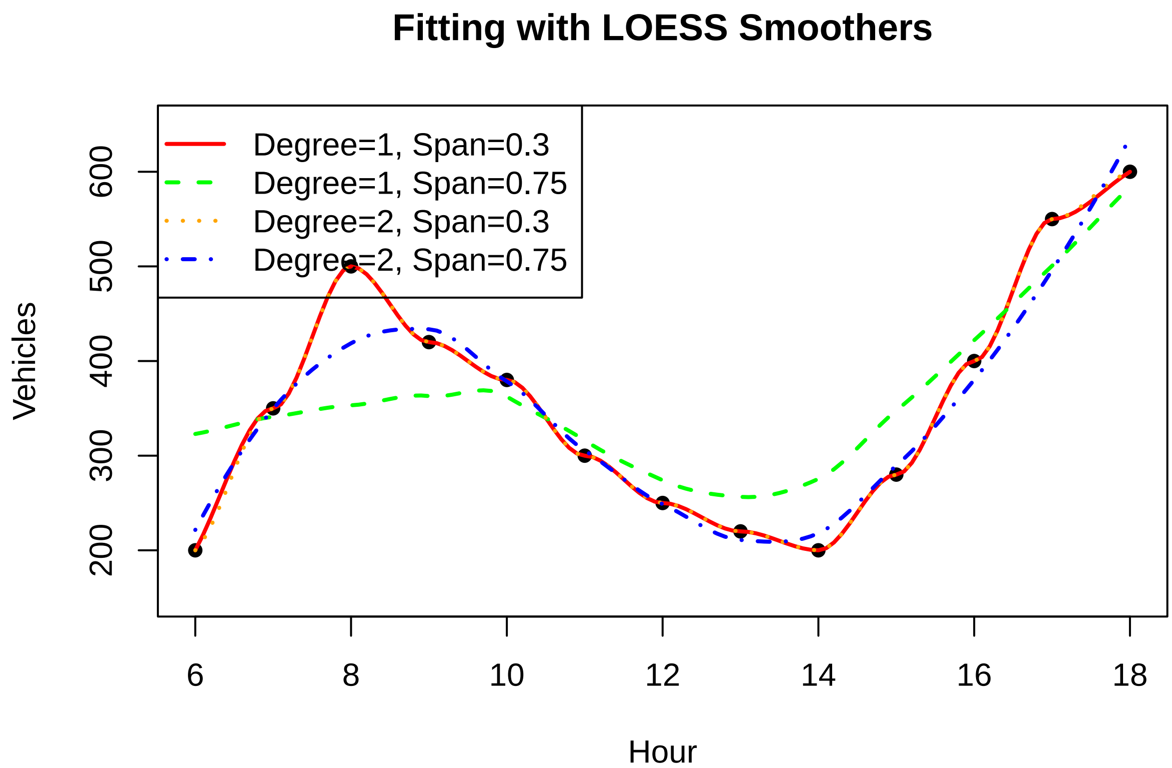
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : pseudoinverse used at 12

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : neighborhood radius 1

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : reciprocal condition number -0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : There are other near singularities as well. 1
```

```
loess_1_075 <- loess(vehicles ~ hour, data = traffic, degree = 1, span = 0.75)
loess_2_03 <- loess(vehicles ~ hour, data = traffic, degree = 2, span = 0.3)
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : span too small.  fewer data values than degrees of freedom.
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : pseudoinverse used at 5.94

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : neighborhood radius 2.06

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : There are other near singularities as well. 4.2436
```

```r
loess_2_075 <- loess(vehicles ~ hour, data = traffic, degree = 2, span = 0.75)

# Predicting smoothed values at integer hours
hours <- seq(6, 18, by = 0.1)
pred_1_03 <- predict(loess_1_03, newdata = data.frame(hour = hours))
pred_1_075 <- predict(loess_1_075, newdata = data.frame(hour = hours))
pred_2_03 <- predict(loess_2_03, newdata = data.frame(hour = hours))
pred_2_075 <- predict(loess_2_075, newdata = data.frame(hour = hours))

# Creating scatter plot
plot(traffic$hour, traffic$vehicles, xlab = "Hour", ylab = "Vehicles", main = "Fitting with LOESS Smoot
     pch = 16, col = "black", ylim = c(150, 650))

# Adding LOESS smoother lines
lines(hours, pred_1_03, col = "red", lwd = 2, lty = 1)
lines(hours, pred_1_075, col = "green", lwd = 2, lty = 2)
lines(hours, pred_2_03, col = "orange", lwd = 2, lty = 3)
lines(hours, pred_2_075, col = "blue", lwd = 2, lty = 4)
legend("topleft",
       legend = c("Degree=1, Span=0.3",
                  "Degree=1, Span=0.75",
                  "Degree=2, Span=0.3",
                  "Degree=2, Span=0.75"),
       col = c("red", "green", "orange", "blue"),
       pch = c(NA, NA, NA, NA),
       lty = c(1, 2, 3, 4),
       lwd = c(2, 2, 2, 2))
```



Intepreting the behaviour

- From the LOESS smoothing plots, the span controls how much of the data is used in each local fit Smaller spans like, 0.3) produce a more wiggly curve that closely follows the data but risks overfitting, while larger spans like 0.75 create smoother trends that may underfit local variation. The degree determines the type of local regression. A degree of 1 fits local lines, while degree of 2 fits local quadratic functions (more flexible). Smaller spans and higher degrees increase sensitivity to local patterns, while larger spans and lower degrees prioritize smoothness. A span = 0.75 and degree = 2 appears to fit the data well.

SHC 798 Assignment 1, 2025

Richard Lubega

2025-07-14

SHC 798 Assignment 1, 2025

Part 3: Simple regression

Question 1

```r
# The dataset cars
# A SLR to analyse the relationship between speed and stopping distance

cat("\n A SLR between speed and stopping distance \n")
```

```
##
##  A SLR between speed and stopping distance
```

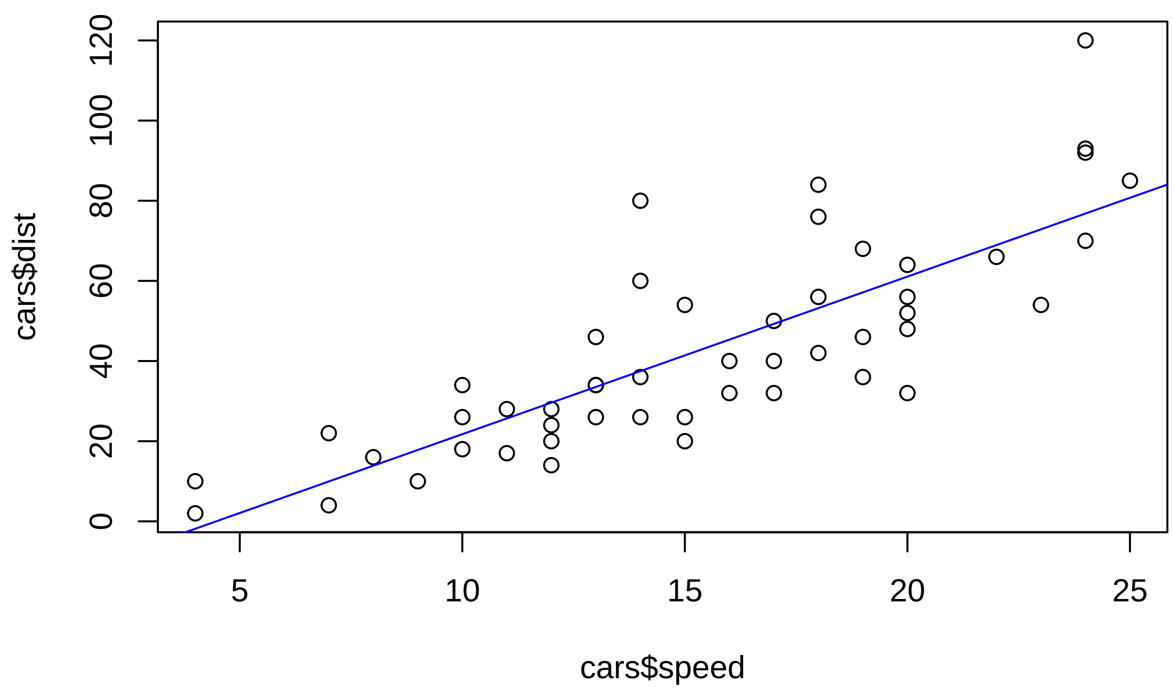
```r
lm_s.sd <- lm(dist ~ speed, data = cars)
summary(lm_s.sd)
```

```
##
## 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
```

```r
cat("\n ===  SLR Model Plot  === \n")
```

```
##
## ===  SLR Model Plot  ===
```

```r
plot(cars$speed, cars$dist)
abline(lm_s.sd, col = "blue")
```



(a) From the model summary, **Multiple R-squared**: 0.6511, Adjusted R-squared: 0.6438

Thus, **65.11%** of the variation in stopping distance is explained by speed

(b) **Intercept** (-17.5791): This means that for a theoretical speed of 0 mph the predicted stopping distance is -17.5791 feet. This is not practically rational but ensures the regression line fits the data best within the observed speed range. It is not meaningful to extrapolate to speed = 0.

- It's p-value (0.0123) is small and statistically significant at the 5% level, but its practical importance is limited.

Slope (3.9324): For every 1 mph increase in speed, stopping distance increases by about 3.9324 feet. Higher driving speeds require longer stopping distances.

- The p-value (1.49e-12) is much smaller than 0.05 (even at a 1% significance level), so the relationship between speed and stopping distance is statistically significant.We reject the null hypothesis that speed has no effect on stopping distance. Thus, speed has an considerable impact on stopping distance.

```r
# Predicting stopping distance for  speed = 20 mph;  compute a 95% prediction interval.
cat("\n ===  Stopping distance at a speed of 20 mp and the 95% prediction interval ===\n ")
```

(c)

```
##
##  ===  Stopping distance at a speed of 20 mp and the 95% prediction interval ===
##
```

```r
predict(lm_s.sd, newdata = data.frame(speed = 20), interval = "prediction", level = 0.95)
```

