498818_SDS496_HW3

2024-09-16

Homework Set 3 SDS 496

```r
# Load in all Data for Problem 1, 2 and 3

library(data.table)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::between()     masks data.table::between()
## x dplyr::filter()      masks stats::filter()
## x dplyr::first()       masks data.table::first()
## x lubridate::hour()    masks data.table::hour()
## x lubridate::isoweek() masks data.table::isoweek()
## x dplyr::lag()         masks stats::lag()
## x dplyr::last()        masks data.table::last()
## x lubridate::mday()    masks data.table::mday()
## x lubridate::minute()  masks data.table::minute()
## x lubridate::month()   masks data.table::month()
## x lubridate::quarter() masks data.table::quarter()
## x lubridate::second()  masks data.table::second()
## x purrr::transpose()   masks data.table::transpose()
## x lubridate::wday()    masks data.table::wday()
## x lubridate::week()    masks data.table::week()
## x lubridate::yday()    masks data.table::yday()
## x lubridate::year()    masks data.table::year()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(nlme)
```

```
##
## Attaching package: 'nlme'
##
## The following object is masked from 'package:dplyr':
##
##     collapse
```

```r
air_quality <- fread('/Users/aidanashrafi/Downloads/airquality.txt',sep= ' ')
air_quality2 <- fread('/Users/aidanashrafi/Downloads/airquality.txt',sep= ' ')
brake <- fread('/Users/aidanashrafi/Downloads/brake.txt',sep= ' ')
```

```r
seatpos <- fread('/Users/aidanashrafi/Downloads/seatpos.txt',sep= ' ')
```

Problem 1

Make sure the data was read in properly.

```r
head(air_quality)
```

```
##    Ozone Solar.R  Wind  Temp Month   Day
##    <int>   <int> <num> <int> <int> <int>
## 1:    41     190   7.4    67     5     1
## 2:    36     118   8.0    72     5     2
## 3:    12     149  12.6    74     5     3
## 4:    18     313  11.5    62     5     4
## 5:    NA      NA  14.3    56     5     5
## 6:    28      NA  14.9    66     5     6
```

```r
# Determine if there are any missing values in the data
sum(is.na(air_quality))
```

```
## [1] 44
```

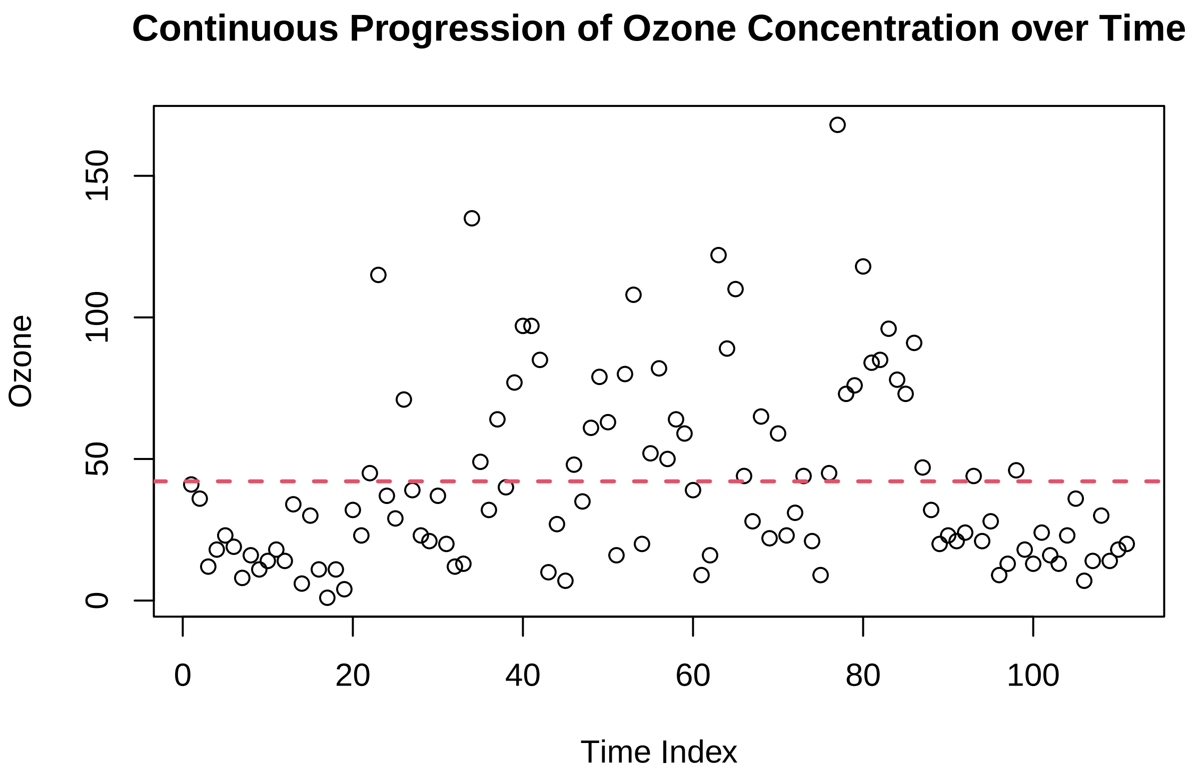
```r
# Remove missing values
air_quality <- drop_na(air_quality)
# Check to make sure NA is removed
sum(is.na(air_quality))
```

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

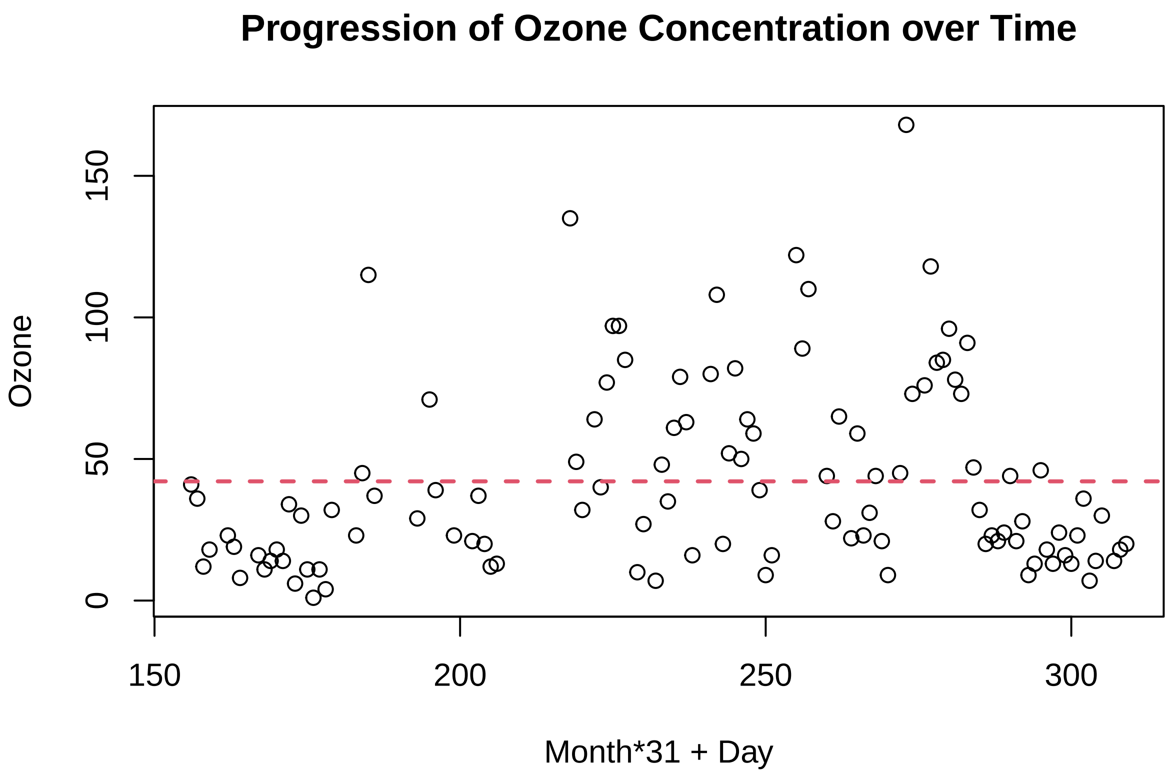
```r
air_quality2 <- drop_na(air_quality2)
```

i. Create plots to examine the progression of ozone concentration over time

```r
# Simple Plot
plot(air_quality$Ozone, xlab = 'Time Index', ylab = 'Ozone',main ='Continuous Progression of Ozone Conc
abline(h=mean(air_quality$Ozone), col = 2 ,lwd=2, lty=2)
```



```
# Create a variable for Date with no overlap
# Combine Month and Day into Date to examine progression
air_quality$Date <- as.numeric(air_quality$Month*31 + air_quality$Day)

# Plot the progression of ozone concentration over time
plot(Ozone ~ Date, air_quality, xlab="Month*31 + Day", main ='Progression of Ozone Concentration over Ti
# Create a line showing the average for better visualization
abline(h=mean(air_quality$Ozone), col=2, lwd=2, lty=2)
```



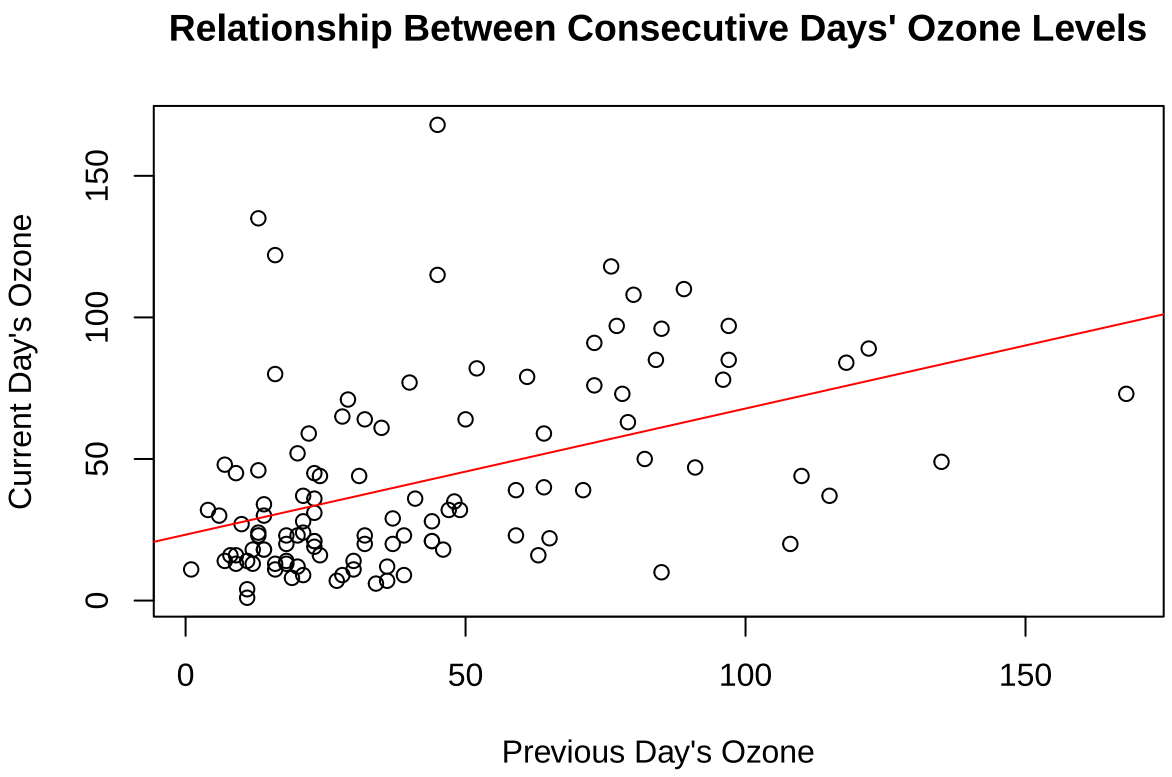
The progression of ozone concentration varies significantly over the time period covered. Initially, there is a tight cluster of values around the lower range, but as time progresses, the spread increases, suggesting more variability in ozone levels. While not strictly increasing or decreasing linearly, the plot hints at a potential seasonal trend where ozone concentrations might increase up to a certain midpoint and then decrease. Furthermore, there seems to be a peak around the middle of the data set.

i. Create plots to examine the relationship between ozone concentration on consecutive days

```
# Create a lagged variable, representing the previous observation in the dataset
air_quality[, lagged_Ozone := shift(Ozone,1,type = 'lag')]
# Check to make sure lagged variable was created
head(air_quality)
```

```
##    Ozone Solar.R  Wind  Temp Month   Day  Date lagged_Ozone
##    <int>   <int> <num> <int> <int> <int> <num>        <int>
## 1:    41     190   7.4    67     5     1   156           NA
## 2:    36     118   8.0    72     5     2   157           41
## 3:    12     149  12.6    74     5     3   158           36
## 4:    18     313  11.5    62     5     4   159           12
## 5:    23     299   8.6    65     5     7   162           18
## 6:    19      99  13.8    59     5     8   163           23
```

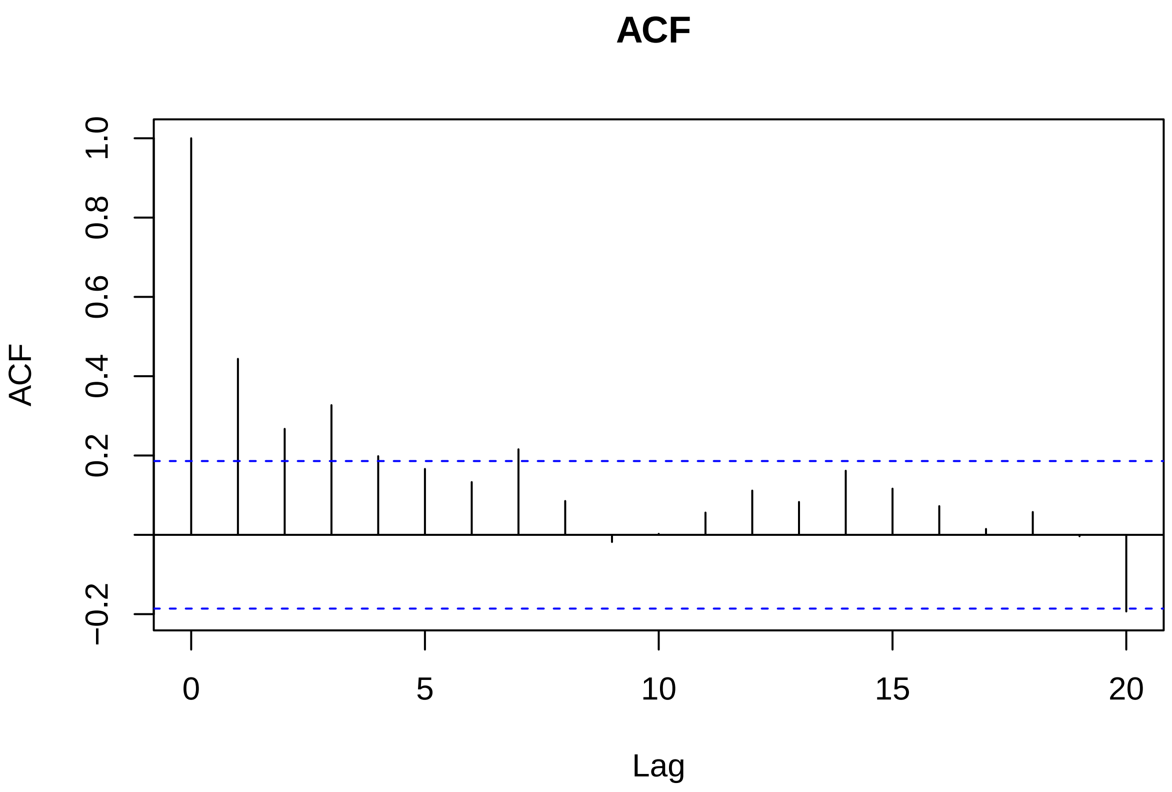
```
# Plotting the current day's ozone levels against the previous day's levels
plot(air_quality$lagged_Ozone, air_quality$Ozone, main = "Relationship Between Consecutive Days' Ozone 
     xlab = "Previous Day's Ozone", ylab = "Current Day's Ozone")


# Add a linear regression line
abline(lm(Ozone ~ lagged_Ozone, data = air_quality), col = "red")
```