```
##        fit      lwr      upr
## 1 61.06908 29.60309 92.53507
```

```r
cat("\n Evaluating Model Assumptions \n")
```

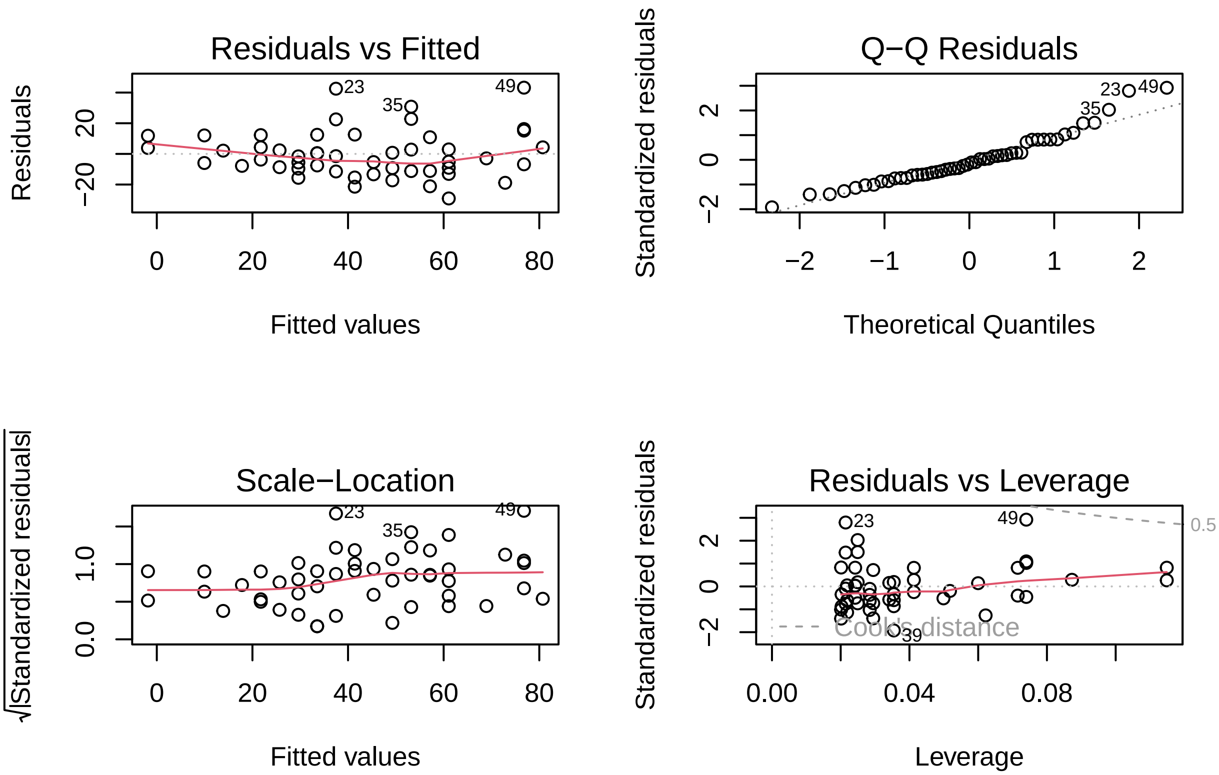
(d)

```
##
##  Evaluating Model Assumptions
```

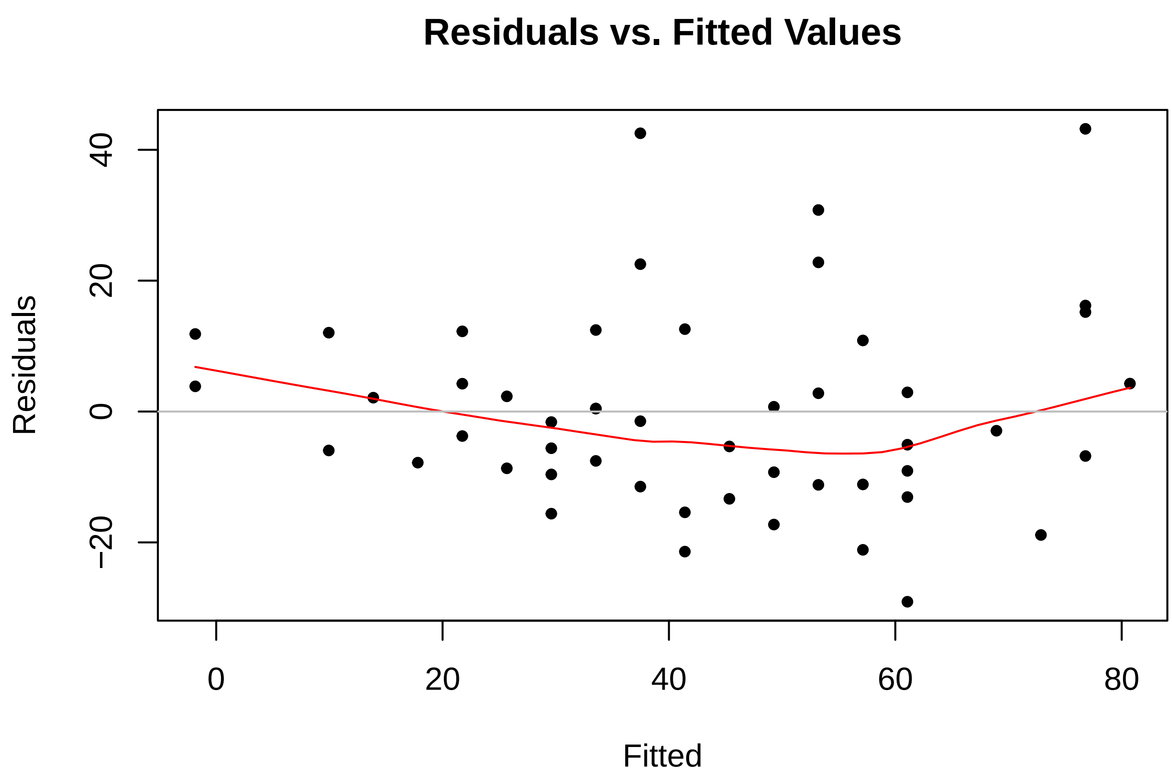
```r
cat("\n ===  Model Diagnostics Plots  === \n")
```

```
##
##  ===  Model Diagnostics Plots  ===
```

```r
# Diagnostics plots
par(mfrow = c(2,2))
plot(lm_s.sd)
```

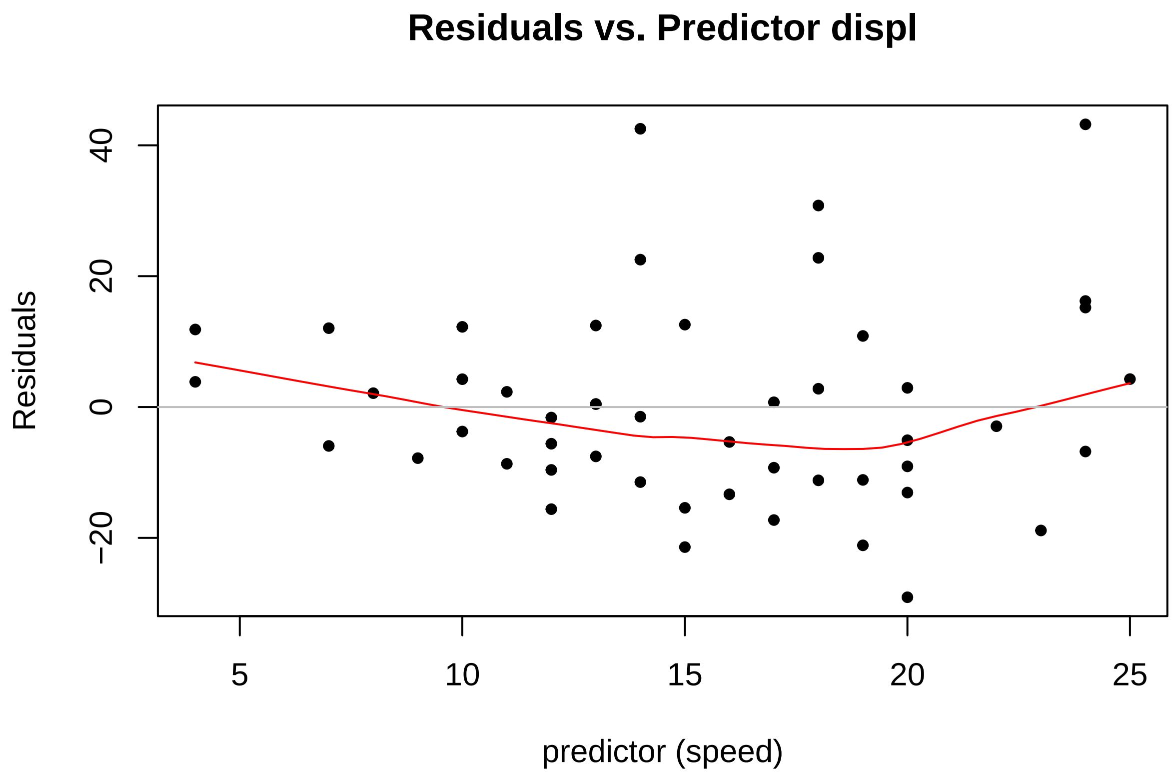



```
par(mfrow = c(1,1))
# Tukey-Anscombe Plot
plot(lm_s.sd$fitted.values, lm_s.sd$residuals, xlab="Fitted", ylab="Residuals", pch=20) +
  title("Residuals vs. Fitted Values") +
  lines(loess.smooth(lm_s.sd$fitted.values, lm_s.sd$residuals),col="red") +
  abline(h=0, col="grey")
```



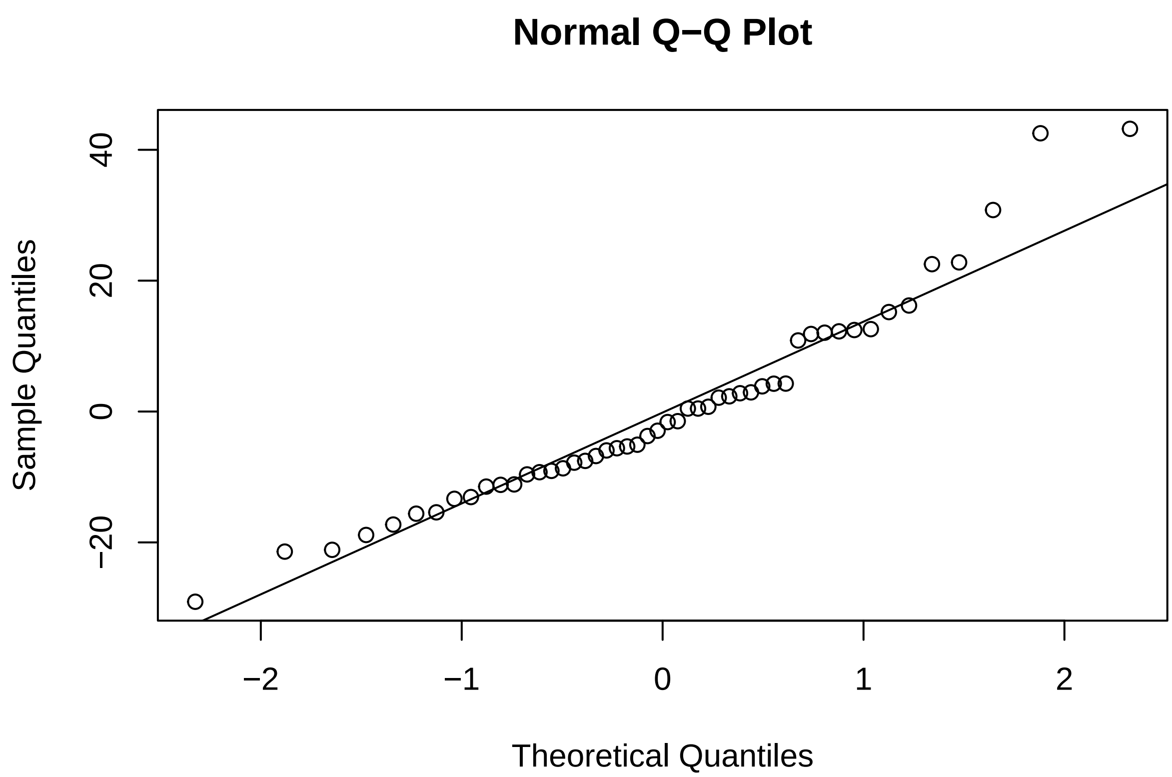

```
## integer(0)
```

```r
# Residuals vs. Predictor Plot
plot(cars$speed, lm_s.sd$residuals, xlab="predictor (speed)", ylab="Residuals", pch=20) +
  title("Residuals vs. Predictor displ") +
  lines(loess.smooth(cars$speed, lm_s.sd$residuals),col="red") +
  abline(h=0, col="grey")
```