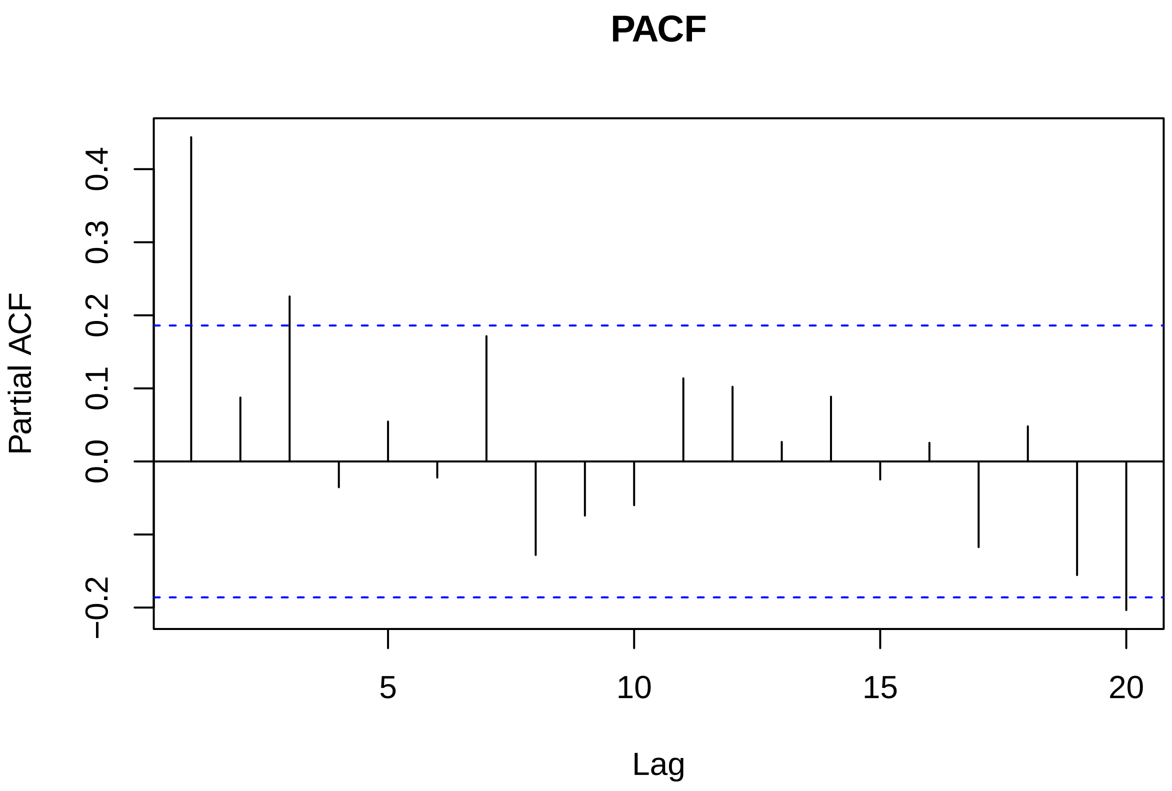
The relationship between consecutive days' ozone levels is positive. It is a relatively strong positive linear relationship, as we can see from the red line, which is a fitted regression of the lagged observation as the predictor and the current observation as the response.

i. Create plots to examine the autocorrelation between ozone concentration on consecutive days

```r
# Autocorrelation between ozone concentration on consecutive days
acf(air_quality2$Ozone, main = "ACF ")
```



```r
pacf(air_quality2$Ozone, main = 'PACF')
```



The autocorrelation function (ACF), provides insight into the linear relationship between time series observations at different time lags. In this case, the ACF plot shows a significant correlation for the first lag, indicating a strong dependency of each Ozone observation on its immediate predecessor with a correlation coefficient of approximately 0.4. This suggests that the ozone level on a given day is strongly influenced by the level on the previous day.

Furthermore, the ACF reveals that there is a notable correlation at the second lag (about 0.25), suggesting that the ozone levels also retain a memory of the conditions two days prior. This decay in correlation as the lags increase is typical for time series data where the influence of past values gradually diminishes over time.

Refresher on Time Series (From Personal Understanding, and for use for Future Reference)

Please comment and let me know if I am misunderstanding any of these concepts, although I know we are not going to cover them in class.

ACF: Measures the correlation between observations of a time series separated by k time units (lags). Expresses the degree of linear dependency between an observation at time t and the observations at previous times $t-k$.

PACF: Measures the correlation between observations at two points in time, ignoring the influence of the observations in between. Expresses the direct effect of past data points on the future data point, without the influence of intermediary data points.

Stationarity Requirement: Before analyzing ACF and PACF, ensure the data is stationary. Stationary data has a constant mean, variance, and autocorrelation structure that does not depend on time. Non stationary data can often be made stationary through trend differencing and, if necessary, accounting for seasonality.

Statistical Tests for Stationarity: Use statistical tests such as the Augmented Dickey Fuller (ADF) test to check for stationarity. If the test indicates non stationarity, consider applying transformations like differencing.

Model Identification:

MA(q) Indicators: If PACF decays to 0, and ACF cuts off at lag q, consider a Moving Average process of order q.

AR(p) Indicators: If PACF cuts off at lag p and ACF decays to 0, consider an Auto regressive process of order p.

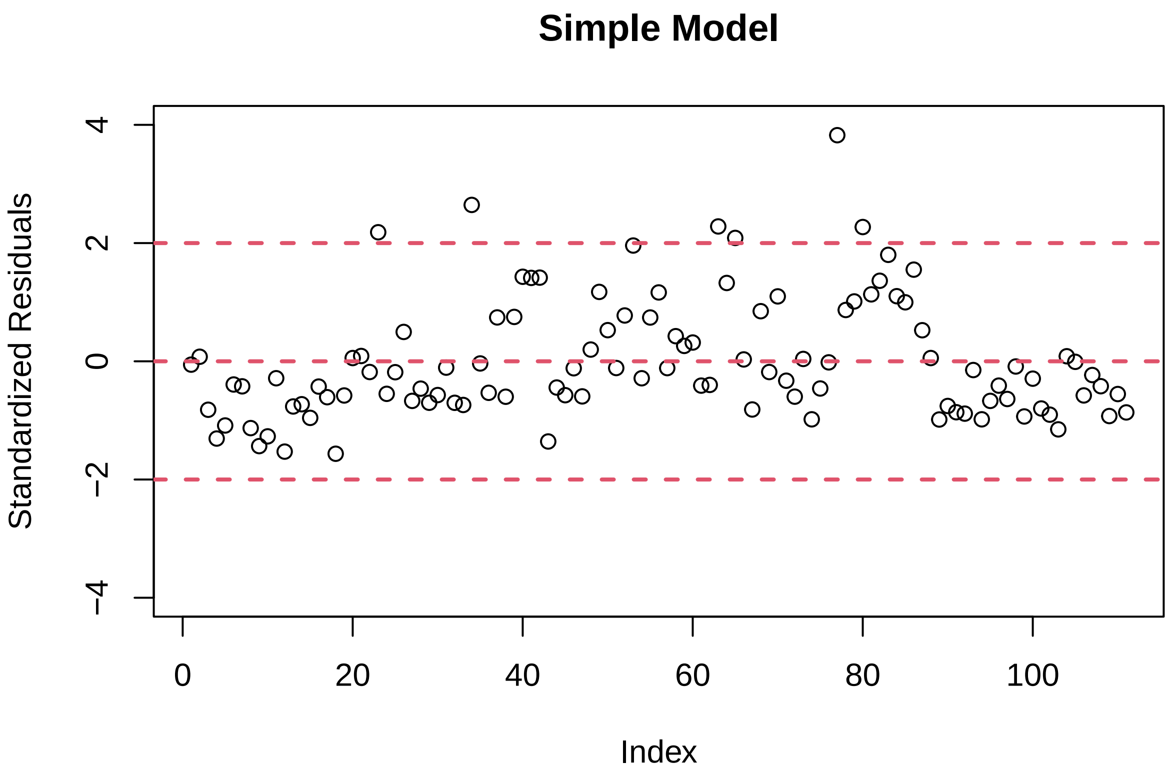
ARMA(p, q) Indicators: Slow decay in both ACF and PACF suggests a Mixed ARMA model.

ARIMA(p, d, q) Indicators: If the data required differencing to become stationary and shows damped oscillations in ACF and PACF, consider an ARIMA model.

This framework helps identify the underlying patterns in time series data, facilitating appropriate model selection for analysis and forecasting.

ii. Fit a suitable simple linear model and create the same plots for the standardized residuals of the fitted linear model. Are your findings consistent with those from the previous part?

```r
# Fit a suitable simple linear model
simple <- lm(Ozone ~ Solar.R, data = air_quality)

library(dplyr)
plot(rstandard(simple), ylab='Standardized Residuals', main='Simple Model', ylim=c(-4,4))
abline(h=0,col=2,lty=2,lwd=2)
abline(h=2,col=2,lty=2,lwd=2)
abline(h=-2,col=2,lty=2,lwd=2)
```



Yes, my findings are consistent with the previous part. **The plot of standardized residuals reveals a pattern that deviates from randomness, which typically suggests model inadequacies. One potential issue is the presence of autocorrelation, as indicated by the high ACF value in ozone concentrations on consecutive days. This autocorrelation may cause the residuals to exhibit systematic patterns, as evident from clusters of residuals that do not fluctuate symmetrically around the zero line.**

Such patterns in the residuals indicate that the model fails to capture some of the systematic variation in the data, potentially leading to biased or inefficient estimates. The non random distribution of residuals suggests that the underlying assumptions of independence and identical distribution of errors are violated, which can compromise the reliability of the model's inference and predictive accuracy.

This is especially problematic as it might mask the true variability of the model errors, leading to underestimation of the standard errors and overly optimistic confidence intervals for the model parameters.

iii. Fit a linear model by using the generalized least squares method with autoregressive covariance structure. What's the value of the estimated correlation coefficient?

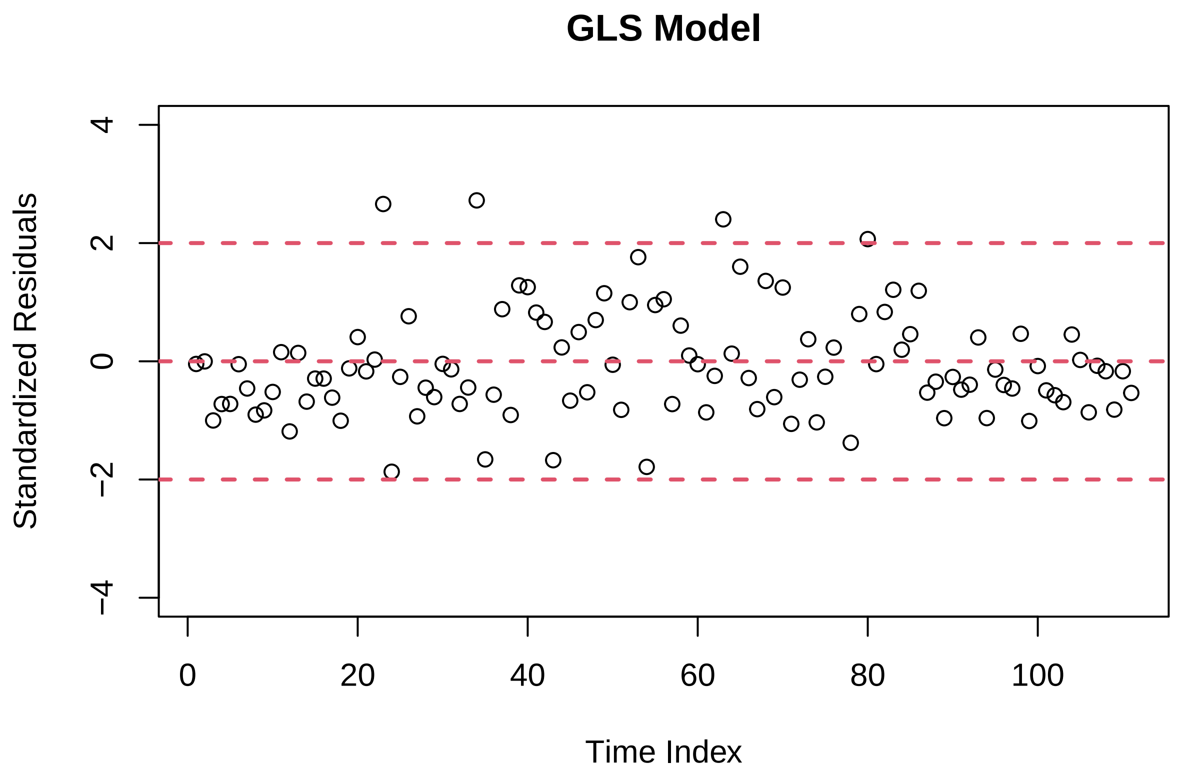
```
# Fit a linear model using GLS with autoregressive covariance structure
gls <- gls(Ozone ~ Solar.R, data=air_quality, correlation = corAR1(form = ~ Date))

# Find value of estimated correlation coefficient
summary(gls)
```

```
## Generalized least squares fit by REML
##   Model: Ozone ~ Solar.R
##   Data: air_quality
##        AIC      BIC    logLik
##   1056.746 1067.512 -524.3731
##
## Correlation Structure: ARMA(1,0)
##  Formula: ~Date
##  Parameter estimate(s):
##      Phi1
## 0.5378207
##
## Coefficients:
##                 Value Std.Error  t-value p-value
## (Intercept) 27.620424  6.970653 3.962387  0.0001
## Solar.R      0.077941  0.026644 2.925286  0.0042
##
##  Correlation:
##         (Intr)
## Solar.R -0.704
##
## Standardized residuals:
##        Min         Q1        Med         Q3        Max
## -1.3091050 -0.7069387 -0.3895044  0.5704416  3.8373787
##
## Residual standard error: 31.74812
## Degrees of freedom: 111 total; 109 residual
```

The value of the estimated correlation coefficient is **-0.704**. The value of the estimate of beta hat for Solar.R is **0.077941.**

iii. Create the same plots for the normalized residuals of the fitted GLS model. Comment on whether the GLS method has alleviated any linear regression violations you observed in the previous part.

```
# Create the same plots for the normalized residuals of the fitted GLS model.
plot(residuals(gls,'normalized'), ylab='Standardized Residuals', main='GLS Model', xlab='Time Index', yl

# Consider bounds based on residuals to make comparisons later
abline(h=0, col=2, lwd=2, lty=2)
abline(h=2,col=2,lty=2,lwd=2)
abline(h=-2,col=2,lty=2,lwd=2)
```



Yes, GLS has effectively addressed the linear regression violation related to the randomness of the standardized residuals, a fundamental requirement for robust linear regression analysis. The distribution of standardized residuals now shows improved alignment around the zero line, suggesting that the model's assumptions are better met.

Notably, there are fewer residuals exceeding the bounds $(-2, 2)$, indicating reduced influence of outliers or extreme values and more stable variance across the data. This improvement enhances the reliability of the model's statistical tests and confidence in the regression coefficients.

iv. Include the mean ozone concentration on the previous day as an additional predictor in the ordinary linear regression model from part ii.

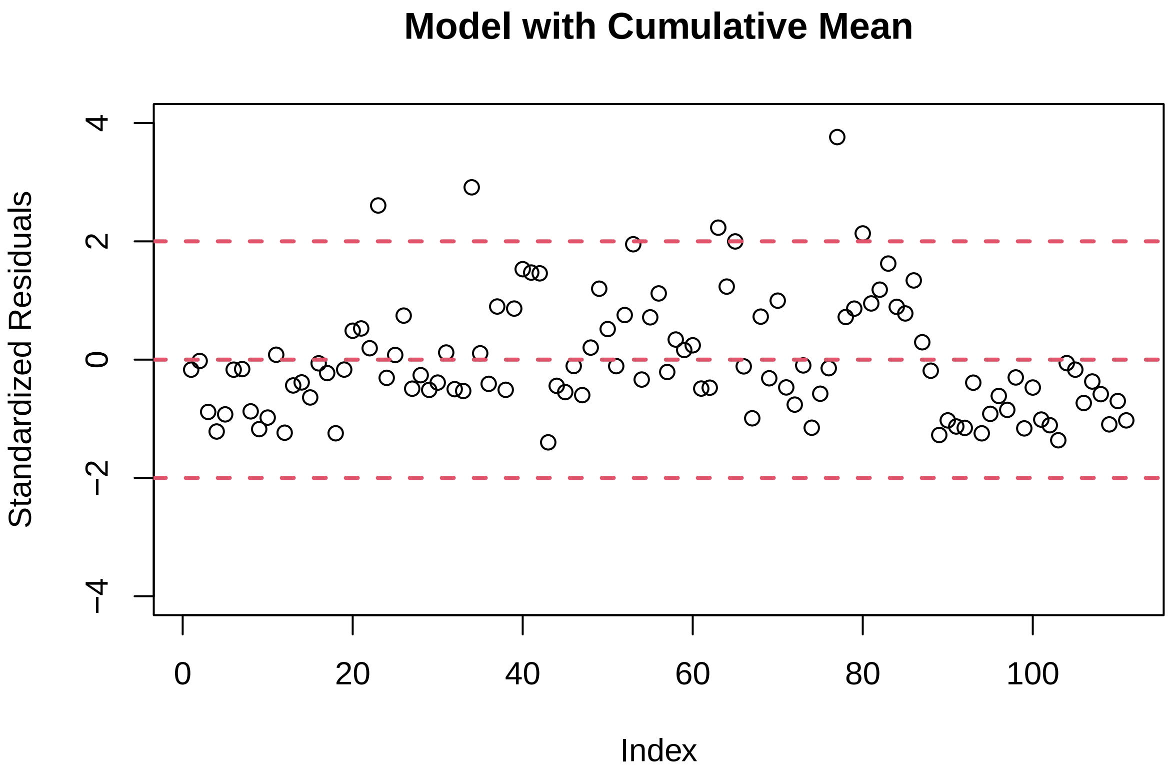
```
# Code to calculate mean ozone concentration using the previous day's mean as an additional predictor
setDT(air_quality)
air_quality[ , Date := as.numeric(Month * 31 + Day)]

air_quality <- as.data.frame(air_quality)

# Arrange by Date and calculate the cumulative mean correctly
air_quality <- air_quality %>%
  arrange(Date) %>%
  mutate(
    cumulative_mean_Ozone = cummean(Ozone),  # Calculate cumulative mean directly
    cumulative_mean_Ozone = ifelse(row_number() == 1, 41, lag(cumulative_mean_Ozone))
    # Adjust the first value as needed (this keeps it within mutate and correctly uses row_number)
  )

# Check the result to ensure it starts from the first observation
head(air_quality)
```

```
##   Ozone Solar.R Wind Temp Month Day Date lagged_Ozone cumulative_mean_Ozone
## 1    41     190  7.4   67     5   1  156           NA              41.00000
## 2    36     118  8.0   72     5   2  157           41              41.00000
## 3    12     149 12.6   74     5   3  158           36              38.50000
## 4    18     313 11.5   62     5   4  159           12              29.66667
## 5    23     299  8.6   65     5   7  162           18              26.75000
## 6    19      99 13.8   59     5   8  163           23              26.00000
```

```
# Create model with mean on the previous day as additional predictor
# I used a cumulative mean which calculated a new mean as every day progressed
cum_mean_model <- lm(Ozone ~ Solar.R + cumulative_mean_Ozone, data=air_quality)
```

iv. Create the same plots for the standardized residuals of the newly fitted linear model. Comment on whether any linear regression violations you observed in part ii have been ameliorated.

```
# Create plots for standardized residuals
plot(rstandard(cum_mean_model), ylab='Standardized Residuals', ylim=c(-4,4), main="Model with Cumulative
abline(h=0, col=2, lty=2,lwd=2)
abline(h=2, col=2, lty=2,lwd=2)
abline(h=-2, col=2, lty=2,lwd=2)
```



Yes, the distribution of the standardized residuals is no longer clustered and does not exhibit any patterns, showing randomness as desired. Furthermore, there are fewer residuals exceeding the bounds of $(-2, 2)$, indicating reduced influence of outliers or extreme values and more stable variance across the data. This improvement enhances the reliability of the model's statistical tests and confidence in the regression coefficients.

Problem 2

```r
# Make sure the data was read in properly
head(seatpos)
```

```
##       Age Weight HtShoes    Ht Seated   Arm Thigh   Leg hipcenter
##     <int>  <int>   <num> <num>  <num> <num> <num> <num>     <num>
## 1:     46    180   187.2 184.9   95.2  36.1  45.3  41.3  -206.300
## 2:     31    175   167.5 165.5   83.8  32.9  36.5  35.9  -178.210
## 3:     23    100   153.6 152.2   82.9  26.0  36.6  31.0   -71.673
## 4:     19    185   190.3 187.4   97.3  37.4  44.1  41.0  -257.720
## 5:     23    159   178.0 174.1   93.9  29.5  40.1  36.9  -173.230
## 6:     47    170   178.7 177.0   92.4  36.0  43.2  37.4  -185.150
```

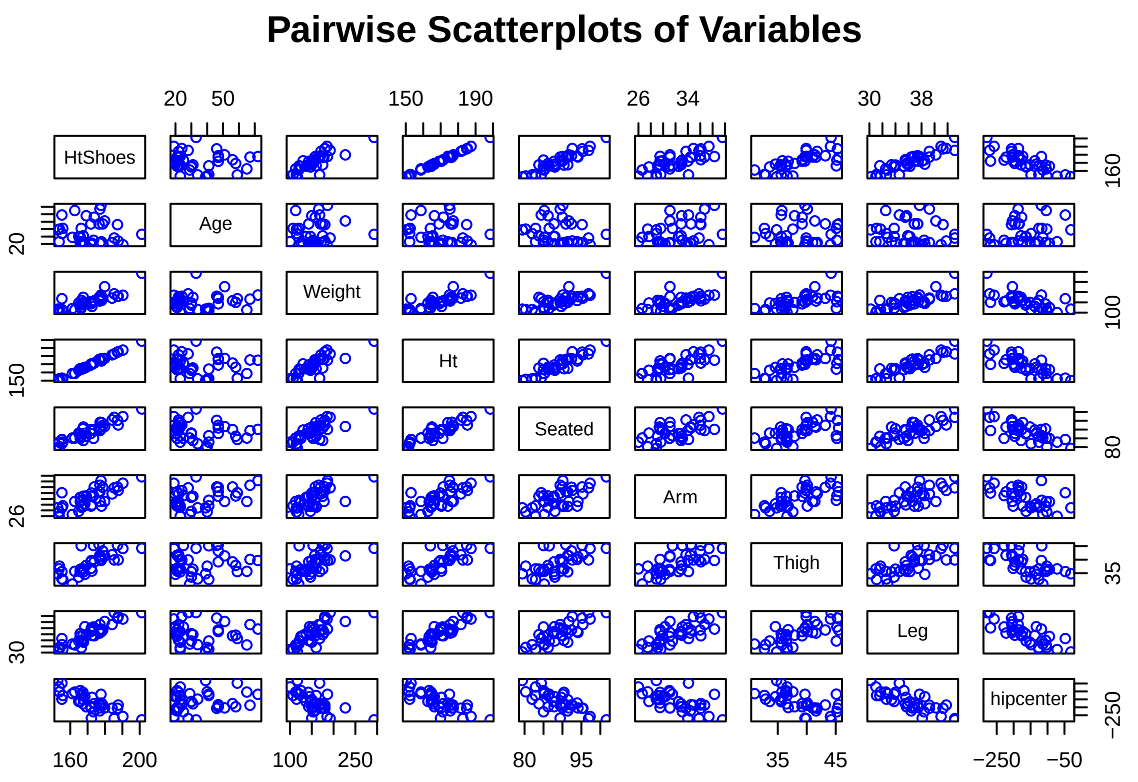
```r
# Determine if there are any missing values in the data
sum(is.na(seatpos))
```

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

```r
# No missing values
```

i. Create a pairwise scatterplot for each combination of 2 variables. Does any of the predictors appear to be particularly useful in making predictions about the response variable? Do any of the predictors appear to be strongly associated with each other?

```r
library(data.table)

variables <- seatpos[,.(HtShoes, Age, Weight,Ht,Seated,Arm,Thigh, Leg, hipcenter)]
# Create pairwise scatterplots for all combinations of quantitative variables
pairs(variables, main = "Pairwise Scatterplots of Variables", col = "blue")
```



Leg and HtShoes are highly negatively correlated with our predictor variable hipcenter. These predictors can be potentially useful to help make predictions about hipcenter. Ht and HtShoes are almost perfectly positively correlated, which is very rare to see, and is going to cause problems for our regression model because of multicollinearity. Furthermore, HtShoes and Seated are highly correlated , and HtShoes and Leg are highly positively correlated. This will be important to consider, when thinking about removal of variables to improve the model.

ii. Calculate a pairwise correlation coefficient for each combination of 2 variables. Is there any strong linear relationship between any of the predictors and the the response variable? Are there any strong linear relationships among the predictors?

```
# Calculate correlation matrix for another visualization of relationships between variables
cor(variables)
```

```
##               HtShoes         Age      Weight          Ht     Seated        Arm
## HtShoes    1.00000000 -0.07929694  0.82817733  0.99814750  0.9296751  0.7519530
## Age       -0.07929694  1.00000000  0.08068523 -0.09012812 -0.1702040  0.3595111
## Weight     0.82817733  0.08068523  1.00000000  0.82852568  0.7756271  0.6975524
## Ht         0.99814750 -0.09012812  0.82852568  1.00000000  0.9282281  0.7521416
## Seated     0.92967507 -0.17020403  0.77562705  0.92822805  1.0000000  0.6251964
## Arm        0.75195305  0.35951115  0.69755240  0.75214156  0.6251964  1.0000000
## Thigh      0.72486225  0.09128584  0.57261442  0.73496041  0.6070907  0.6710985
## Leg        0.90843341 -0.04233121  0.78425706  0.90975238  0.8119143  0.7538140
## hipcenter -0.79659640  0.20517217 -0.64033298 -0.79892742 -0.7312537 -0.5850950
##                 Thigh         Leg  hipcenter
## HtShoes    0.72486225  0.90843341 -0.7965964
## Age        0.09128584 -0.04233121  0.2051722
## Weight     0.57261442  0.78425706 -0.6403330
## Ht         0.73496041  0.90975238 -0.7989274
## Seated     0.60709067  0.81191429 -0.7312537
## Arm        0.67109849  0.75381405 -0.5850950
## Thigh      1.00000000  0.64954120 -0.5912015
## Leg        0.64954120  1.00000000 -0.7871685
## hipcenter -0.59120155 -0.78716850  1.0000000
```

Leg and HtShoes are highly negatively correlated with our predictor variable hipcenter (< negative 0.70). These predictors can be potentially useful to help make predictions about hipcenter. Ht and HtShoes are almost perfectly positively correlated (approximately 1.00), which is very rare to see, and is going to cause problems for our regression model because of multicollinearity. Furthermore, HtShoes and Seated is highly correlated (approximately 0.93), and HtShoes and Leg are highly positively correlated (approximately 0.91).

It is correct to see that all results from the scatterplots and the correlation matrix will agree, and therefore, we can arrive at the exact same conclusions from performing each task. Scatterplots give a more visual representation without being able to extract specific numbers, while the correlation matrix will give you the exact relationship that the scatterplot is showing.

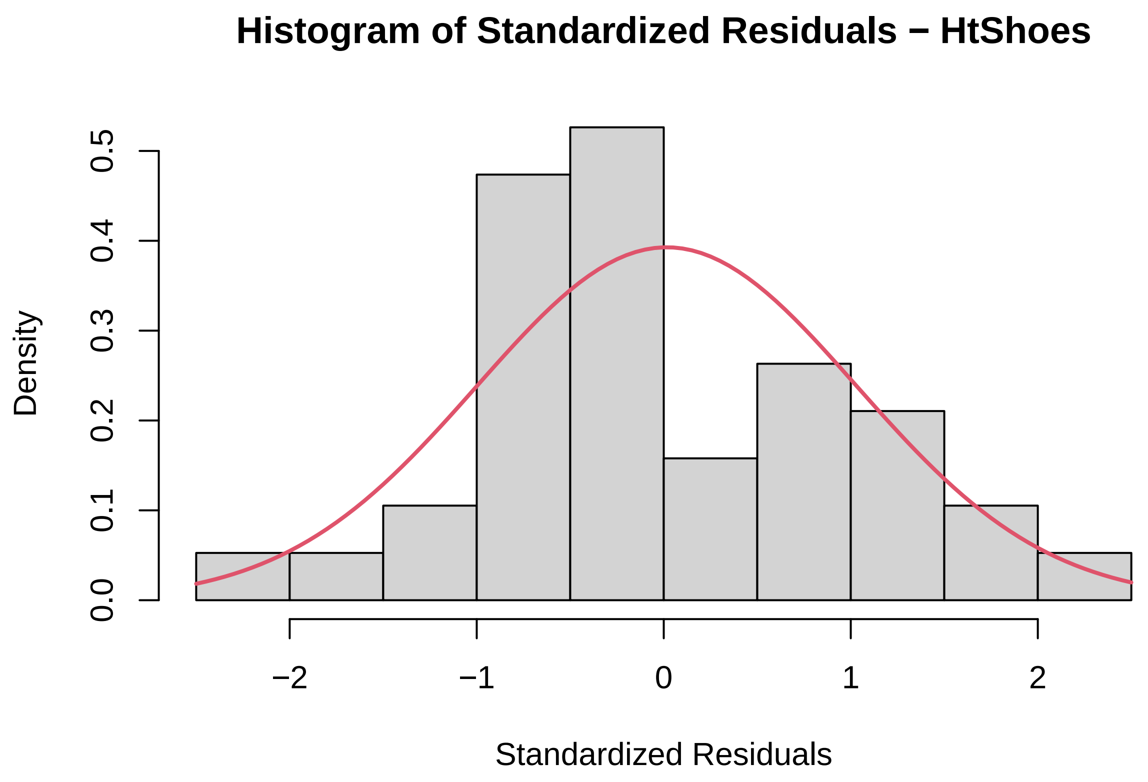
iii. Fit a multiple linear model with all available predictors. Comment on the estimated standard errors of your regression coefficients and the statistical significance of the relationship between each predictor and the response variable. How do those findings relate to your findings from the previous parts?

```
# Create a new model with all available predictors
all_var_model<- lm(hipcenter  ~ ., data=seatpos)
summary(all_var_model)
```

```
##
```

```
## Call:
## lm(formula = hipcenter ~ ., data = seatpos)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -73.827 -22.833  -3.678  25.017  62.337
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 436.43213  166.57162   2.620   0.0138 *
## Age           0.77572    0.57033   1.360   0.1843
## Weight        0.02631    0.33097   0.080   0.9372
## HtShoes      -2.69241    9.75304  -0.276   0.7845
## Ht            0.60134   10.12987   0.059   0.9531
## Seated        0.53375    3.76189   0.142   0.8882
## Arm          -1.32807    3.90020  -0.341   0.7359
## Thigh        -1.14312    2.66002  -0.430   0.6706
## Leg          -6.43905    4.71386  -1.366   0.1824
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 37.72 on 29 degrees of freedom
## Multiple R-squared:  0.6866, Adjusted R-squared:  0.6001
## F-statistic:  7.94 on 8 and 29 DF,  p-value: 1.306e-05
```

```
hist(rstandard(all_var_model), probability =TRUE, main="Histogram of Standardized Residuals - HtShoes",
# Overlay a normal distribution curve on top
curve(dnorm(x,mean=mean(rstandard(all_var_model)),sd=sd(rstandard(all_var_model))),col=2,lwd=2,add=TRUE
```



The regression model fitted with all available predictors exhibits several issues that compromise its effectiveness and interpretability:

Standard Errors: The standard errors of the regression coefficients, particularly for HtShoes and Ht, are excessively large. This suggests that the confidence intervals for these coefficients will be wide, diminishing the precision and reliability of these estimates. Such large standard errors are indicative of multicollinearity, especially given the near perfect correlation between HtShoes and Ht.