```
## integer(0)
```

```
# Quantile-Quantile Plot
qqnorm(lm_s.sd$residuals) #Quantile-Quantile Plot
qqline(lm_s.sd$residuals) # adds the diagonal line
```



Model Assumption Evaluation

1. **Linearity** — *From the Tukey-Anscombe Plot (Residuals vs. Fitted)*:

- By inspection, the residuals generally hover around the zero line which suggests that they likely approximate a mean of zero. There is, however, slight curvature (a kink) in the red LOESS smoother line (deviation from the horizontal) which implies mild (misspecified) non-linearity. This is confirmed by the systematic misprediction in the middle (overpredicting) and the extremes (underpredicting). In this case, there is is a clear violation of the linearity ($E[E_i] = 0$) assumption; a straight line is not the correct fit to the data and the model ought to be improved.
- **Transformation**: Add a quadratic term ($dist = \beta_0 + \beta_1 \cdot speed + \beta_2 \cdot speed^2 + E_i$) to fix this and improve the model (as for this pair, the true relationship is quadratic). This constitutes a multiple linear regression problem.

2. **Homoskedasticity** — *From the Scale-Location Plot*:

- The red line is slightly upward-trending, indicating that variance increases with fitted values (minor heteroscedasticity)
- The Tukey-Anscombe plot also seems to indicate that the scatter is not constant for the entire range of speed/fitted values (less scatter for lower values and more scatter for higher values). There is an obvious violation of homoskedasticity.
- **Transformation**: Log-transform on dist (since stopping distance cannot be negative)

3. **Independence**

- Since the data is *not time-dependent*, residual independence is likely satisfied (no autocorrelation expected)
- **Transformation**: None needed

4. **Normality** — *From the Q-Q Plot*:

- The bulk of the residuals (in the central region) are approximately Gaussian distributed. A noticeable deviations (or outliers) at the upper tail indicates right skewness hence departure from normality. The assumption of Gaussian errors is slightly violated by the model due to this moderate non-normality.
- **Transformation**: Log-transform on dist to correct right-skewness (improve normality and heteroskedasticity)

Model Evaluation and Improvements

- Therefore, this model (lm_s.sd = dist ~ speed) has minor assumption violations (non-linearity, heteroscedasticity, non-normality).

- Suggested transformations like the log(dist) ~ speed and a quadratic term could be made and the diagnostics re-checked. The best model is the one with the most stable residuals, best-fulfilled assumptions, and highest adjusted R^2.

SHC 798 Assignment 1, 2025

Richard Lubega

2025-07-14

SHC 798 Assignment 1, 2025

Part 3: Simple regression

Question 2

```r
# Load the housing.rda data file
load(file.choose())
head(housing) # View first few rows of the dataset
```

```
##   size  price
## 1  125 132358
## 2  139 153827
## 3  237 154427
## 4  152 143527
## 5  154 131707
## 6  248 159368
```

```r
summary(housing) # Get an overview of the dataset
```

```
##       size           price
##  Min.   : 74.0   Min.   :109141
##  1st Qu.:128.0   1st Qu.:133684
##  Median :151.5   Median :146803
##  Mean   :158.3   Mean   :148389
##  3rd Qu.:182.2   3rd Qu.:159955
##  Max.   :286.0   Max.   :208648
```

```r
str(housing)
```

```
## 'data.frame':    100 obs. of  2 variables:
##  $ size : num  125 139 237 152 154 248 170 102 121 130 ...
##  $ price: num  132358 153827 154427 143527 131707 ...
```

```r
# use regression analysis to explore the relationship between house size and price.
cat("A SLR between house price and house size \n")
```

```
## A SLR between house price and house size
```

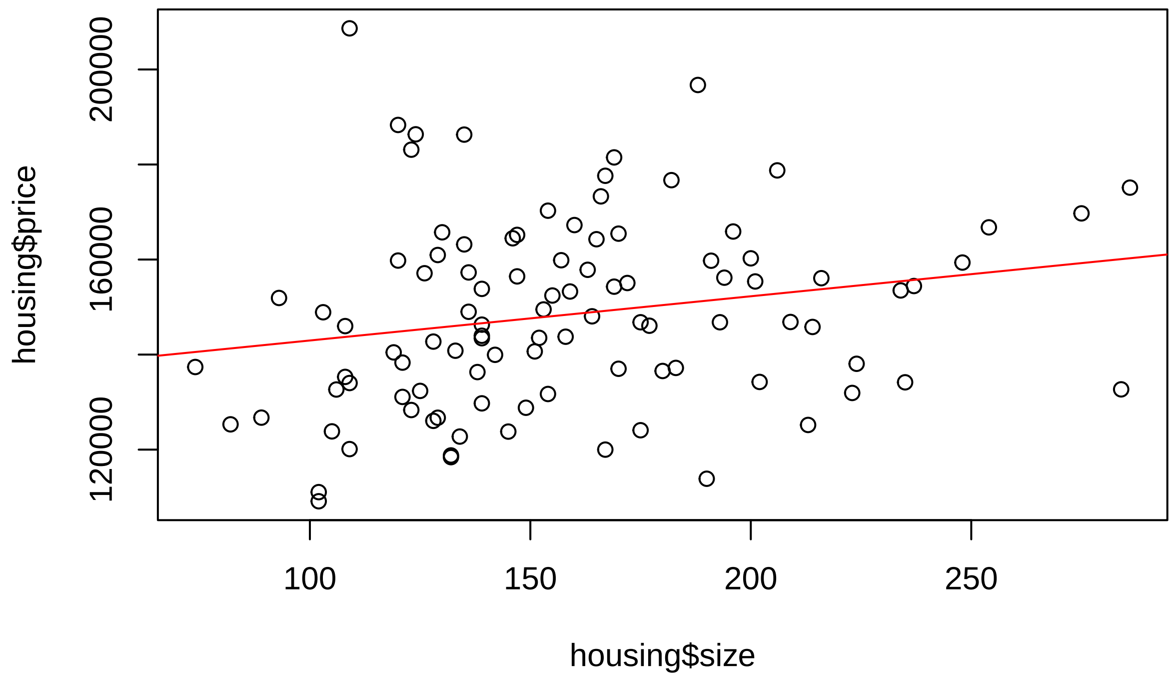
```r
lm_p.s <- lm(price ~ size, data = housing)
summary(lm_p.s)
```

```
##
## Call:
## lm(formula = price ~ size, data = housing)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -37465 -14199  -2284  11120  64842
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 133667.87    7271.00  18.384   <2e-16 ***
## size            93.01      44.26   2.102   0.0381 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19500 on 98 degrees of freedom
## Multiple R-squared:  0.04313,    Adjusted R-squared:  0.03336
## F-statistic: 4.417 on 1 and 98 DF,  p-value: 0.03814
```

```r
cat("===  SLR Model Plot  === \n")
```

```
## ===  SLR Model Plot  ===
```

```r
plot(housing$size, housing$price) +
  abline(lm_p.s, col = "red")
```




```
## integer(0)
```

(a) Comment on the Model Summary

Regression Coefficients:

- **Intercept** (133667.87): This means that for a theoretical house size of 0 units the predicted house price is 133667.87 units. This is not practically useful but ensures the regression line fits the data best within the observed size range. It is not meaningful to extrapolate to house size = 0.
  - It's *p-value* (< 2e-16) is small and statistically significant at the 5% level, but its practical value is limited.
- **Slope** (93.01): For every 1 unit increase in house size, the house price increases by about 93.01 units Bigger houses cost higher to buy.
  - The *p-value* (0.0381) is smaller than 0.05 (even at a 1% significance level). We reject the null hypothesis at the 5% significance level. This means that house size has a statistically significant effect on house price, and we can be fairly confident (with 95% confidence) that the relationship isn't due to chance.

Statistical Significance:

- From the F-statistic (4.417), the p-value for size is small (0.03814 < 0.05), meaning the model is statistically significant at the 5% level.

Model Goodness of Fit:

- The **multiple R-squared** (0.04313) and **adjusted R-squared** (0.03336) indicate that only 4.313% of the variability in house prices is explained by house size. This suggests the model is very weak in explanatory power and that other factors likely have a much bigger influence.

(b) Residual Diagnostics

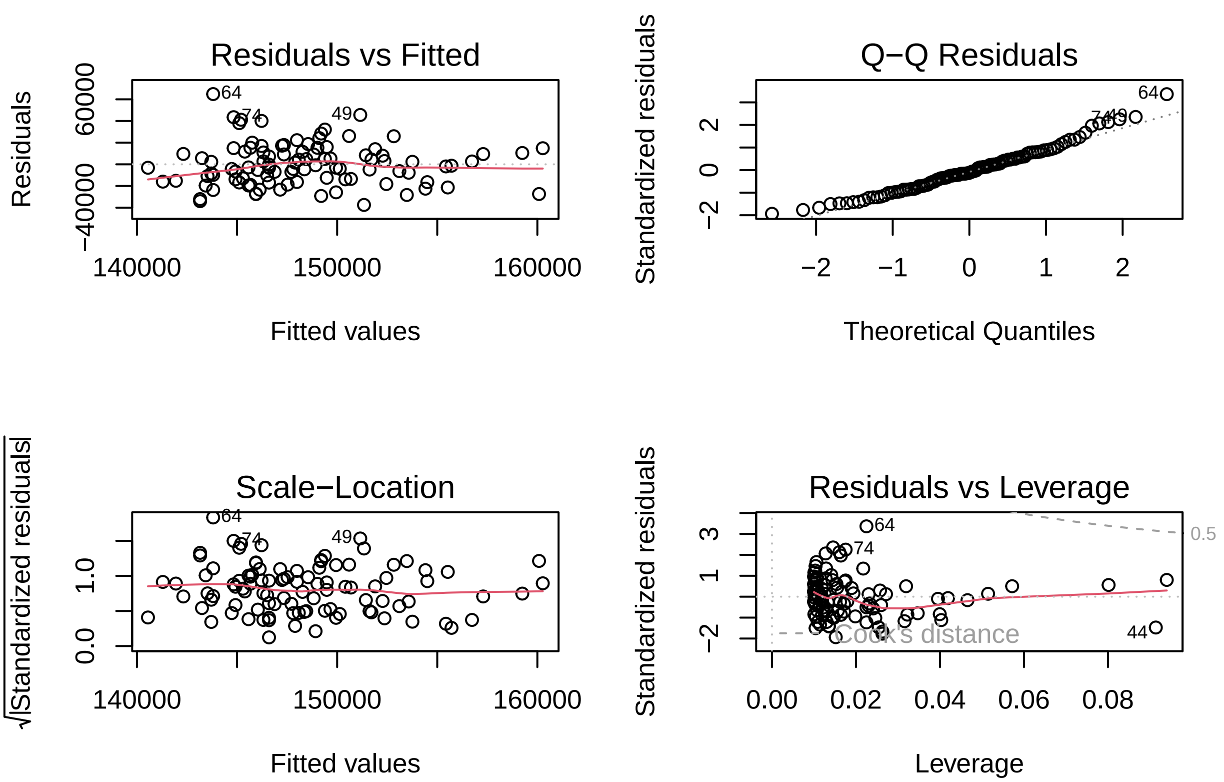
```
# Perform residual diagnostics and comment on model assumptions
cat("Performing Model Diagnostics \n")
```