Statistical Significance: Except for the intercept, all predictors have p values significantly greater than the 0.05 threshold, indicating that they do not have statistically significant linear relationships with the response variable hipcenter when considered alongside all other predictors in the model. This lack of significance underscores the potential redundancy and dilution of explanatory power due to multicollinearity.

Relating Findings to Previous Parts: The lack of statistical significance and inflated standard errors correlate with earlier observations about multicollinearity among predictors. As previously noted, the high correlation among variables such as HtShoes, Ht, Seated and Leg introduces redundancy into the model, which prevents individual predictors from showing significant independent effects on the response variable. This multicollinearity complicates the model, reducing the clarity and utility of the predictors' coefficients.

Conclusion: The analysis reveals that while the model explains a moderate proportion of the variance in hipcenter (as indicated by an R^2 of approximately 0.6866), the issues of multicollinearity and non significant predictor effects suggest that a simpler model or one with fewer correlated variables might provide more reliable and interpretable results.

iv. Calculate the variance inflation factor of each predictor included in the fitted linear model. Which predictor displays the highest variance inflation factor? How does this finding relate to your findings on the standard errors of the estimated regression coefficients and the correlations between predictors?

```
library(car)
```

```
## Loading required package: carData

##
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':
##
##     recode

## The following object is masked from 'package:purrr':
##
##     some
```

```
vif_values<- vif(all_var_model)
vif_values
```

```
##        Age     Weight    HtShoes         Ht     Seated        Arm      Thigh
##   1.997931   3.647030 307.429378 333.137832   8.951054   4.496368   2.762886
##        Leg
##   6.694291
```

Based on the initial Variance Inflation Factor (VIF) Values for each predictor variable in the fitted linear model, **Ht has the highest variance inflation factor, 333.137832.** In addition, HtShoes, Seated and Leg all have VIF > 5, which can pose collinearity problems within the regression model.

This finding relates to the findings on the standard errors of the estimated regression coefficients and the correlations between predictors because we saw Ht and HtShoes were almost perfectly positively correlated, HtShoes and Seated, and HtShoes and Leg were also very strongly positively correlated, and as a result we saw the standard errors for HtShoes and Ht extremely large and came to the conclusion that none of our predictor variables are statistically significant in helping explain the variation in the response variable.

v. Remove any highly collinear predictors from the multiple linear model. How does that affect the estimated standard errors of the remaining regression coefficients?

The plan is to remove all predictors with VIF > 5. However, to understand the data better, we will incrementally remove highly collinear predictors.

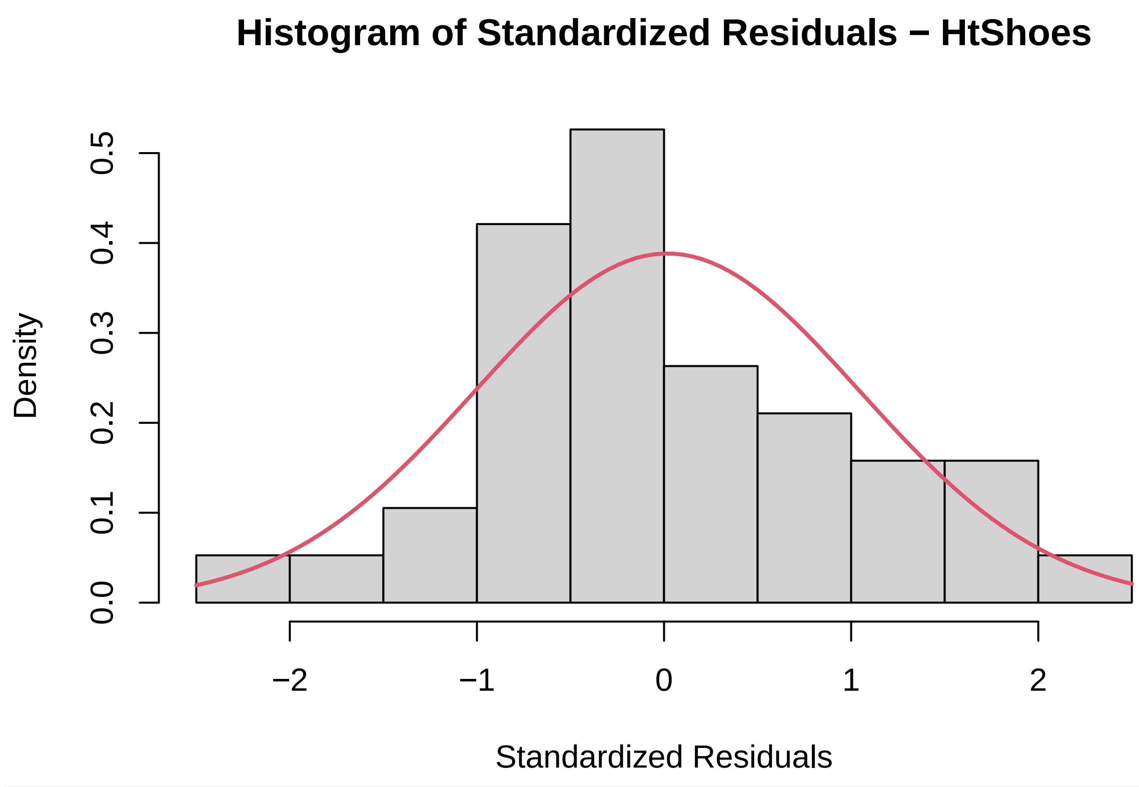
```
model <- lm(hipcenter ~ . - HtShoes, data=seatpos)
vif(model)
```

```
##       Age    Weight        Ht    Seated       Arm     Thigh       Leg
##  1.875729  3.628705 23.352154  8.808440  4.482567  2.626556  6.690858
```

```
summary(model)
```

```
##
## Call:
## lm(formula = hipcenter ~ . - HtShoes, data = seatpos)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -74.570 -21.565  -5.471  24.739  61.595
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 435.80897  163.97188   2.658   0.0125 *
## Age           0.73678    0.54404   1.354   0.1858
## Weight        0.03279    0.32502   0.101   0.9203
## Ht           -2.09530    2.64036  -0.794   0.4337
## Seated        0.40267    3.67390   0.110   0.9135
## Arm          -1.26842    3.83378  -0.331   0.7431
## Thigh        -0.98000    2.55332  -0.384   0.7038
## Leg          -6.46852    4.63953  -1.394   0.1735
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 37.13 on 30 degrees of freedom
## Multiple R-squared:  0.6857, Adjusted R-squared:  0.6124
## F-statistic: 9.351 on 7 and 30 DF,  p-value: 4.157e-06
```

```
hist(rstandard(model), probability =TRUE, main="Histogram of Standardized Residuals - HtShoes", xlab="S

# Overlay a normal distribution curve on top
curve(dnorm(x,mean=mean(rstandard(model)),sd=sd(rstandard(model))),col=2,lwd=2,add=TRUE)
```



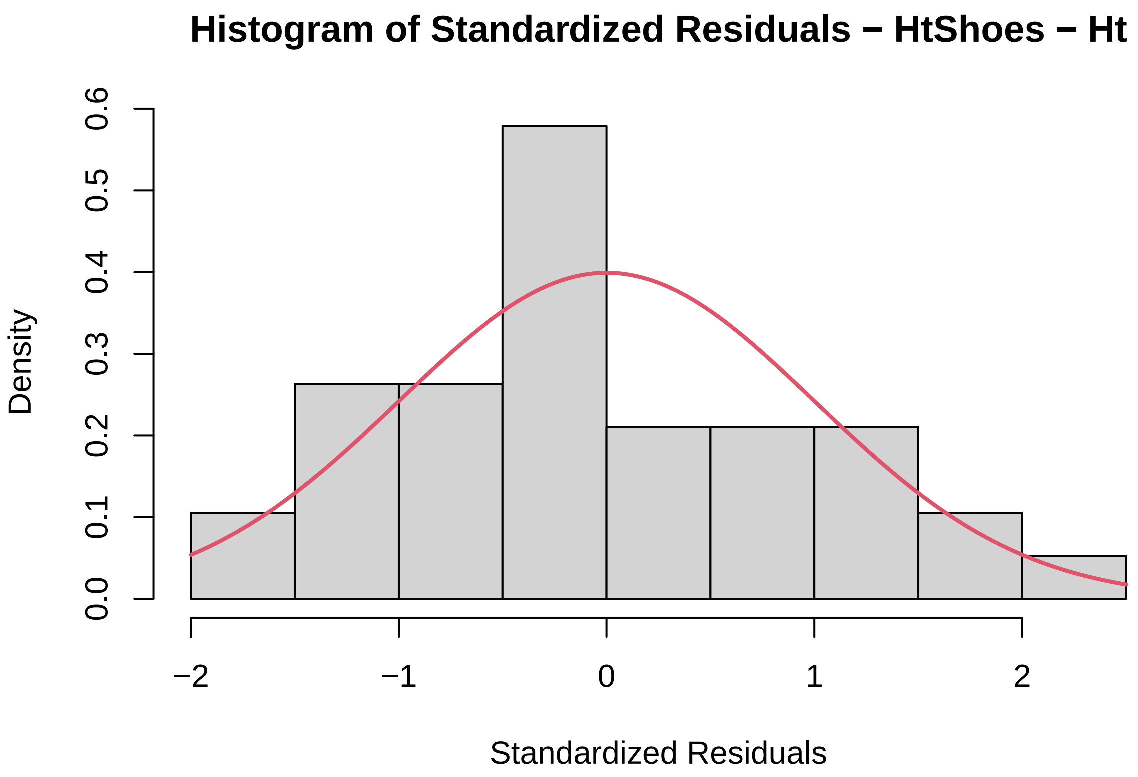
```r
new_model <- lm(hipcenter ~ . - HtShoes - Ht, data=seatpos)
vif(new_model)
```

```
##      Age   Weight   Seated      Arm    Thigh      Leg
## 1.786192 3.396124 4.069626 4.219519 2.061632 4.832701
```

```r
summary(new_model)
```

```
##
## Call:
## lm(formula = hipcenter ~ . - HtShoes - Ht, data = seatpos)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -68.296 -23.340  -5.672  24.183  74.065
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 409.00851  159.49517   2.564   0.0154 *
## Age           0.83110    0.52771   1.575   0.1254
## Weight       -0.03251    0.31254  -0.104   0.9178
## Seated       -1.73576    2.48225  -0.699   0.4896
## Arm          -2.00541    3.69731  -0.542   0.5914
## Thigh        -1.91970    2.24858  -0.854   0.3998
## Leg          -8.40876    3.91939  -2.145   0.0399 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 36.91 on 31 degrees of freedom
## Multiple R-squared:  0.6791, Adjusted R-squared:  0.617
## F-statistic: 10.94 on 6 and 31 DF,  p-value: 1.571e-06
```

```r
hist(rstandard(new_model), probability=TRUE,main="Histogram of Standardized Residuals - HtShoes - Ht",
# Overlay a normal distribution curve on top
curve(dnorm(x,mean=mean(rstandard(new_model)),sd=sd(rstandard(new_model))),col=2,lwd=2,add=TRUE)
```



```r
newer_model <- lm(hipcenter ~ . - HtShoes - Ht - Seated, data=seatpos)
vif(newer_model)
```

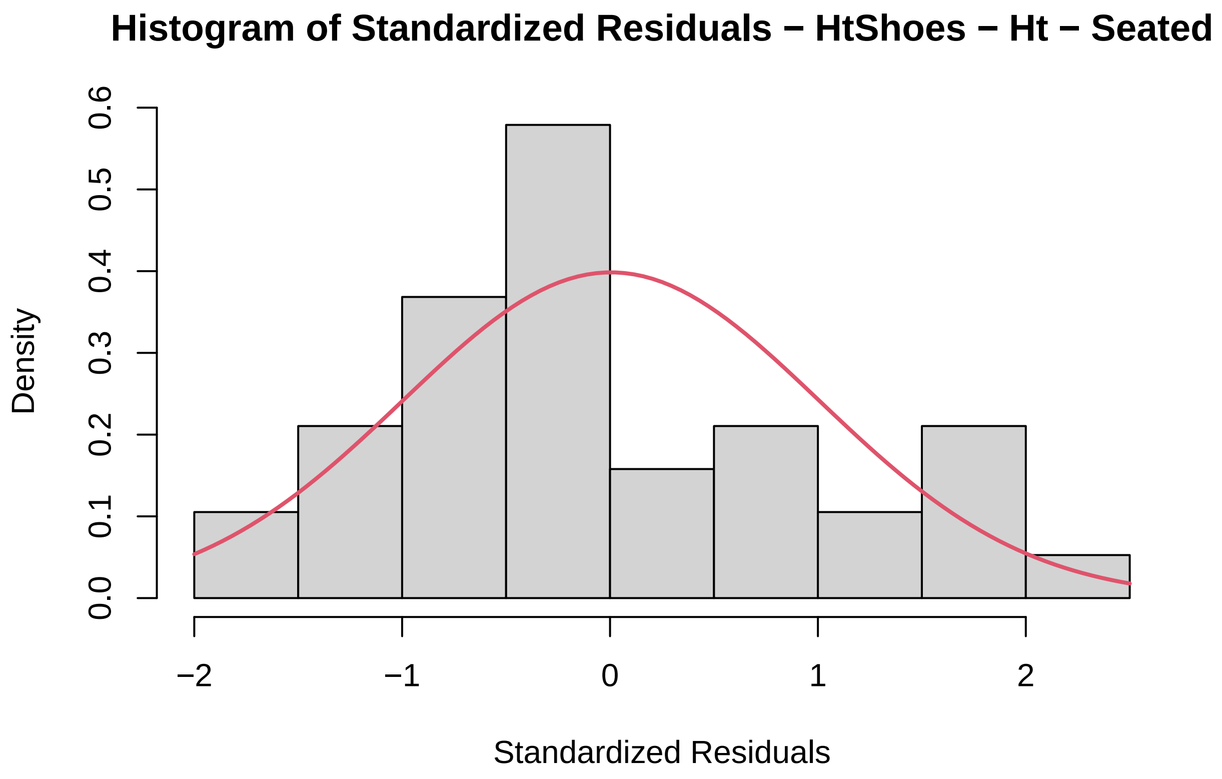
```
##      Age   Weight      Arm    Thigh      Leg
## 1.566978 2.797352 4.179555 2.010088 4.324249
```

```r
summary(newer_model)
```

```
##
## Call:
## lm(formula = hipcenter ~ . - HtShoes - Ht - Seated, data = seatpos)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -65.145 -19.641  -4.314  21.094  78.270
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 314.2850    83.5218   3.763 0.000678 ***
## Age           0.9604     0.4903   1.959 0.058909 .
## Weight       -0.1243     0.2814  -0.442 0.661702
## Arm          -2.2570     3.6503  -0.618 0.540742
## Thigh        -2.1683     2.2025  -0.984 0.332263
## Leg          -9.2977     3.6778  -2.528 0.016601 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 36.62 on 32 degrees of freedom
## Multiple R-squared:  0.6741, Adjusted R-squared:  0.6231
## F-statistic: 13.24 on 5 and 32 DF,  p-value: 5.045e-07
```

```
hist(rstandard(newer_model), probability=TRUE,main="Histogram of Standardized Residuals - HtShoes - Ht -
# Overlay a normal distribution curve on top
curve(dnorm(x,mean=mean(rstandard(newer_model)),sd=sd(rstandard(newer_model))),col=2,lwd=2,add=TRUE)
```