```
## Performing Model Diagnostics
```

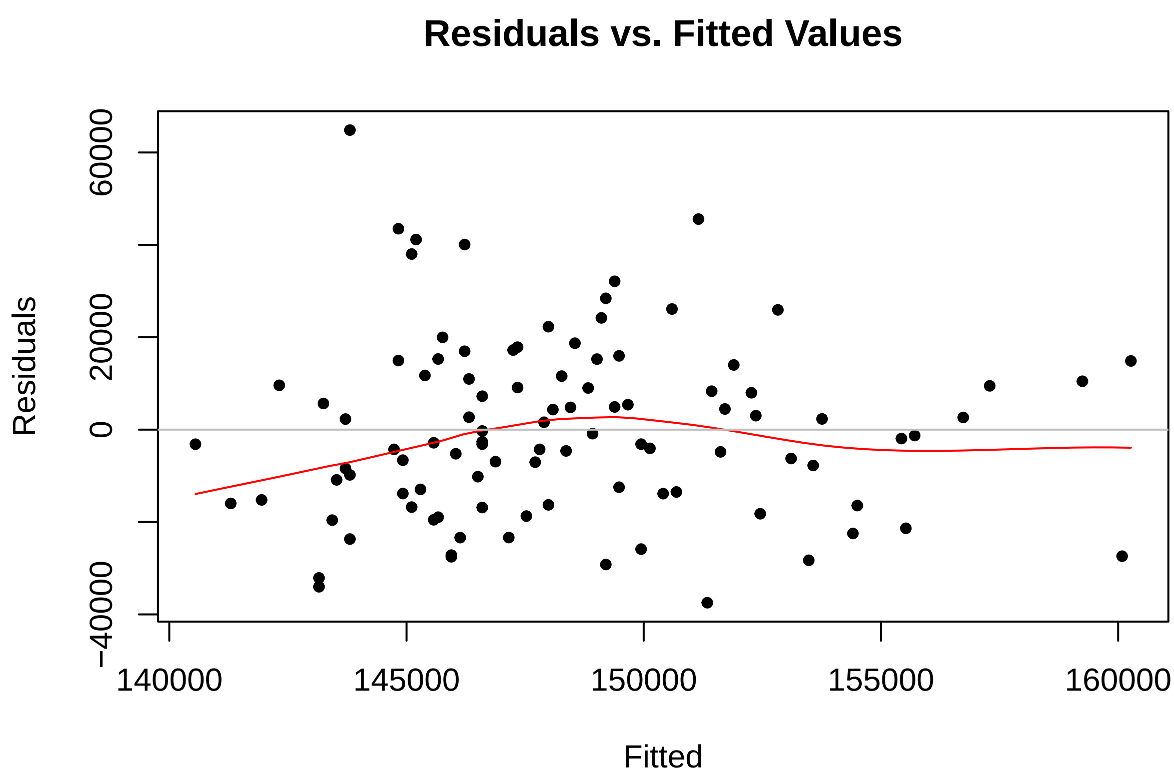
```
cat(" ===  Model Diagnostics Plots  === \n")
```

```
##  ===  Model Diagnostics Plots  ===
```

```
# Diagnostics plots
par(mfrow = c(2,2))
plot(lm_p.s)
```

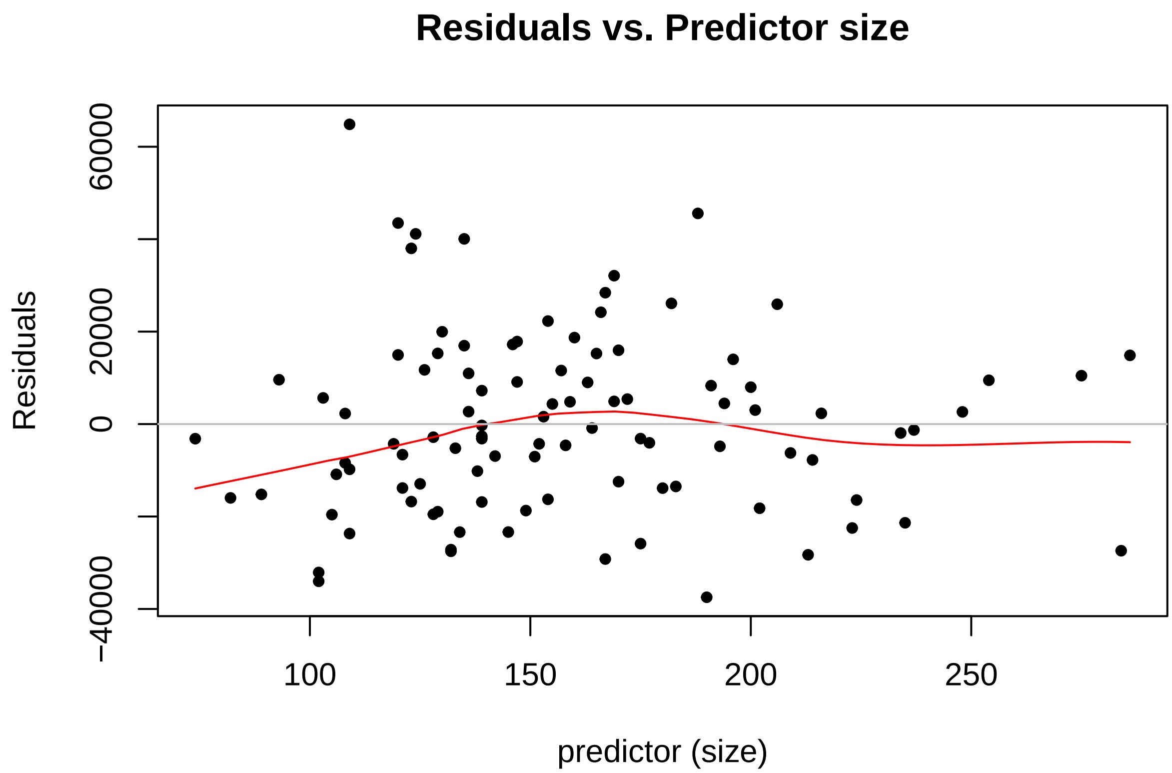


```r
par(mfrow = c(1,1))
# Tukey-Anscombe Plot
plot(lm_p.s$fitted.values, lm_p.s$residuals, xlab="Fitted", ylab="Residuals", pch=20) +
  title("Residuals vs. Fitted Values") +
  lines(loess.smooth(lm_p.s$fitted.values, lm_p.s$residuals),col="red") +
  abline(h=0, col="grey")
```



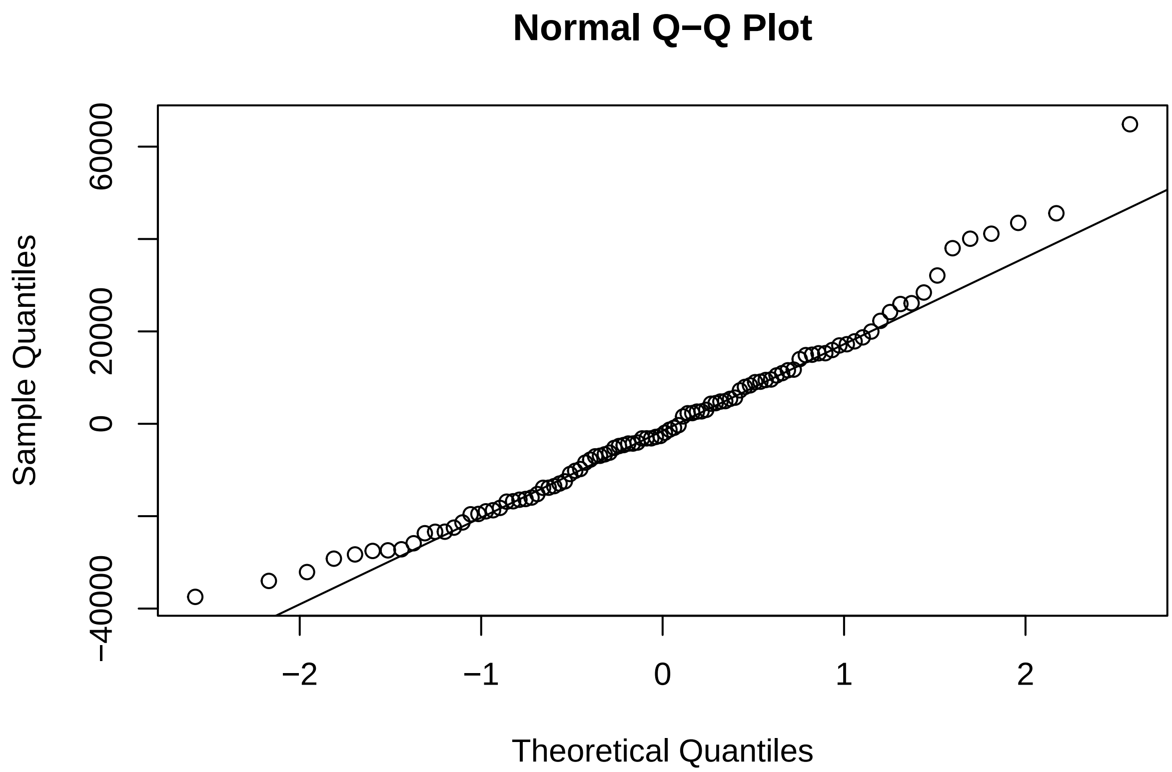

```
## integer(0)
```

```r
# Residuals vs. Predictor Plot
plot(housing$size, lm_p.s$residuals, xlab="predictor (size)", ylab="Residuals", pch=20) +
  title("Residuals vs. Predictor size") +
  lines(loess.smooth(housing$size, lm_p.s$residuals),col="red") +
  abline(h=0, col="grey")
```




```
## integer(0)
```

```
# Quantile-Quantile Plot
qqnorm(lm_p.s$residuals) #Quantile-Quantile Plot
qqline(lm_p.s$residuals) # adds the diagonal line
```