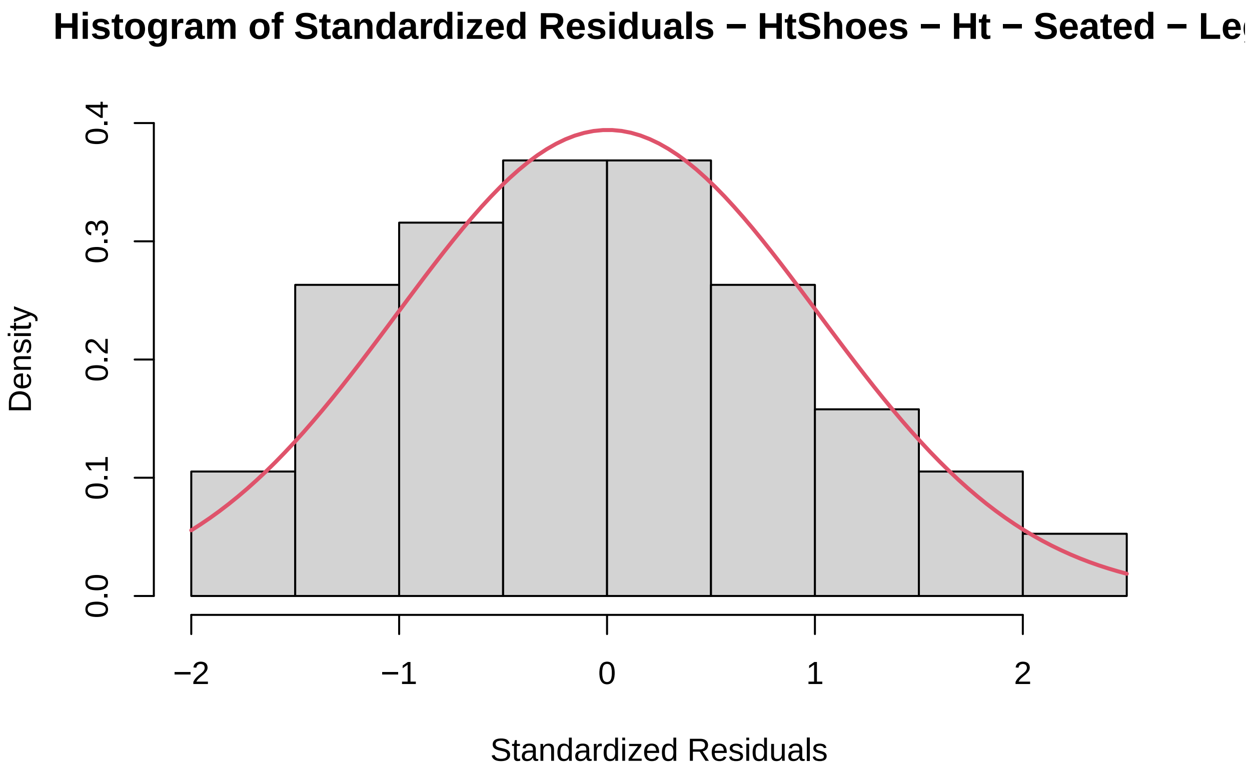
```
newest_model <- lm(hipcenter ~ . - HtShoes - Ht - Seated - Leg, data=seatpos)
vif(newest_model)
```

```
##      Age   Weight      Arm    Thigh
## 1.267087 2.130374 3.109446 1.953049
```

```
summary(newest_model)
```

```
##
## Call:
## lm(formula = hipcenter ~ . - HtShoes - Ht - Seated - Leg, data = seatpos)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -66.050 -26.644  -1.077  23.468  91.091
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 198.7446    75.4030   2.636  0.01269 *
## Age           1.5026     0.4756   3.160  0.00337 **
## Weight       -0.4716     0.2649  -1.781  0.08417 .
## Arm          -6.9265     3.3959  -2.040  0.04946 *
## Thigh        -3.1063     2.3417  -1.327  0.19377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.49 on 33 degrees of freedom
## Multiple R-squared:  0.609,  Adjusted R-squared:  0.5616
## F-statistic: 12.85 on 4 and 33 DF,  p-value: 2.064e-06
```

```
hist(rstandard(newest_model),probability=TRUE, main="Histogram of Standardized Residuals - HtShoes - Ht
# Overlay a normal distribution curve on top
curve(dnorm(x,mean=mean(rstandard(newest_model)),sd=sd(rstandard(newest_model))),col=2,lwd=2,add=TRUE)
```



The estimated standard errors of the remaining regression coefficients decrease as we remove highly collinear variables. Furthermore, the histogram of the standardized residuals of the regression model with all very collinear variables removed, $VIF > 5$, emulate a normal distribution the best out of all regression models tested.

However, the model removing all highly collinear variables, $VIF > 8$, is the model with the highest Adjusted R^2 out of the ones we tested using VIF criteria for removal.

The insight here is that removing variables that are collinear with VIF will improve the model as the estimated standard errors for the remaining regression coefficients go down and the standardized residuals are corrected to follow a normal distribution, but to find the best model, we need to inspect other diagnostics, and verify with other information criteria.

Problem 3

```
# Make sure the data was read in properly
head(brake)
```

```
##    distance speed
##       <int> <int>
## 1:        4     4
## 2:        2     5
## 3:        4     5
## 4:        8     5
## 5:        8     5
## 6:        7     7
```

```
# Determine if there are any missing values in the data
sum(is.na(brake))
```

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

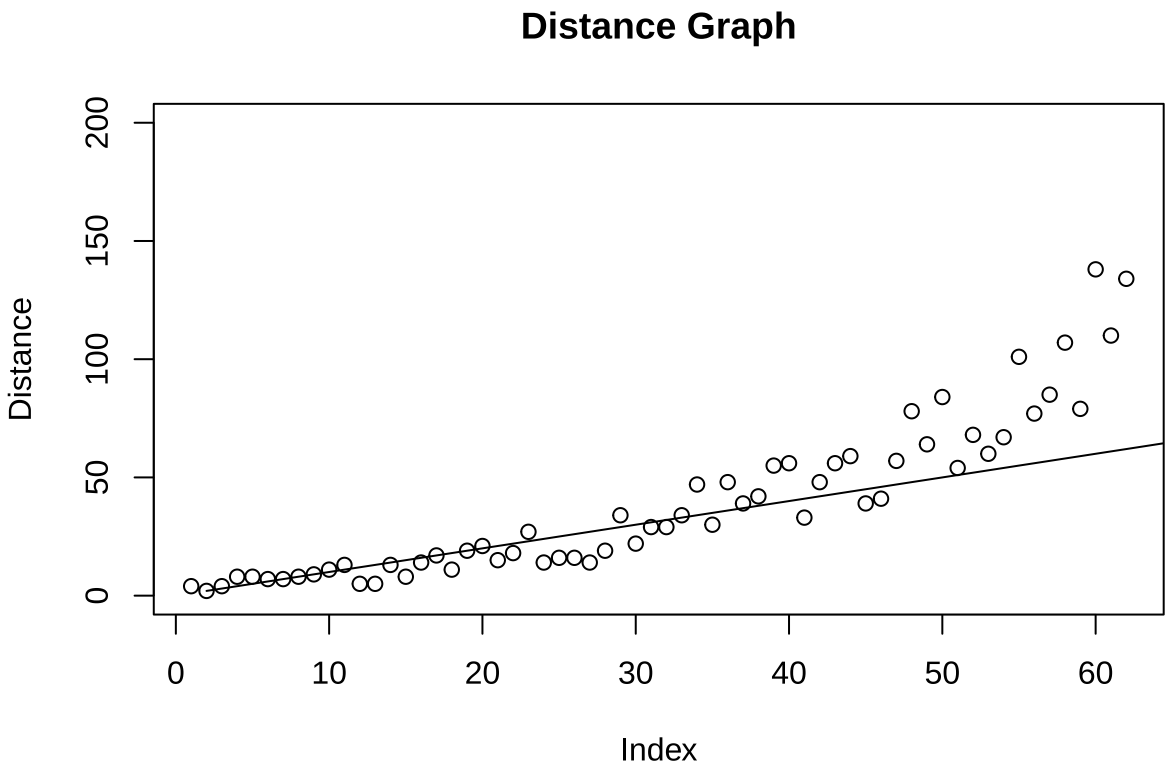
```
# No missing values
```

i. Create plots to examine the distribution of each variable separately, as well as the relationship between them. Comment on your findings.

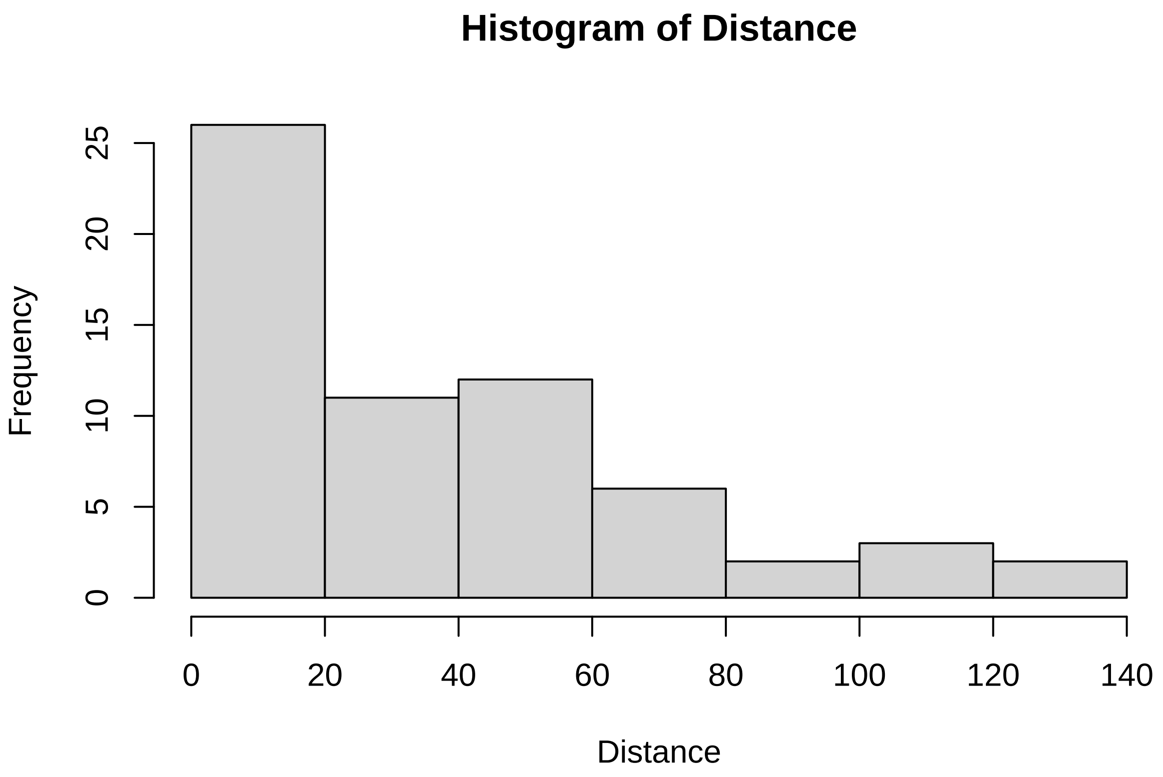
For examining the distribution of each variable separately, we will create histograms representing the frequency and spread of observation values for each variable.

The relationship between them is plotting one against the other, and overlaying a simple linear regression model.

```
# Plot each variable
plot(brake$distance, ylab = 'Distance',main="Distance Graph", ylim=c(0,200))
# Overlay a y=x curve, showing a positive linear relationship
curve((x),from=min(brake$distance), to = max(brake$distance), add=TRUE)
```

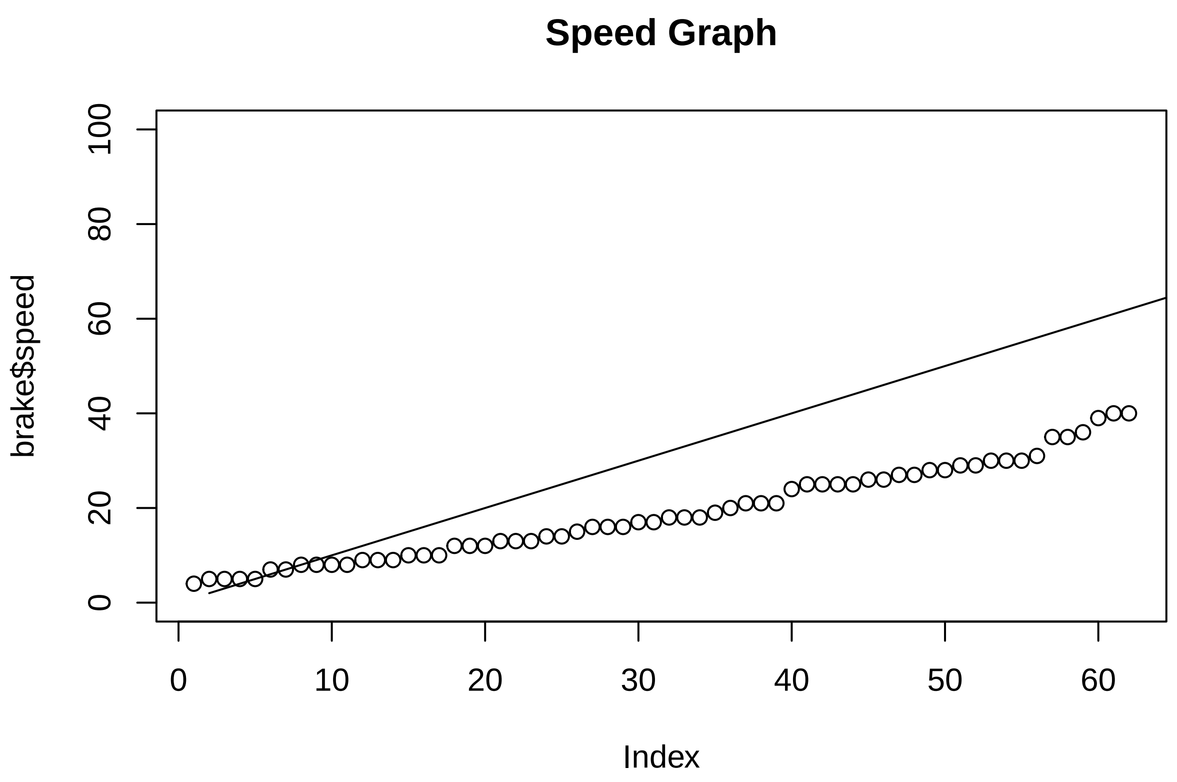


```
# Plot the distribution of each variable
hist(brake$distance, main = "Histogram of Distance", xlab='Distance')
```