```r
# Evaluating Model Assumptions
cat("Model Assumption Evaluation \n")
```

```
## Model Assumption Evaluation
```

Comments on Model Assumptions

1. **Linearity** — *From the Tukey-Anscombe Plot (Residuals vs. Fitted)*:

- From the plot, the residuals generally hover around the zero line which suggests that the $E[E_i] = 0$ is approximately met. However, LOESS smoother line has a kink in the middle and largely deviates from the horizontal. The residuals for low and high house size (and respective fitted house price) values are systematically negative and they are positive for medium values. The linearity assumption is violated and a straight line is not the correct fit to the data. The model may be improved by variable transformation.

2. **Homoskedasticity** — *From the Tukey-Anscombe plot and the Scale-Location Plot*:

- The Tukey-Anscombe plot indicates a more or less constant scatter for the entire range of house size (& fitted) values. There is no obvious violation of homoskedasticity. The red line in the Scale-Location Plot is fairly horizontal which implies constant variance with fitted values (no heteroscedasticity).

3. **Independence**

- The residuals can be considered independent and uncorrelated.

4. **Normality** — *From the Q-Q Plot*:

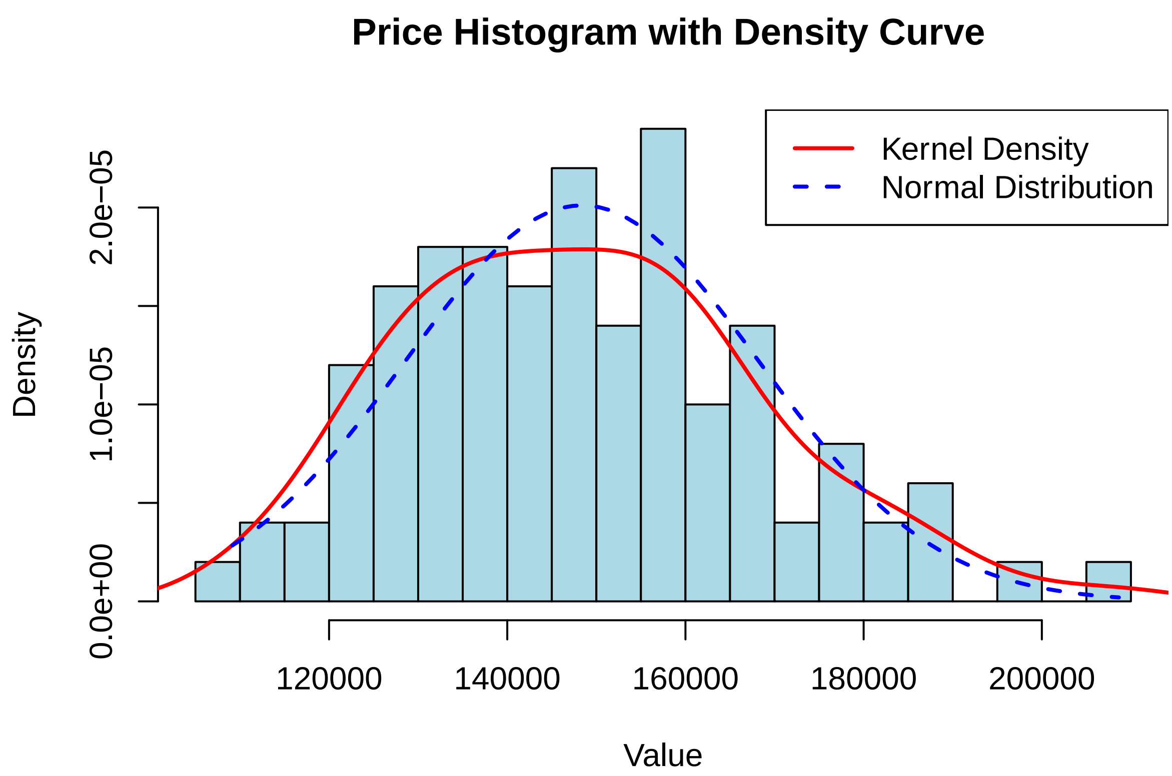
- The bulk of the residuals (in the central region) lie on the 45º line and thereby follow the Gaussian distribution. There are slight deviations (or outliers) at the lower and upper tail which indicate right skewness. The assumption of Gaussian errors is slightly violated by the model due to this moderate non-normality. Despite this, the approximation to normality in the center may be sufficient to validate this model.

(c) Check if a log transformation improves the model fit. Are any of the models useful?

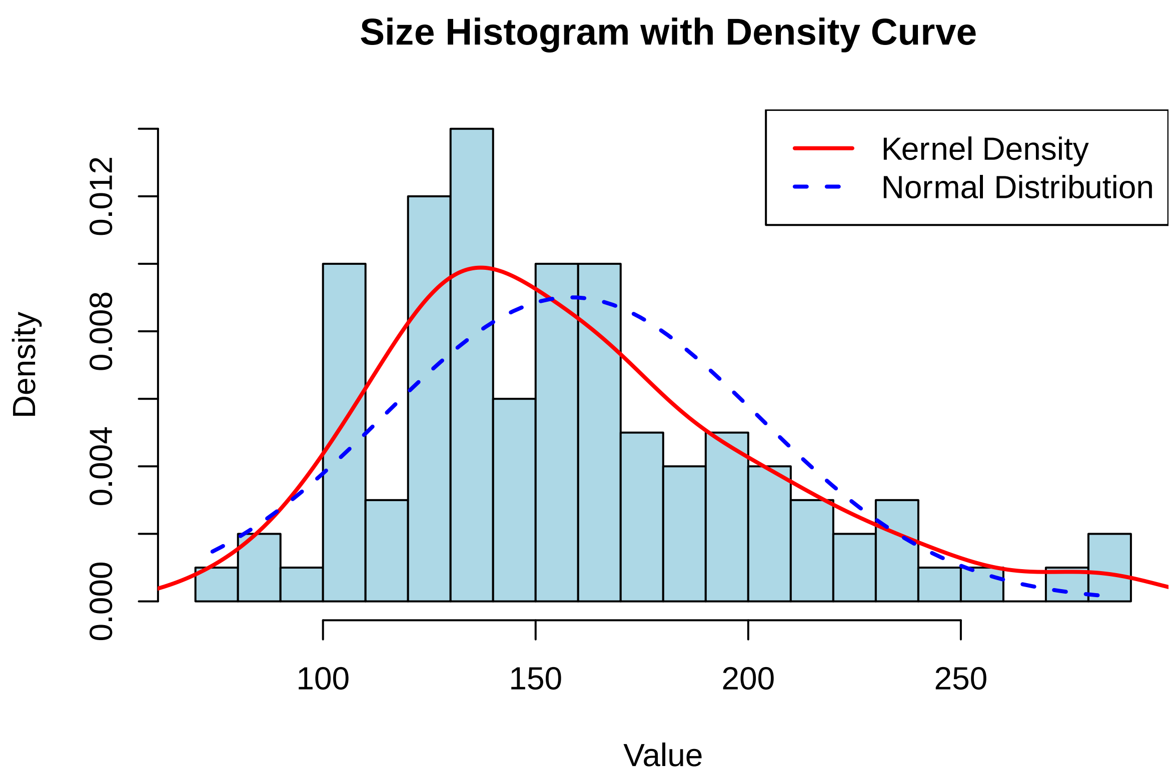
```
# Any right-skewness in the data?
# View distribution (histogram)
cat(" Viewing Paramter Distributions: Check for skewness\n")
```

```
##  Viewing Paramter Distributions: Check for skewness
```

```
# Viewing House prices
hist(housing$price, freq = FALSE, breaks = 30, col = "lightblue",
     main = " Price Histogram with Density Curve", xlab = "Value", ylab = "Density",
     border = "black")

lines(density(housing$price, na.rm = TRUE), col = "red",lwd = 2) # Add density curve



# Adding a normal distribution curve for comparison
h_price <- seq(min(housing$price, na.rm = TRUE), max(housing$price, na.rm = TRUE), length.out = 100)
normal_price <- dnorm(h_price, mean = mean(housing$price, na.rm = TRUE), sd = sd(housing$price, na.rm =
lines(h_price, normal_price, col = "blue", lwd = 2, lty = 2)

# Add legend
legend("topright", legend = c("Kernel Density", "Normal Distribution"), col = c("red", "blue"),
       lwd = 2, lty = c(1, 2))
```




```r
# Viewing House Sizes
hist(housing$size, freq = FALSE, breaks = 30, col = "lightblue",
     main = " Size Histogram with Density Curve", xlab = "Value", ylab = "Density",
     border = "black")

lines(density(housing$size, na.rm = TRUE), col = "red",lwd = 2) # Add density curve

# Adding a normal distribution curve for comparison
h_size <- seq(min(housing$size, na.rm = TRUE), max(housing$size, na.rm = TRUE), length.out = 100)
normal_size <- dnorm(h_size, mean = mean(housing$size, na.rm = TRUE), sd = sd(housing$size, na.rm = TRUI
lines(h_size, normal_size, col = "blue", lwd = 2, lty = 2)

# Add legend
legend("topright", legend = c("Kernel Density", "Normal Distribution"), col = c("red", "blue"),
       lwd = 2, lty = c(1, 2))
```



From the **plots**, the house **price** data is only *slightly right-skewed* (in agreement with the Q-Q plot) while the house **size** data is **clearly right-skewed**. A log-log transform is appropriate for the pair (which are right-skewed variables taking on only positive values).

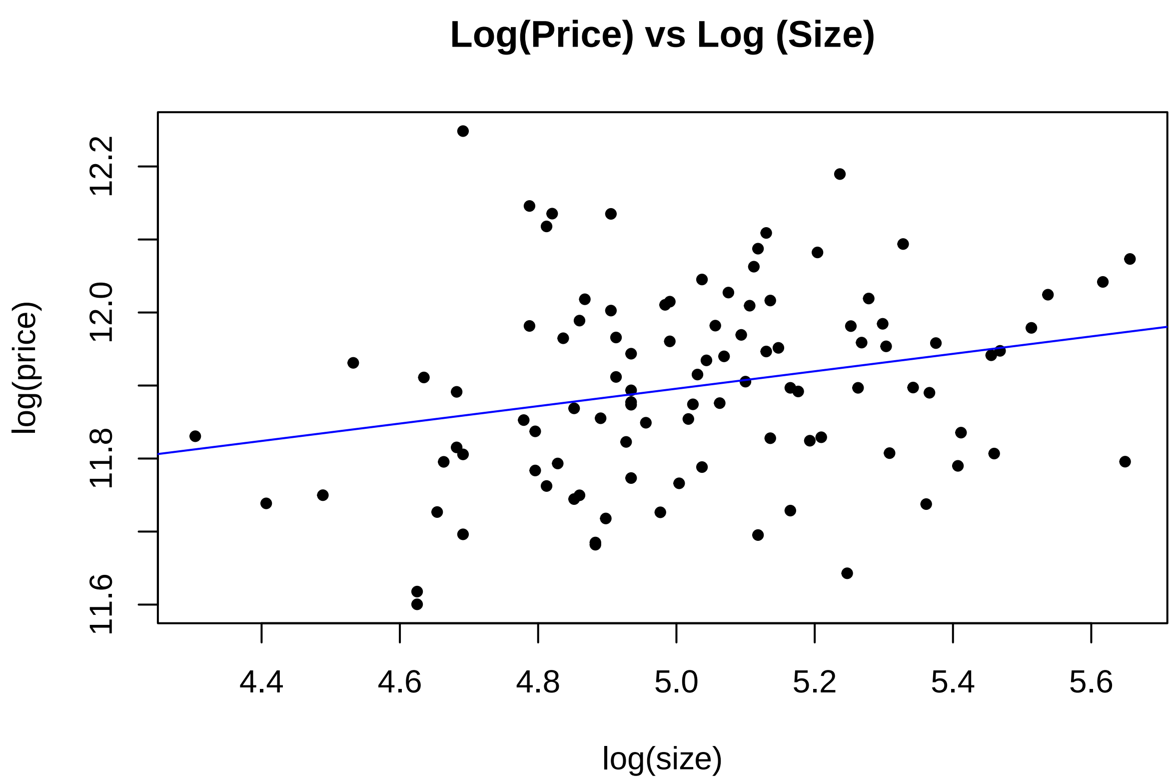
```r
# 1. Log-log Transformation
lg.lg <- lm(log(price) ~ log(size), data = housing)
summary(lg.lg)
```

```
##
## Call:
## lm(formula = log(price) ~ log(size), data = housing)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.28226 -0.08861 -0.00627  0.08258  0.38958
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 11.29914    0.23830   47.42   <2e-16 ***
## log(size)    0.11930    0.04733    2.52   0.0133 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.129 on 98 degrees of freedom
## Multiple R-squared:  0.06088,    Adjusted R-squared:  0.05129
## F-statistic: 6.353 on 1 and 98 DF,  p-value: 0.01334
```

```r
cat("===  Log-log SLR Model Plot  === \n")
```

```
## ===  Log-log SLR Model Plot  ===
```

```r
plot(log(price) ~ log(size), data = housing,  main = "Log(Price) vs Log (Size)", pch=20) +
  abline(lg.lg, col = "blue")
```