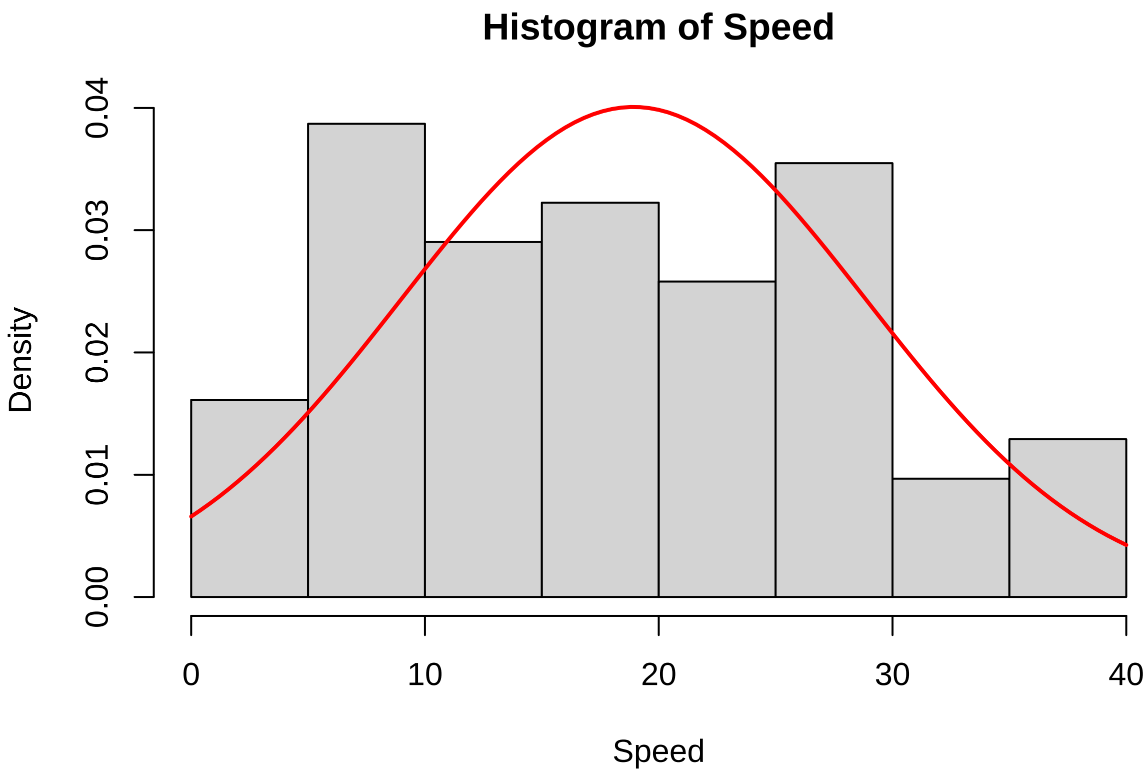


Based on the findings, the distribution of the distance variable is skewed. It is right skewed as the bulk of the data is on the left side of the distribution. A transformation would be good to try and account for this.

```r
# Plot each variable
plot(brake$speed, main="Speed Graph", ylim=c(0,100))

# Overlay a y=x curve, showing a positive linear relationship
curve((x),from=min(brake$distance), to = max(brake$distance), add=TRUE)
```

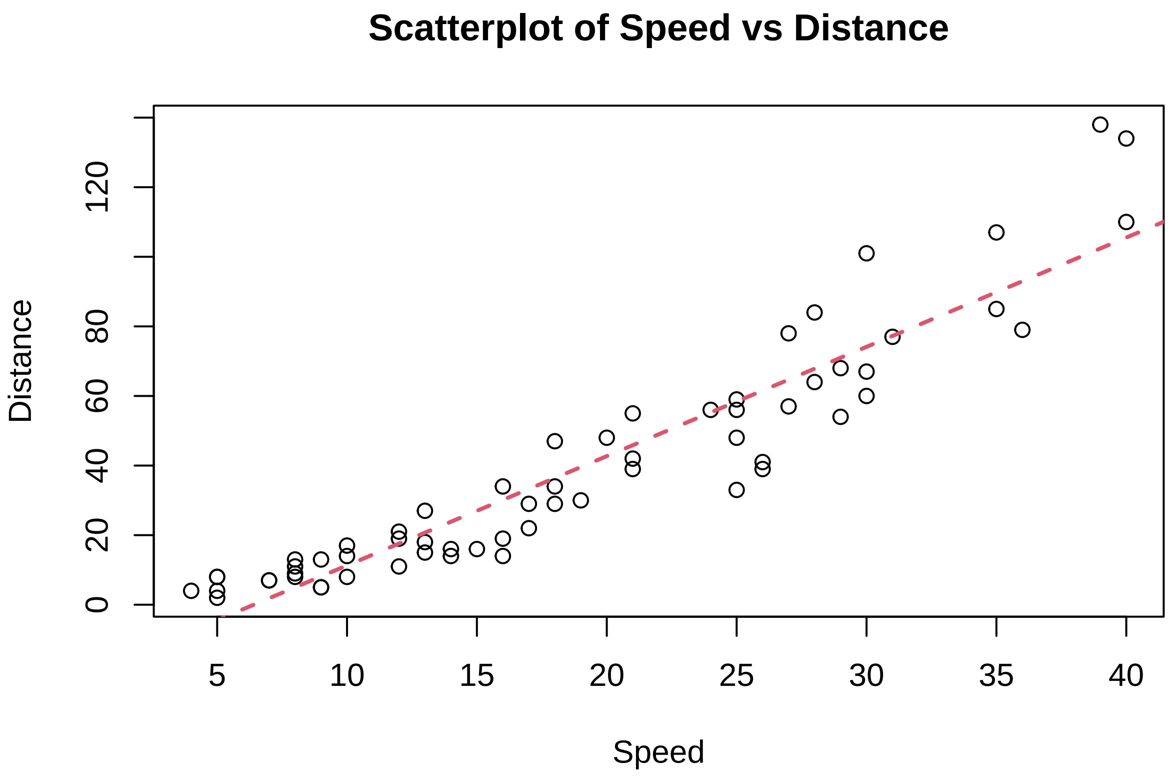


```r
# Plot the distribution of each variable
hist(brake$speed,probability=TRUE, main = "Histogram of Speed", xlab='Speed')

# Add a normal distribution to see if it could fit variable
curve(dnorm(x, mean=mean(brake$speed), sd=sd(brake$speed)), col="red", lwd=2, add=TRUE)
```



Based on the findings, the distribution of the speed variable is relatively normal, but a transformation couldl also potentially be used to make it better.

```r
# Plot the relationship between variables
plot(brake$speed,brake$distance, main = "Scatterplot of Speed vs Distance", xlab = 'Speed', ylab='Distar

# Overlay the regression model for clearer insights
brake_model <- lm(distance ~ speed, data=brake)
abline(brake_model,col=2,lwd=2,lty=2)
```



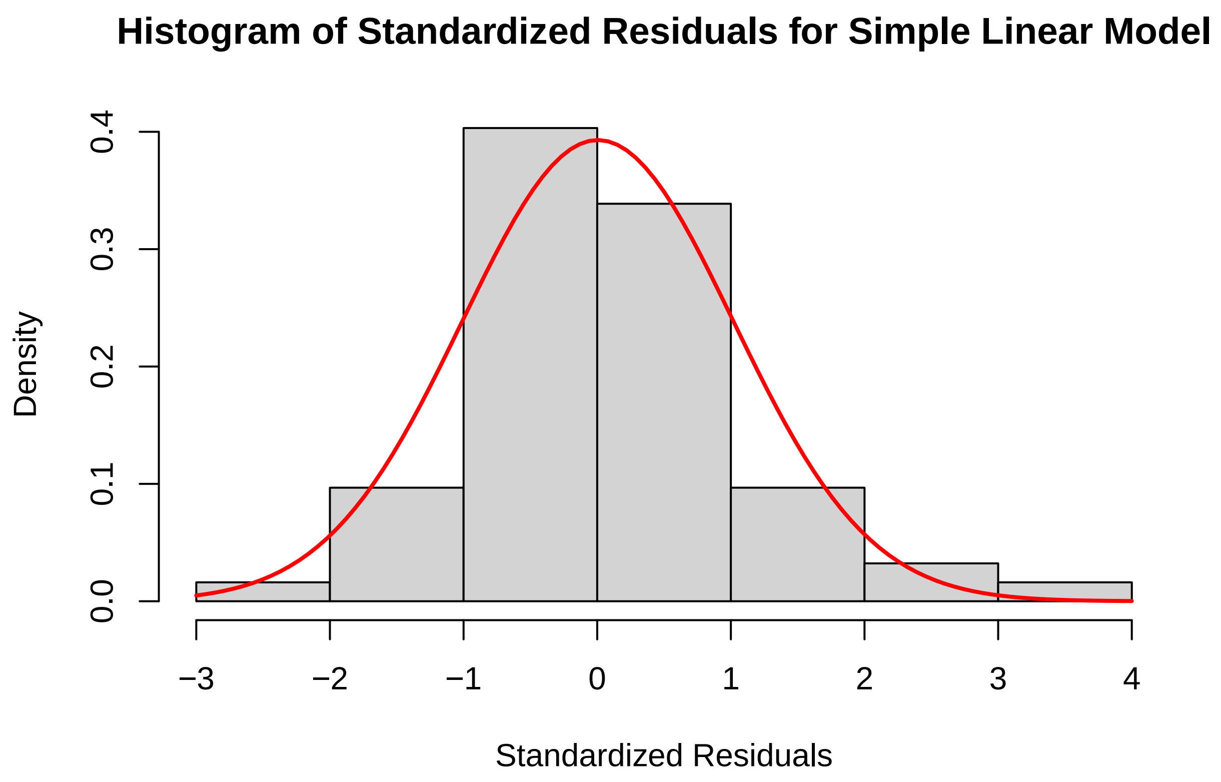
The relationship between distance and speed is highly positively linear. This means that as speed increases, the braking distance also increases.

ii. Fit a suitable simple linear model. Create plots to examine the distribution of the standardized residuals for the fitted linear model, as well as their relationship with the predictor. Are your findings consistent with those from the previous part?

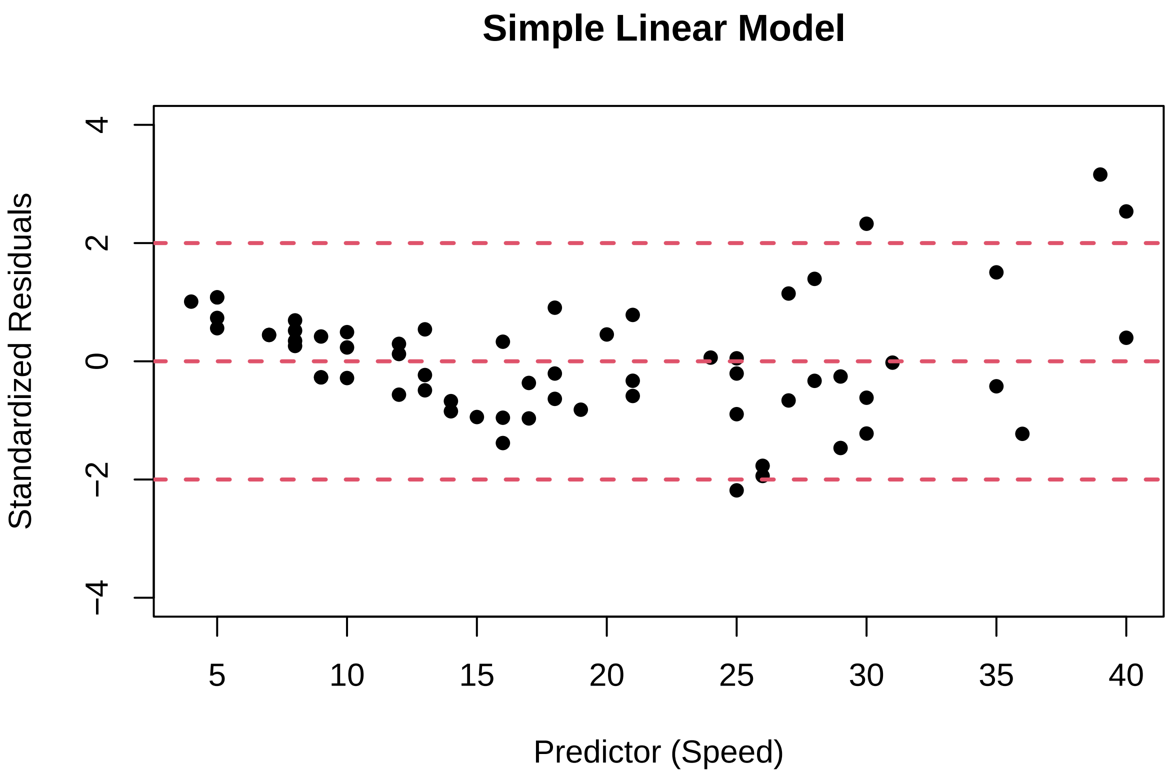
```r
# Fit a simple linear model
brake_model <- lm(distance ~ speed, data=brake)
summary(brake_model)
```

```
##
## Call:
## lm(formula = distance ~ speed, data = brake)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -25.410  -7.343  -1.334   5.927  35.608
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -20.1309     3.2308  -6.231 5.04e-08 ***
## speed         3.1416     0.1514  20.751  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.77 on 60 degrees of freedom
## Multiple R-squared:  0.8777, Adjusted R-squared:  0.8757
## F-statistic: 430.6 on 1 and 60 DF,  p-value: < 2.2e-16
```

```
# Create plots to examine the distribution of the standardized residuals for the fitted linear model
hist(rstandard(brake_model), probability =TRUE, main="Histogram of Standardized Residuals for Simple Li

# Add a normal distribution to see if it could fit variable
curve(dnorm(x, mean=mean(rstandard(brake_model)), sd=sd(rstandard(brake_model))), col="red", lwd=2, add
```



```
# Create plots to examine relationship of residuals with the predictor
plot(rstandard(brake_model) ~ speed, brake, main= " Simple Linear Model", ylim=c(-4,4), xlab = 'Predict
abline(h=0,col=2,lwd=2,lty=2)
abline(h=2,col=2,lwd=2,lty=2)
abline(h=-2,col=2,lwd=2,lty=2)
```



Based on the findings, the standardized residuals for the fitted linear model are following a normal distributed. The relationship of the standardized residuals with the predictor (speed), appears to be random and uniformly distributed throughout.

The histogram follows a normal distribution, and the model has a very high Adjusted R^2 which means that the predictor variable does a great job at explaining the variation in the response variable.

The beta hat regression coefficient for speed is positive, and is statistically significant, which supports our previous conclusion from the last part.