```
## integer(0)
```

```r
# 2. Logged-Response Model Transformation
lm_lg <- lm(log(price) ~ size, data = housing)
summary(lm_lg)
```

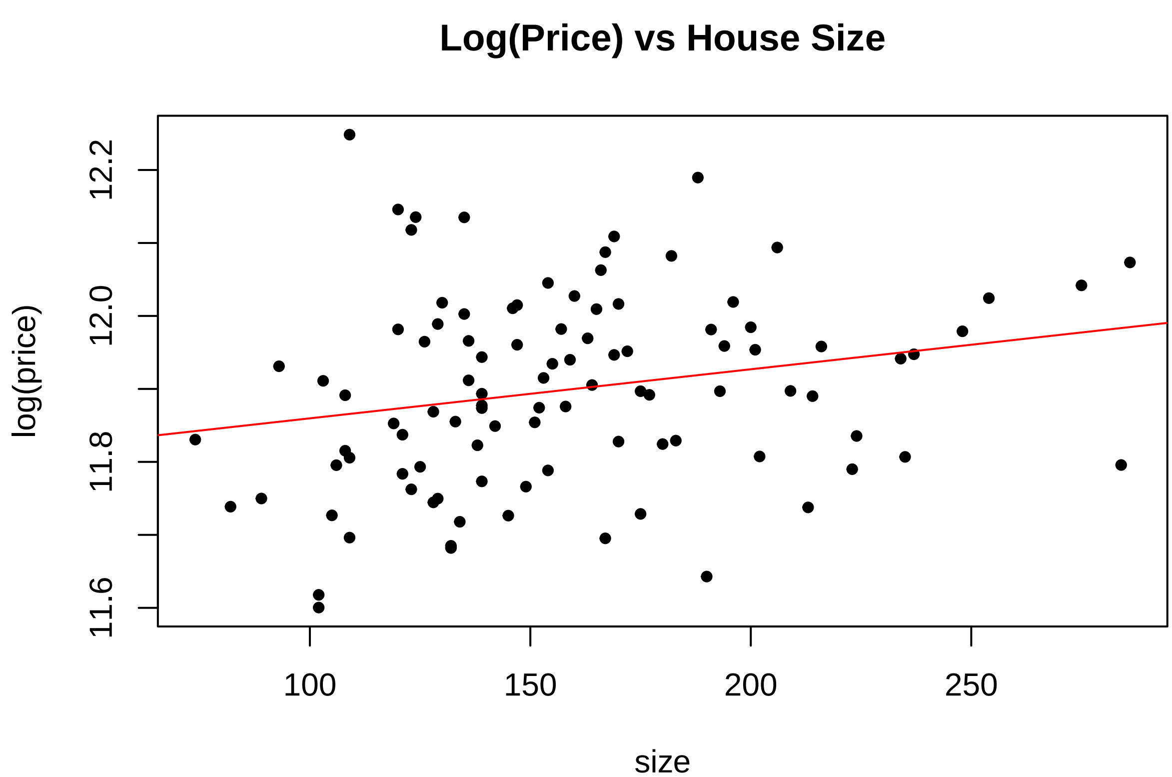
```
##
## Call:
## lm(formula = log(price) ~ size, data = housing)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.27734 -0.09326 -0.00847  0.08242  0.38265
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) 1.179e+01  4.837e-02 243.821   <2e-16 ***
## size        6.722e-04  2.944e-04   2.283   0.0246 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1297 on 98 degrees of freedom
## Multiple R-squared:  0.05051,    Adjusted R-squared:  0.04082
## F-statistic: 5.213 on 1 and 98 DF,  p-value: 0.02457
```

```
cat("===  Log SLR Model Plot  === \n")
```

```
## ===  Log SLR Model Plot  ===
```

```
plot(log(price) ~ size, data = housing, main = "Log(Price) vs House Size", pch=20) +
  # lines(loess.smooth(lm_lg$fitted.values, housing$size),col="red") +
  abline(lm_lg, col = "red")
```




```
## integer(0)
```

```
# 3.Transforming the Predictor
lm_lgh <- lm(price ~ log(size), data = housing)
summary(lm_lgh)
```

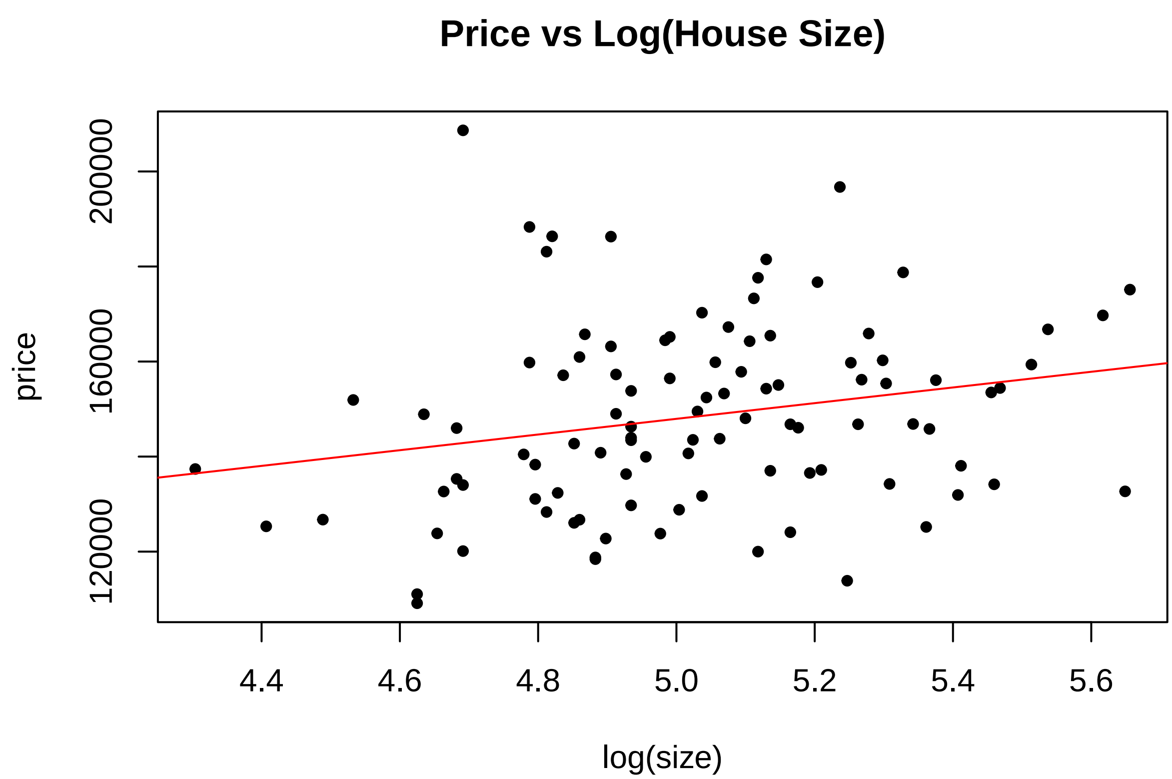
```
##
```

```
## Call:
## lm(formula = price ~ log(size), data = housing)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -38144 -13665  -1748  11615  65798
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    65431      35856   1.825   0.0711 .
## log(size)      16503       7122   2.317   0.0226 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19410 on 98 degrees of freedom
## Multiple R-squared:  0.05194,    Adjusted R-squared:  0.04226
## F-statistic: 5.369 on 1 and 98 DF,  p-value: 0.02258
```

```
cat("===  Log SLR Model Plot  === \n")
```

```
## ===  Log SLR Model Plot  ===
```

```
plot(price ~ log(size), data = housing, main = "Price vs Log(House Size)", pch=20) +
  # lines(loess.smooth(lm_lg$fitted.values, housing$size),col="red") +
  abline(lm_lgh, col = "red")
```



```
## integer(0)
```

```r
cat("Evaluating Model Fit Improvements \n")
```

```
## Evaluating Model Fit Improvements
```

Model Fit of the Log-log Model:

- The **multiple R-squared** (0.06088) and **adjusted R-squared** (0.05129) of the log-log model indicate that only 6.088% of the variability in house prices is explained by house size. This model is also very weak in explanatory power.

Model Fit of the Logged Response Model:

- The **multiple R-squared** (0.05051) and **adjusted R-squared** (0.04082) of the logged response model indicate that only 5.051% of the variability in house prices is explained by house size. This is also very weak model.

Model Fit of the Logged Predictor Model:

- The **multiple R-squared** (0.05194) and **adjusted R-squared** (0.04226) of this model suggest that only 5.194% of the variability in house prices can be explained by house size.

Are any of the models useful?

The residual diagnostics of all four models indicate no assumption violation. But all of them have very low model fits (explanatory power), i.e; 0.04313 (on the original scale), 0.06088 (for the log-log model), 0.05051 (for the logged response model) and 0.05194 (logged predictor model) , and thus **neither of them is useful**.

SHC 798 Assignment 1, 2025

Richard Lubega

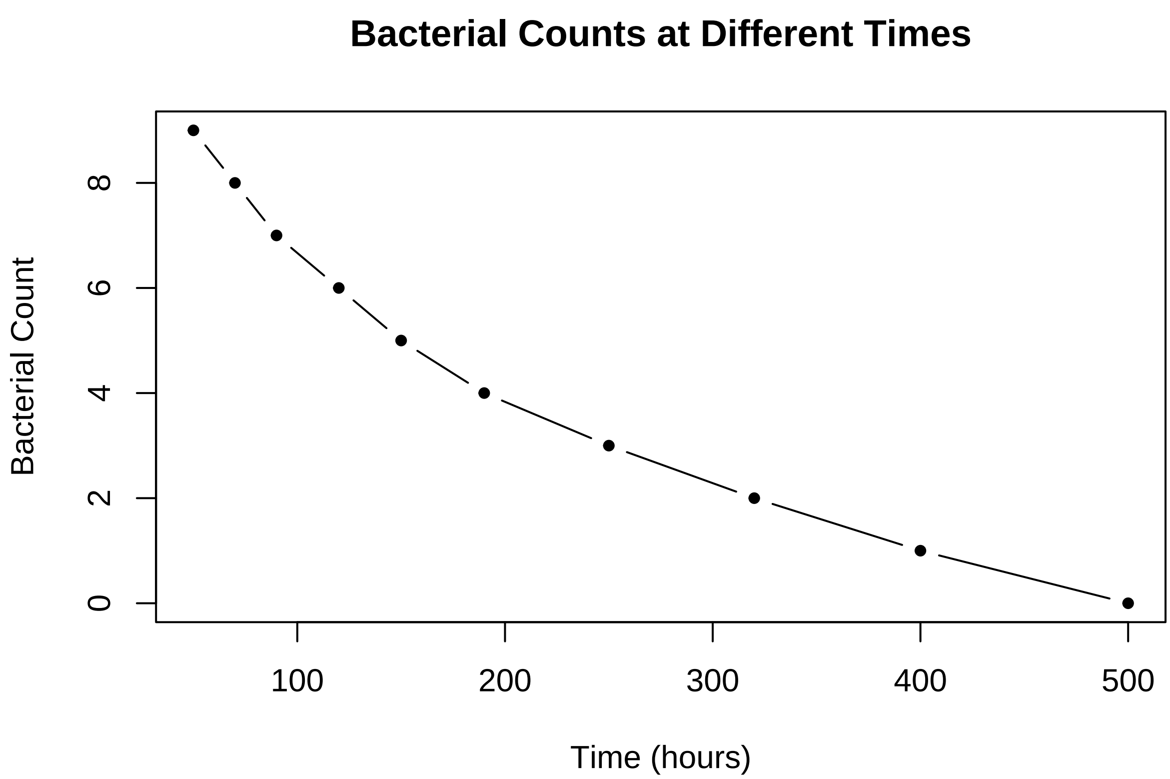
2025-07-14

SHC 798 Assignment 1, 2025

Part 3: Simple regression

Question 3

```
time <- 0:9
count <- c(500, 400, 320, 250, 190, 150, 120, 90, 70, 50)
decay <- data.frame(time, count)

plot(decay$count, decay$time, type = "b", main = "Bacterial Counts at Different Times", pch=20, xlab =
```



(a)

(b) Fit an exponential decay model and determine if this function better explains the data than a simple linear model.

1. **Simple Linear Model**

```r
# Simple Linear Model
lm_decay <- lm(count ~ time, data = decay)
summary(lm_decay)
```

```
##
## Call:
## lm(formula = count ~ time, data = decay)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -48.06 -32.59  -9.00  22.71  69.45
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)             24.935   17.27 1.29e-07 ***
## time         -48.121      4.671  -10.30 6.79e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 42.42 on 8 degrees of freedom
## Multiple R-squared:  0.9299, Adjusted R-squared:  0.9212
## F-statistic: 106.1 on 1 and 8 DF,  p-value: 6.792e-06
```

2. **Exponential Model**

- The exponential Model is obtained from a logged response model.
- From general decay models, $\mathrm{C(t)} = \mathrm{C_0} \,.\, \mathrm{e}^{-\mathrm{k}\,\cdot\,\mathrm{t}}$, where C implies bacterial count and t, time (hours).
- We linearise to $\log[\mathrm{C(t)}] = \log[\mathrm{C_0}] - \mathrm{k} \,.\, \mathrm{t}$, which is generally written as
  - $\log(\text{count}) = \beta_0 + \beta_1 \,.\, \text{time} + E_i$ ………… a logged response model

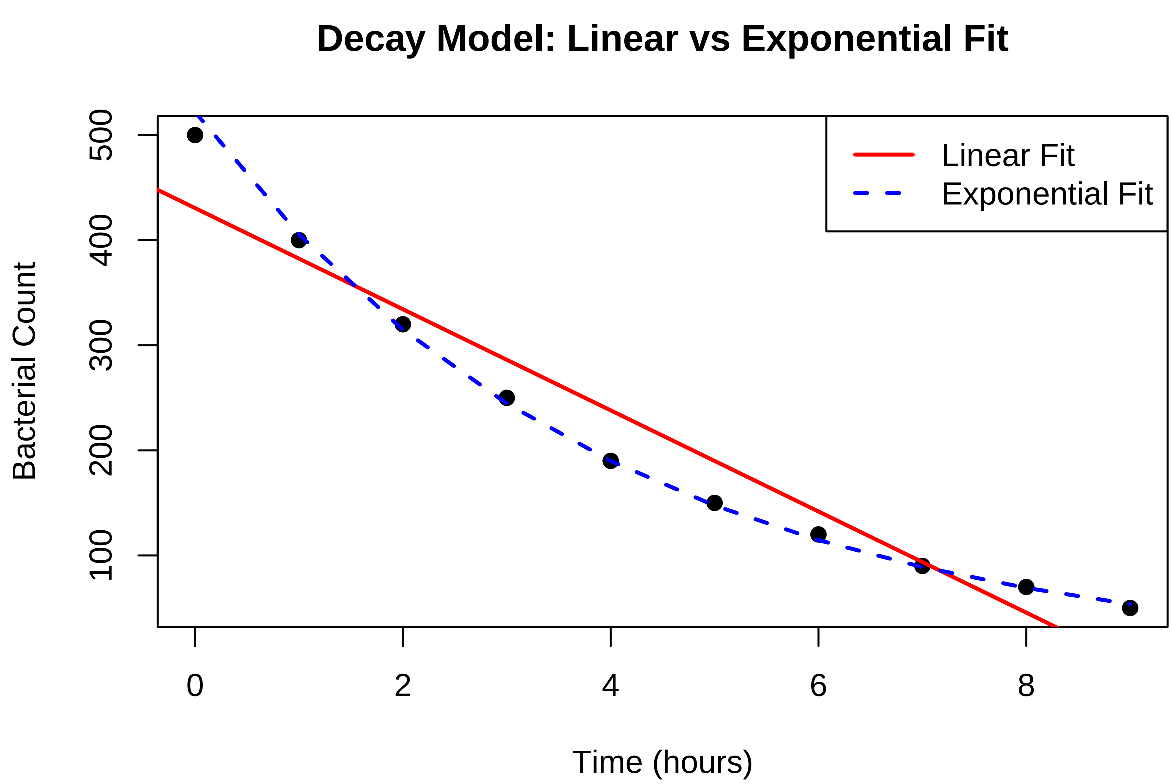
```r
exp_decay <- lm(log(count) ~ time, data = decay)
summary(exp_decay)
```

```
##
## Call:
## lm(formula = log(count) ~ time, data = decay)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.070704 -0.009861  0.012290  0.016734  0.046494
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)            0.021321  293.49  < 2e-16 ***
## time        -0.252757   0.003994  -63.29 4.32e-12 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03628 on 8 degrees of freedom
## Multiple R-squared:  0.998,  Adjusted R-squared:  0.9978
## F-statistic:  4005 on 1 and 8 DF,  p-value: 4.321e-12
```

Visualising the models

```r
plot(time, count, pch = 19, col = "black",
     xlab = "Time (hours)", ylab = "Bacterial Count",
     main = "Decay Model: Linear vs Exponential Fit")
abline(lm_decay, col = "red", lwd = 2)
# abline(exp_decay, col = "red", lwd = 2)

lines(time, exp(predict(exp_decay)), col = "blue", lwd = 2, lty = 2)
legend("topright", legend = c("Linear Fit", "Exponential Fit"),
       col = c("red", "blue"), lwd = 2, lty = c(1, 2))
```



Comparing Explanatory Power:

The exponential model *fits* (explains) the data **better** because it does have a higher R-Squared value (0.998) compared to the simple linear model (with 0.9299).

(c) Predict the bacterial count at time = 10 hours

```r
pred_c <- predict(exp_decay, newdata = data.frame(time = 10))
pred_10 <- exp(pred_c)
cat("the bacterial count at time = 10 hours is:", pred_10, "\n")
```

```
## the bacterial count at time = 10 hours is: 41.67785
```

(d) Compute a 95% confidence interval for the estimated decay rate

```r
# The decay rate is the slope
cat("the 95% confidence interval for the estimated decay rate is: \n")
```