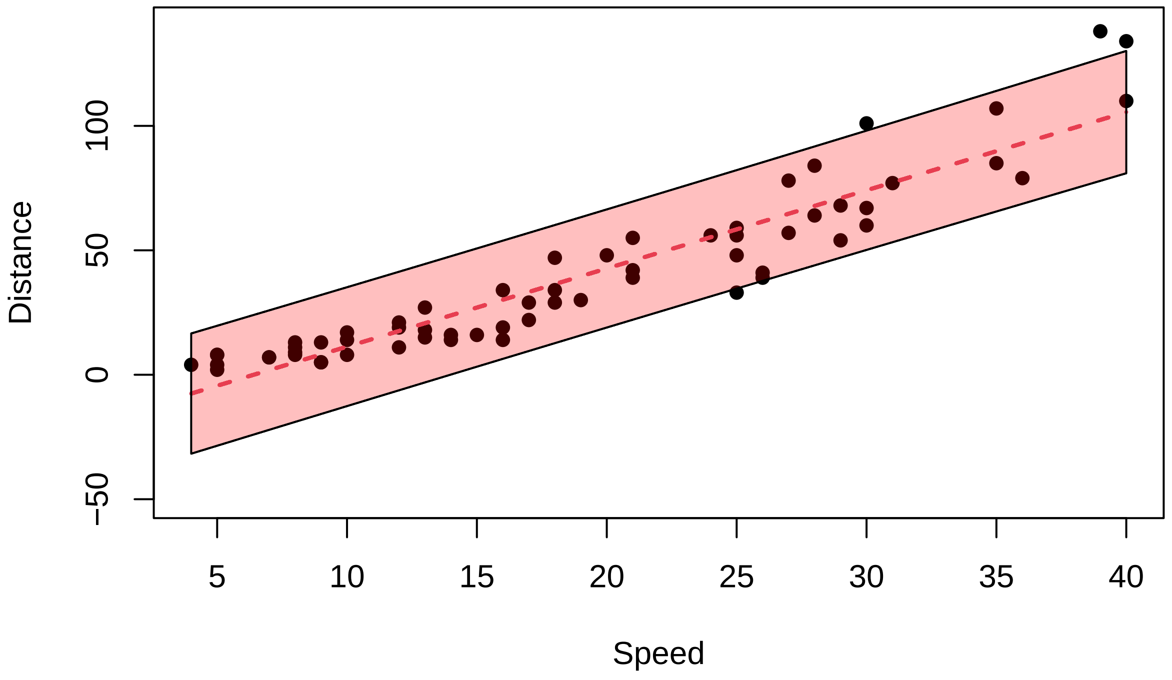
iii. Plot a 95% prediction region for the fitted regression line from the previous part. How flexibly does it perform at adapting to the relationship between the response variable and the predictor?

```r
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

##
## Attaching package: 'forecast'

## The following object is masked from 'package:nlme':
##
##     getResponse
```

```r
# Define the sequence of x values (soil water content in this case)
x <- seq(min(brake$speed), max(brake$speed), 1e-3)

# Get predictions along with prediction intervals
predictions <- predict(brake_model, newdata = data.frame(speed = x), interval = "prediction")
```

```r
# Plot the original data,
# Manually changed ylim to include all data and prediction interval
plot(distance ~ speed, data = brake,
     ylim = c(-50,140),
     pch = 16,
     xlab = "Speed",
     ylab = "Distance")

# Add the fitted line (predicted values)
lines(x, predictions[,1],'l', col = 2, lty = 2, lwd = 2)

# Add the prediction intervals as shaded areas
polygon(c(x, rev(x)),
        c(predictions[,2], rev(predictions[,3])),
        col = rgb(1, 0, 0, 0.25))
```



It is a good prediction interval, but the interval is not very flexible in the sense that it adjusts itself overtime based on the existing data. This is more likely to occur from other methods of prediction interval creation other than simple linear regression because of the inherent nature of the calculations SLR uses.

iv. It looks like the residuals from the simple linear model aren't quite normally distributed. Apply the bootstrap method to get a nonparametric estimate of the standard error for the slope coefficient. Does it differ by a large margin from the parametric estimate of the standard error you got from the ordinary linear regression model? How does any inaccurate estimation of the standard error of a regression coefficient impact your inference about the relationship between the response variable and the corresponding predictor?

```r
# Apply the bootstrap method
B = 10000
n = dim(brake)[1]
beta = numeric(B)
for (k in 1:B){
  boot = lm(distance ~ speed, data = brake, sample(n,replace=TRUE))
  beta[k] = boot$coefficients[2]

}

# Non-parametric estimate of the Standard Error for the slope coefficient
sd(beta)
```

```
## [1] 0.2011691
```

The standard error in the ordinary linear regression model is 0.1514, this standard error based on bootstrap method is 0.2019204, it is not a very large difference, but it is a difference.

Significance: An inaccurate estimation of the standard error of a regression coefficient impacts your inference about the relationship between the response and the corresponding predictor variable because it creates a wider confidence interval, which will deem statistical significance and consider good predictive power when in reality, the interval should be narrower, and less likely to be statistically significant, potentially affecting the model, what variables go into the model, and creating bias in the model's performance.

v. It looks like the variation in breaking distance increases as a function of speed. Try out 2 different data dependent choices of weights to fit a suitable weighted least squares model. Create the same plots from parts ii and iii for each WLS model. Which one of your 2 choices for weights appears to be performing best?

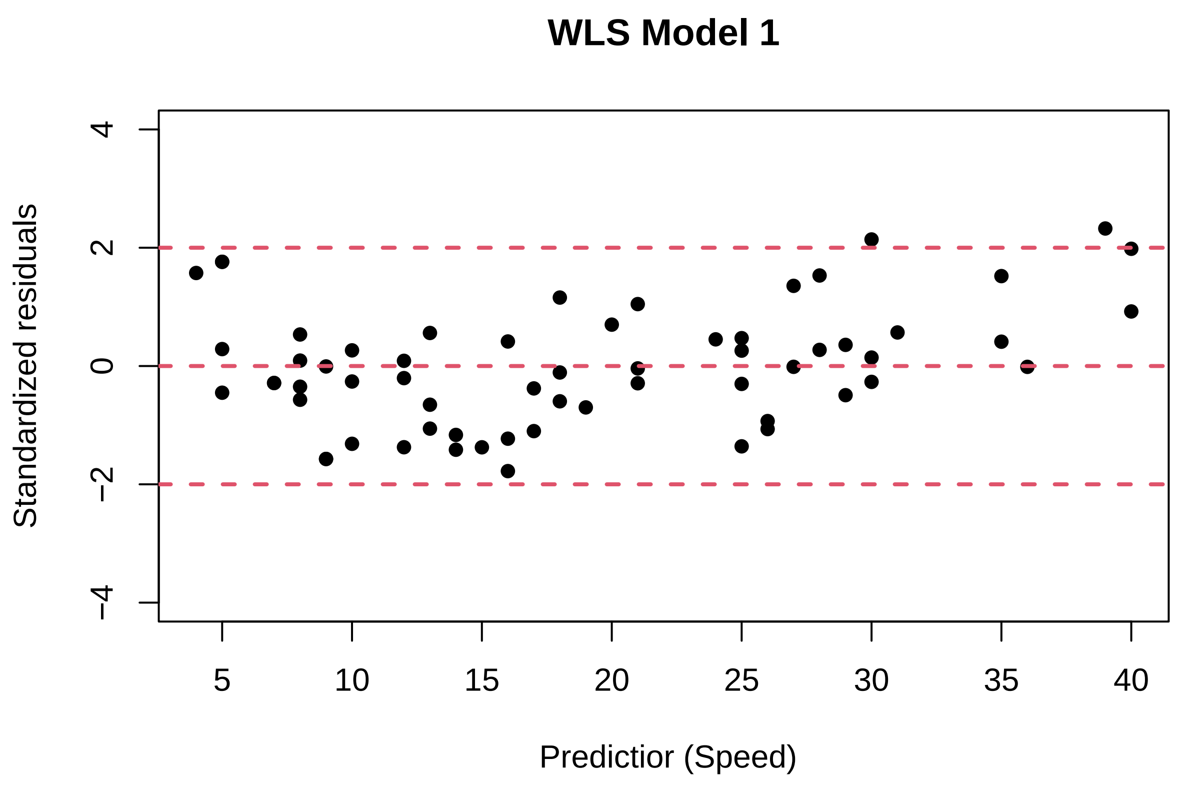
We observe a positive linear relationship between speed and distance, therefore, considering data-dependent choices for weights should include negative weights.

We can fit negative weights of polynomial degree 1,2,3. . . , depending on the relationship between the variables of interest. Here, we see that a weight of $1/speed$ and $1/speed^2$ are two potential weights to try out.

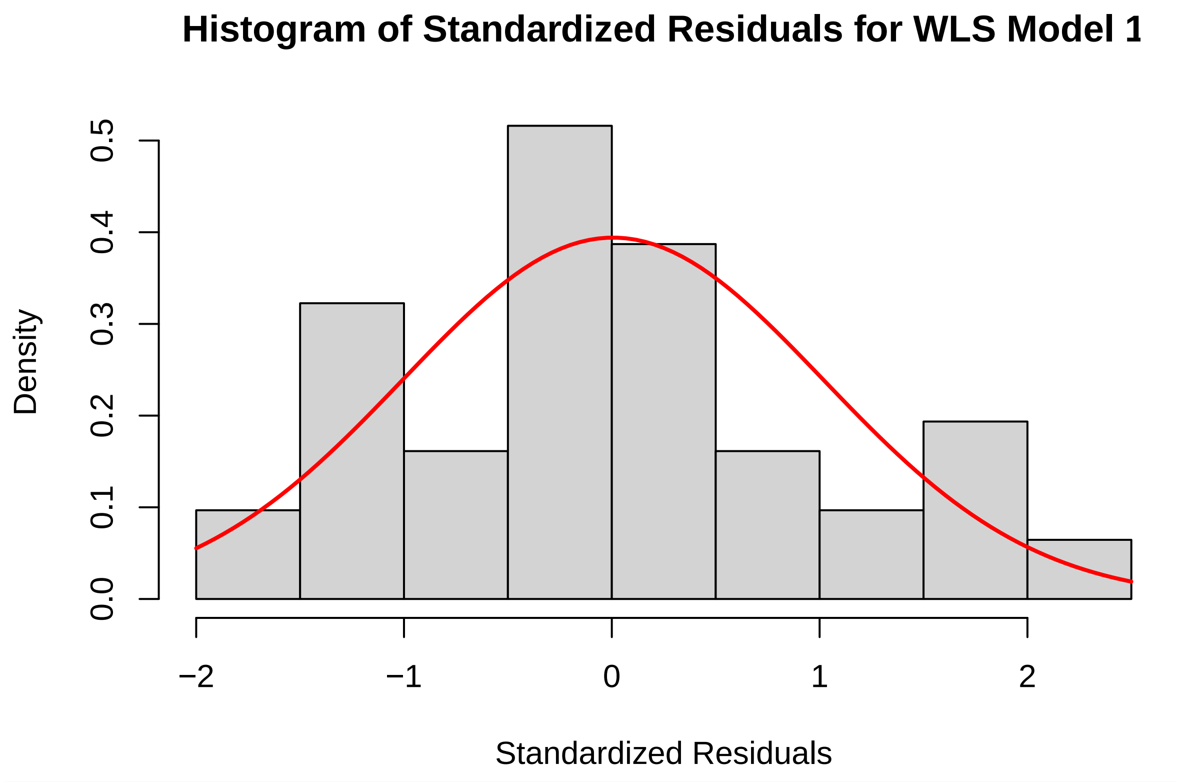
```
# First data-dependent choice of weight 1/speed^2
WLS <- lm(distance ~ speed, brake, weights= speed^(-2))
summary(WLS)
```

```
##
## Call:
## lm(formula = distance ~ speed, data = brake, weights = speed^(-2))
##
## Weighted Residuals:
##      Min       1Q   Median       3Q      Max
## -1.01336 -0.36497 -0.01549  0.29353  1.31668
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -9.0458     1.4257  -6.345 3.24e-08 ***
## speed         2.4537     0.1297  18.911  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5754 on 60 degrees of freedom
## Multiple R-squared:  0.8563, Adjusted R-squared:  0.8539
## F-statistic: 357.6 on 1 and 60 DF,  p-value: < 2.2e-16
```

```
plot(rstandard(WLS) ~ speed, brake, main = 'WLS Model 1', xlab = ' Predictior (Speed)', ylab='Standardi
abline(h=0,col=2,lwd=2,lty=2)
abline(h=2,col=2,lwd=2,lty=2)
abline(h=-2,col=2,lwd=2,lty=2)
```

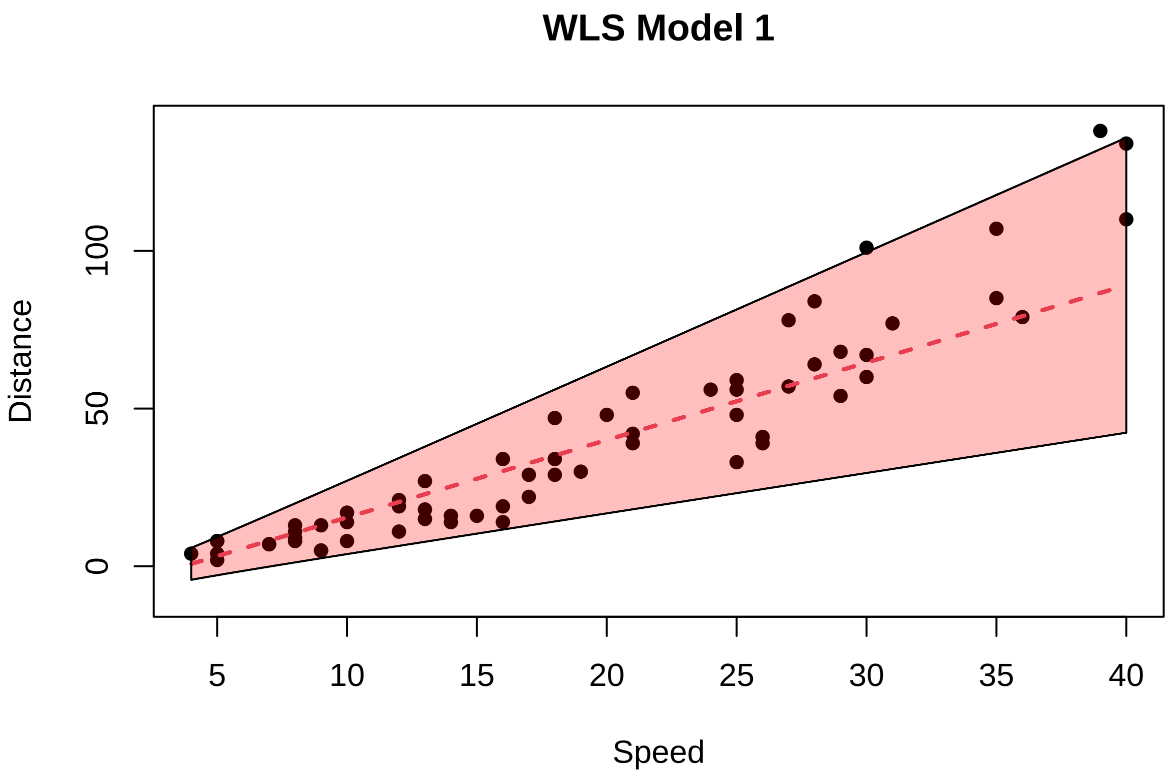


```
hist(rstandard(WLS), probability=TRUE, main="Histogram of Standardized Residuals for WLS Model 1", xlab=
curve(dnorm(x, mean=mean(rstandard(WLS)), sd=sd(rstandard(WLS))), col="red", lwd=2, add=TRUE)
```



```
# Create the plot with prediction interval

# Define the sequence of x values
```