```
## the 95% confidence interval for the estimated decay rate is:
```

```r
confint(exp_decay, "time")
```

```
##           2.5 %     97.5 %
## time -0.2619668 -0.2435471
```

PART 3: QUESTION 4

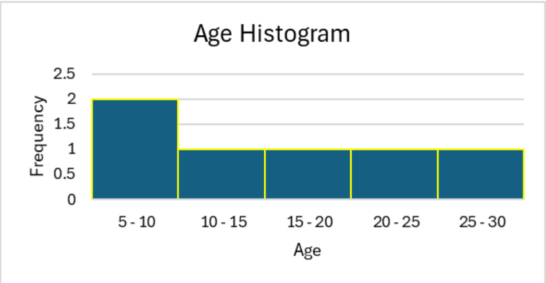
(a-1) Draw histograms (by hand) for both Age and Strength

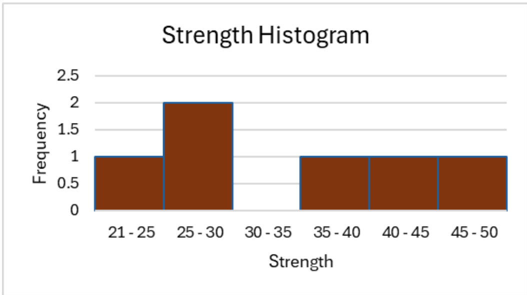
Computation Tables (Grouping)

Age groups	frequency
5 - 10	2
10 - 15	1
15 - 20	1
20 - 25	1
25 - 30	1

Strength groups	frequency
21 - 25	1
25 - 30	2
30 - 35	0
35 - 40	1
40 - 45	1
45 - 50	1

Histograms





(a-2) Comment on the shape of each distribution.

Both distributions can be described as right skewed (with more data to the left)

(b-1) Apply a natural log transformation

Computation Tables (Applying Natural Logarithms, ln)

Original Alien Beam Data		**Log-transformed Data**	
Age, x (years)	**Strength, y (MPa)**	ln(Age), x'	ln(Strength), y'
5	48	1.6094	3.8712
10	42	2.3026	3.7377
15	37	2.7081	3.6109
20	30	2.9957	3.4012
25	27	3.2189	3.2958
30	21	3.4012	3.0445

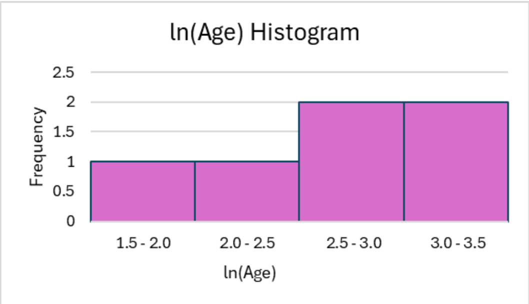
(b-1) Re-draw histograms for the transformed variables

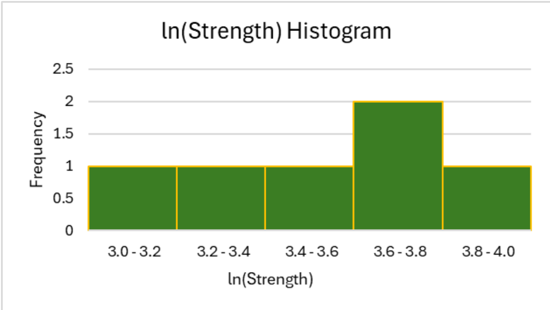
Computation Tables (Grouping)

ln(Age) groups	frequency
1.5 - 2.0	1
2.0 - 2.5	1
2.5 - 3.0	2
3.0 - 3.5	2

ln(Strength) groups	frequency
3.0 - 3.2	1
3.2 - 3.4	1
3.4 - 3.6	1
3.6 - 3.8	2
3.8 - 4.0	1

Histograms





(c) Compute the regression coefficients β0 and β1 for the log-log model using the least squares method

According to the least squares paradigm, the best fitting regression line is obtained with optimal coefficients, i.e.:

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \quad \text{and} \quad \hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

For β_1:

	Computations on Log-transformed Data					
	ln(Age), x'	**ln(Strength), y'**	**x'-x***	**(x'-x*)^2**	**y'-y***	**(x'-x*) . (y'-y*)**
	1.6094	3.8712	-1.0965	1.20240	0.3776	-0.41410
	2.3026	3.7377	-0.4034	0.16273	0.2441	-0.09847
	2.7081	3.6109	0.0021	0.00000	0.1174	0.00024
	2.9957	3.4012	0.2898	0.08396	-0.0924	-0.02676
	3.2189	3.2958	0.5129	0.26306	-0.1977	-0.10141
	3.4012	3.0445	0.6952	0.48333	-0.4490	-0.31218
Summation	**16.23588**	**20.96135**		2.19548		-0.95268
Mean (x* & y*)	x* = 16.23588/6	y* = 20.96135/6				
Mean (x* & y*)	**2.7060**	**3.4936**		**β1**	-0.43393	

For β_0:

From $\hat{\beta}_0 = \bar{y} - \hat{\beta}_1\bar{x}$

β0	4.66776

(d) Determine the R^2 value and interpret its meaning

The formula below is used:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \in [0,1]$$

Computing Predicted values [ln(Strength), y^]				
ln(Strength), y^ is computed using the regression coefficients, β0 and β1				
ln(Strength), y'	**ln(Strength), y^**	**Residuals (y' - y^)**	**RSS =(y' - y^)^2**	**(y'-y*)^2**
3.8712	3.96938	-0.09818	0.00964	0.14261
3.7377	3.66860	0.06907	0.00477	0.05959
3.6109	3.49266	0.11826	0.01399	0.01377
3.4012	3.36783	0.03337	0.00111	0.00853
3.2958	3.27100	0.02484	0.00062	0.03909
3.0445	3.19188	-0.14736	0.02172	0.20163
Summation			0.05184	0.46524

R^2	1 - [sum((y' - y^)^2)/sum((y'-y*)^2)]
R^2	0.88857

(e) Compute the p-value for β_1 and explain whether Age significantly affects Strength.

Testing the Null Hypothesis

$$T_{H_0:\beta_1=0} = \frac{\hat{\beta}_1}{\hat{\sigma}_{\hat{\beta}_1}}$$

Computing the P-value for β1		
n (data points)		6
df (degrees of freedom)	= (n-2)	4
Residual Sum of Squares (RSS) (errors^2) = sum(y' - y^)^2	= Sum(y' - y^)^2	0.05184
Residual Standard Error (RSE)	= RSS/df	**0.11384**
Computing the t-statistic (using the β1 = 0 null hypothesis)		
β1 (Slope Coefficient)		-0.43393
[(x'-x*)^2]^0.5		1.48172
SE(β1) : Standard error		0.07683
t-statistic for β1		-5.64781
Two-tailed p-value; p = 2*P(T>|t|)	(From Excel)	0.004841
Two-tailed p-value; p = 2*P(T>|t|)	(From Tables) = 1-0.995	0.005

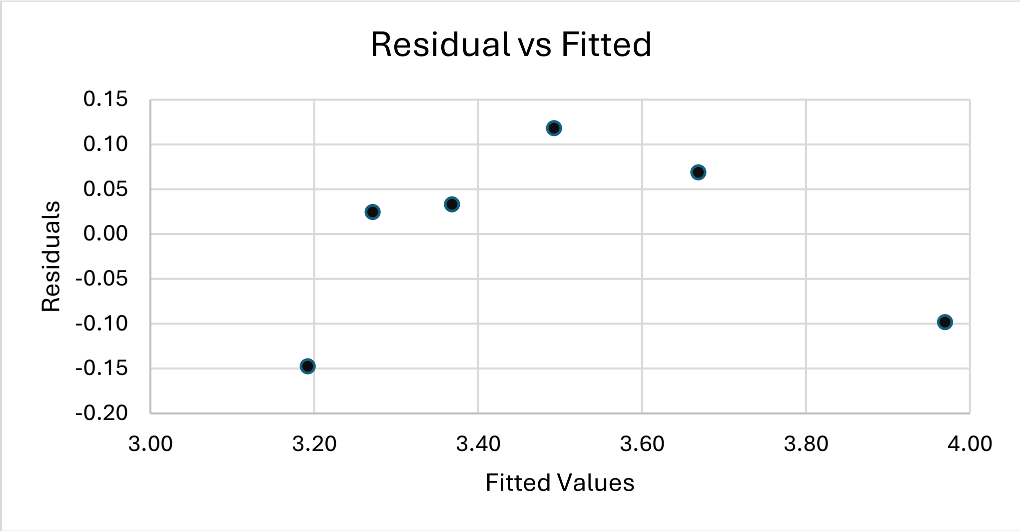
The p-value for $\boldsymbol{\beta_1}$ **= 0.005**. We reject the null hypothesis at the 5% significance level. This means that age has a statistically significant effect on the bending strength of alien beams, and we can be fairly confident (with 95% confidence) that the relationship isn't due to chance.

(f-1) Compute the residuals for the transformed model. Manually plot the residuals against the fitted values.

Computation of the residuals

Residuals vs Fitted	
Residuals (y' - y^)	**Fitted [ln(Strength), y^]**
-0.09818	3.96938
0.06907	3.66860
0.11826	3.49266
0.03337	3.36783
0.02484	3.27100
-0.14736	3.19188

(f-2) Manually plot the residuals against the fitted values.



VALIDATE YOUR HAND CALCULATIONS IN R.

------ (Code and Output shown in the following section) ----

SHC 798 Assignment 1, 2025

Richard Lubega

2025-07-14

SHC 798 Assignment 1, 2025

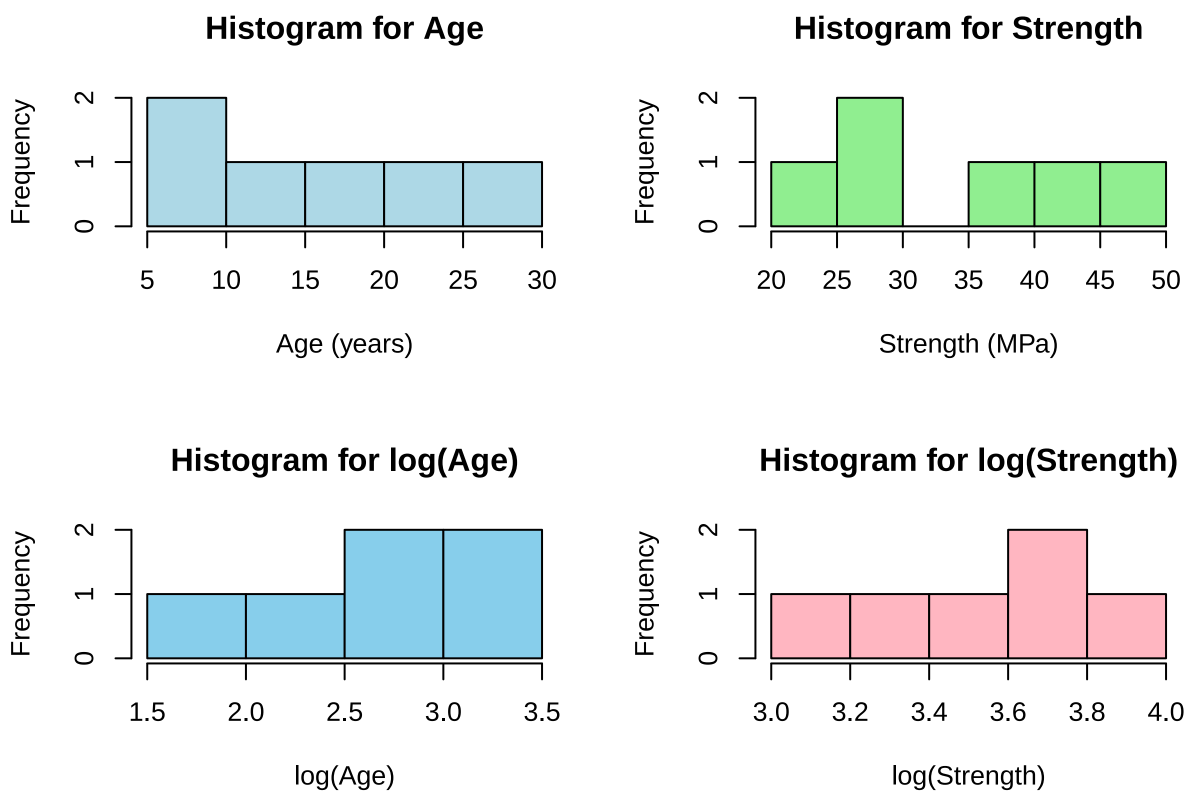
Part 3: Simple regression

Question 4

Validating the Hand Calculations

```
age <- c(5, 10, 15, 20, 25, 30)
strength <- c(48, 42, 37, 30, 27, 21)
a_beams <- data.frame(age, strength)
```

```
par(mfrow = c(2,2))
hist(a_beams$age, main = "Histogram for Age", xlab = "Age (years)",
     col = "lightblue")
hist(a_beams$strength, main = "Histogram for Strength", xlab = "Strength (MPa)",
     col = "lightgreen")

#Applying natural logarithms
a.age <- log(a_beams$age)
a.strength <- log(a_beams$strength)
hist(a.age, main = "Histogram for log(Age)", xlab = "log(Age)",
     col = "skyblue")
hist(a.strength, main = "Histogram for log(Strength)", xlab = "log(Strength)",
     col = "lightpink")
```



(a) and (b)

```r
par(mfrow = c(1,1))
```

```r
log_beam <- lm(log(strength) ~ log(age), data = a_beams)
summary(log_beam)
```

(c)

```
##
## Call:
## lm(formula = log(strength) ~ log(age), data = a_beams)
##
## Residuals:
##        1        2        3        4        5        6
## -0.09818  0.06907  0.11826  0.03337  0.02484 -0.14736
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.66776    0.21304  21.911 2.57e-05 ***
## log(age)    -0.43393    0.07683  -5.648  0.00484 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.1138 on 4 degrees of freedom
## Multiple R-squared:  0.8886, Adjusted R-squared:  0.8607
## F-statistic:  31.9 on 1 and 4 DF,  p-value: 0.004841
```

```r
summary(log_beam)$r.squared
```

(d)

```
## [1] 0.8885725
```

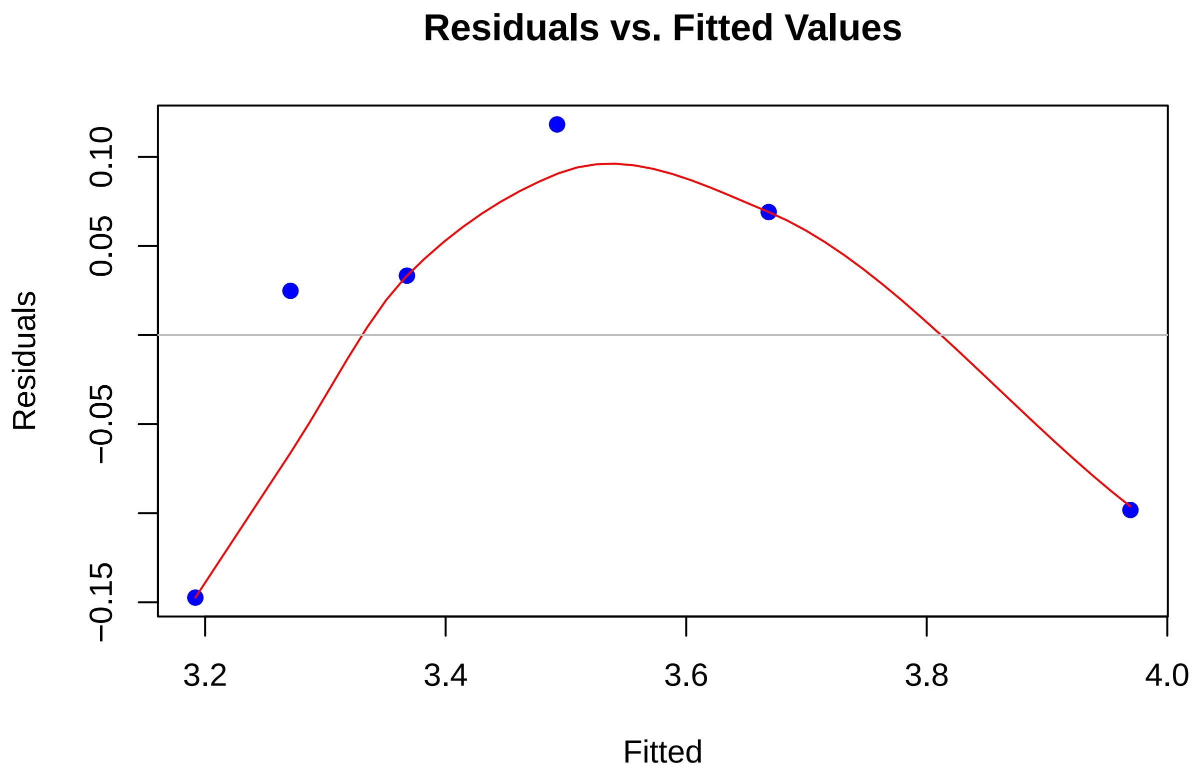
```r
summary(log_beam)$coefficients["log(age)", "Pr(>|t|)"]
```

(e)

```
## [1] 0.00484069
```

####(f)

```r
# Tukey-Anscombe Plot
plot(log_beam$fitted.values, log_beam$residuals, xlab="Fitted", ylab="Residuals", pch = 19, col = "blue
  title("Residuals vs. Fitted Values") +
  lines(loess.smooth(log_beam$fitted.values, log_beam$residuals),col="red") +
  abline(h=0, col="grey")
```


Residuals vs. Fitted Values
Residuals
0.10
0.05
−0.05
−0.15
3.2
3.4
3.6
3.8
4.0
Fitted

```
## integer(0)
```