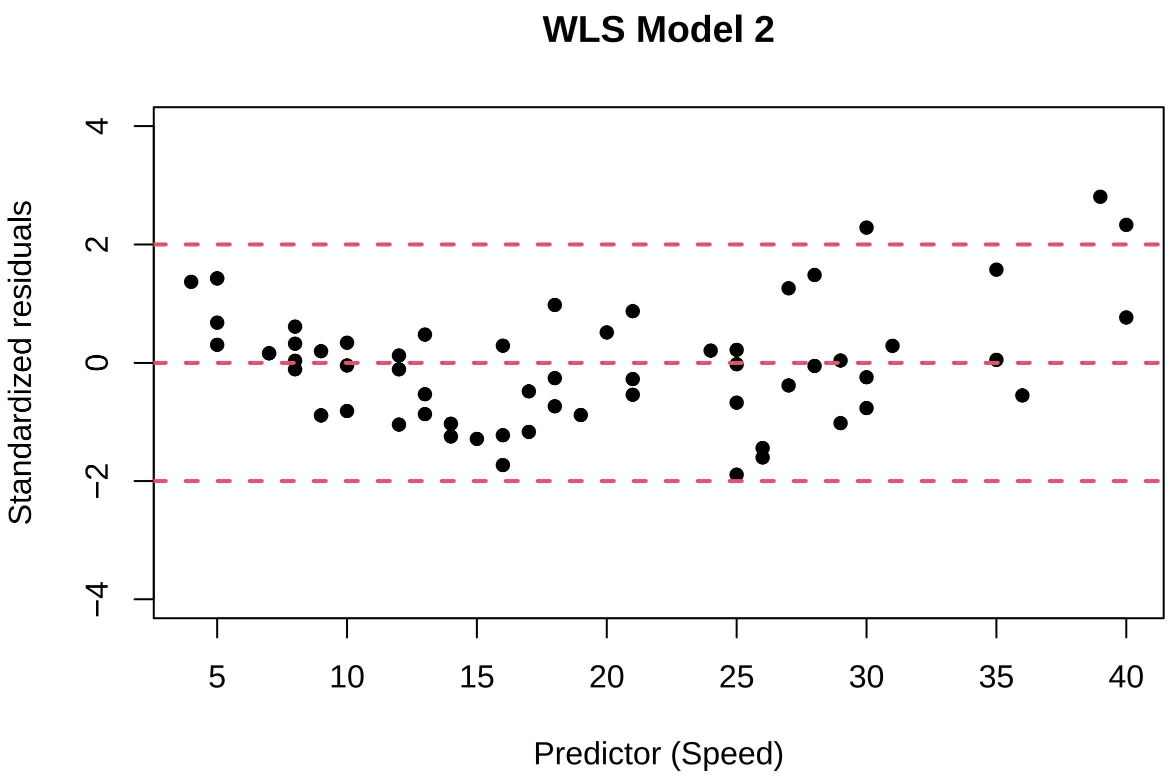
```r
x <- seq(min(brake$speed), max(brake$speed), 1e-3)

# Get predictions along with prediction intervals
predictions <- predict(WLS, newdata = data.frame(speed = x), interval = "prediction", weights = x^(-2))

# Plot the original data
plot(distance ~ speed, data = brake,
     ylim = c(-10,140),
     pch = 16,
     xlab = "Speed",
     ylab = "Distance", main = "WLS Model 1")

# Add the fitted line (predicted values)
lines(x, predictions[,1],'l', col = 2, lty = 2, lwd = 2)

# Add the prediction intervals as shaded areas
polygon(c(x, rev(x)),
        c(predictions[,2], rev(predictions[,3])),
        col = rgb(1, 0, 0, 0.25))
```



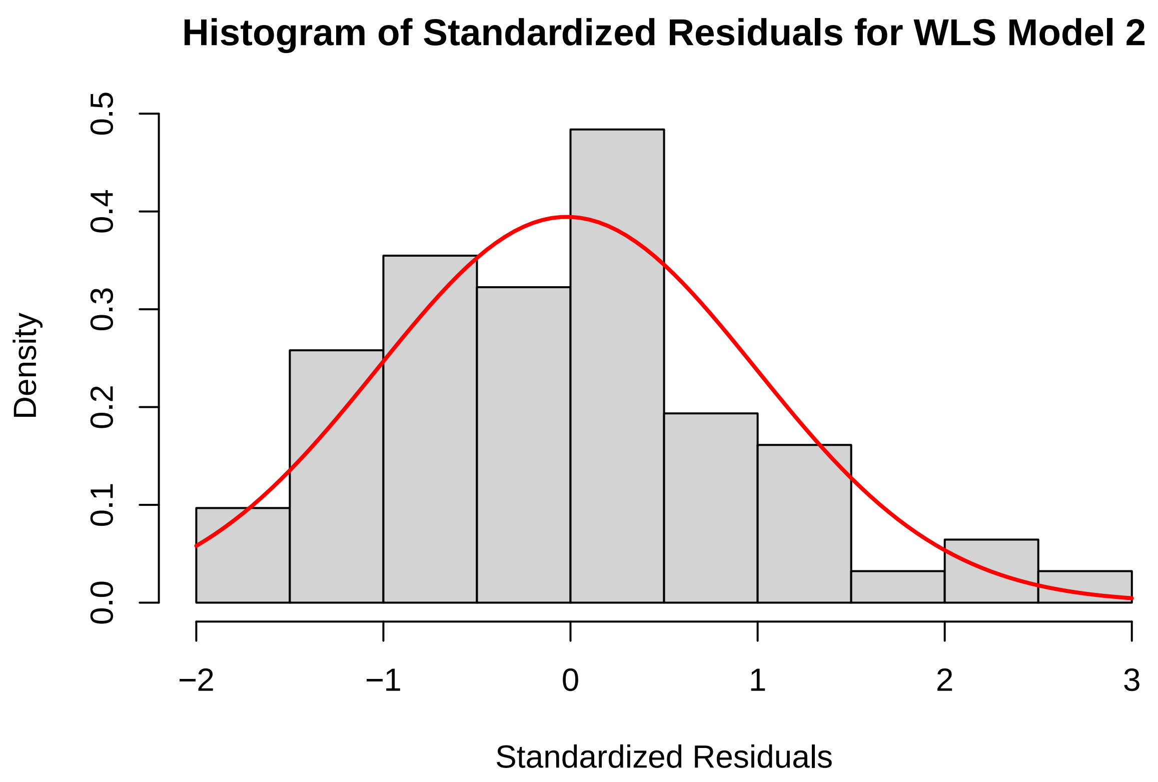
```r
# Second data-dependent choice of weight 1/speed
WLS_2 <- lm(distance ~ speed, brake, weights = speed^(-1))
summary(WLS_2)
```

```
##
## Call:
## lm(formula = distance ~ speed, data = brake, weights = speed^(-1))
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -4.6627 -1.9784 -0.0882  1.0927  6.8102
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -13.6095     2.1198   -6.42 2.42e-08 ***
## speed         2.7969     0.1336   20.93  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.495 on 60 degrees of freedom
## Multiple R-squared:  0.8795, Adjusted R-squared:  0.8775
## F-statistic:   438 on 1 and 60 DF,  p-value: < 2.2e-16
```

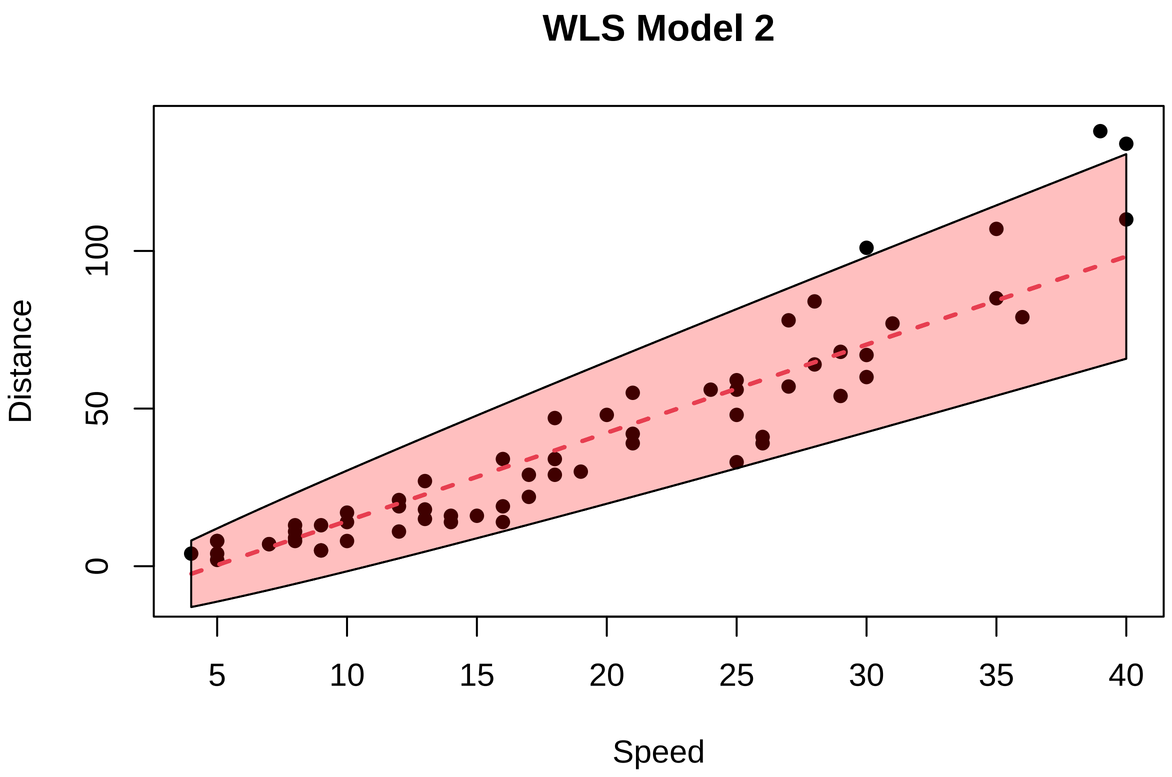
```
plot(rstandard(WLS_2) ~ speed, brake, main= "WLS Model 2", ylim=c(-4,4), xlab = 'Predictor (Speed)', yla
abline(h=0,col=2,lwd=2,lty=2)
abline(h=2,col=2,lwd=2,lty=2)
abline(h=-2,col=2,lwd=2,lty=2)
```



```
hist(rstandard(WLS_2),probability=TRUE, main="Histogram of Standardized Residuals for WLS Model 2", xlab
curve(dnorm(x, mean=mean(rstandard(WLS_2)), sd=sd(rstandard(WLS_2))), col="red", lwd=2, add=TRUE)
```



```r
# Create the plot with prediction interval

# Define the sequence of x values
x <- seq(min(brake$speed), max(brake$speed), 1e-3)

# Get predictions along with prediction intervals
predictions <- predict(WLS_2, newdata = data.frame(speed = x), interval = "prediction", weights = x^(-1

# Plot the original data
plot(distance ~ speed, data = brake,
     ylim =  c(-10,140),
     pch = 16,
     xlab = "Speed",
     ylab = "Distance", main = "WLS Model 2")

# Add the fitted line (predicted values)
lines(x, predictions[,1],'l', col = 2, lty = 2, lwd = 2)

# Add the prediction intervals as shaded areas
polygon(c(x, rev(x)),
        c(predictions[,2], rev(predictions[,3])),
        col = rgb(1, 0, 0, 0.25))
```



As we can see, Model 2 is much better. The inverse weight for Model 2 helps account for the positive linear relationship between Distance and Speed, by applying an inverse weight that is linear instead of quadratic, because we have seen a positive linear relationship between Distance and Speed, the fit of the WLS with an inverse weight that is more characteristic of the relationship between variables will produce better prediction intervals and Model 2 has fewer residuals beyond a specific bounded error $(-2, 2]$, emphasizing the precision and accuracy of the model is better than Model 1 too. The histogram for the distribution of the standardized errors for Model 2 looks more like a normal distribution than Model 1 as well. The linear relationship is good, but it is not perfect.

vi. It looks like the relationship between the response variable and the predictor isn't quite linear. Try out a power transformation of the predictor and create the same plots as part i. Is there any linear association between the response variable and a power transformation of the predictor? Fit a suitable polynomial regression model. How does the summary of the polynomial regression results compare to that of the simple linear model from part ii?

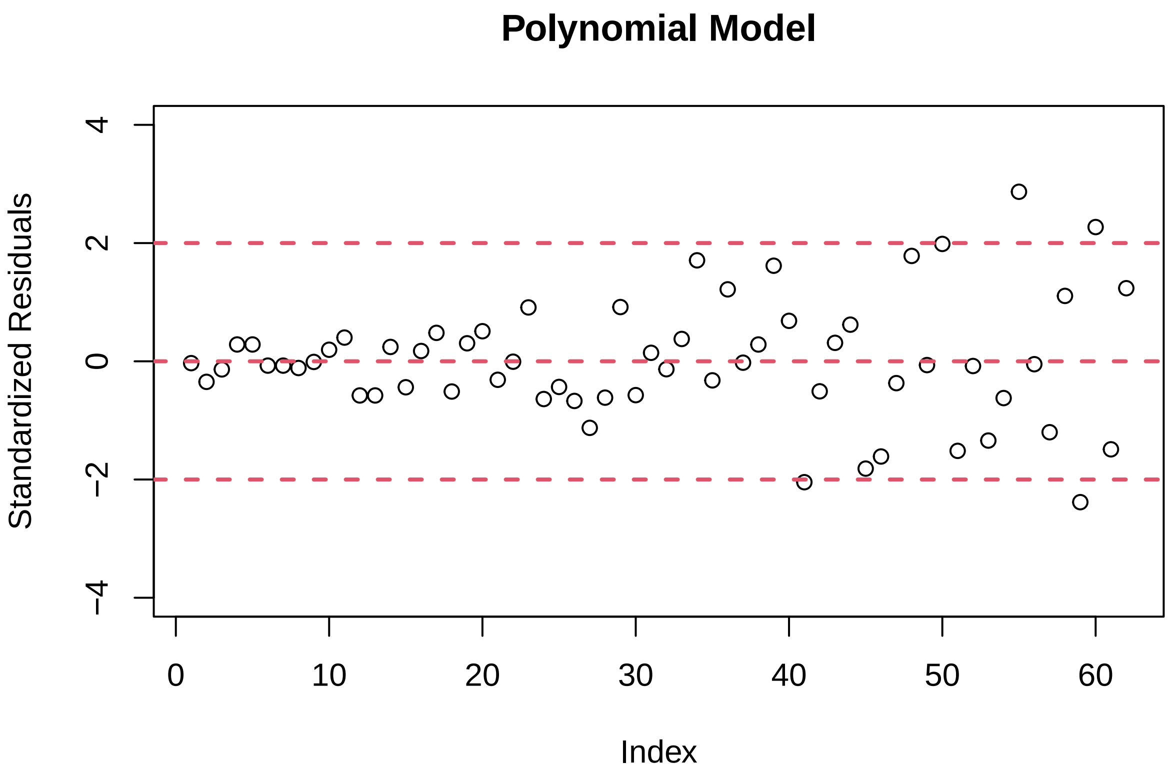
```
# Polynomials of order > 2 are not statistically significant
excessive_model <- lm(distance ~ poly(speed,6), brake)
summary(excessive_model)
```

```
##
## Call:
## lm(formula = distance ~ poly(speed, 6), data = brake)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -20.9471  -5.0606   0.0262   3.9681  30.1257
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)        39.306      1.293  30.402  < 2e-16 ***
## poly(speed, 6)1  244.211     10.180  23.988  < 2e-16 ***
## poly(speed, 6)2   49.960     10.180   4.908  8.6e-06 ***
## poly(speed, 6)3    4.373     10.180   0.430    0.669
## poly(speed, 6)4    8.845     10.180   0.869    0.389
## poly(speed, 6)5    3.104     10.180   0.305    0.762
## poly(speed, 6)6   -2.637     10.180  -0.259    0.797
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.18 on 55 degrees of freedom
## Multiple R-squared:  0.9161, Adjusted R-squared:  0.907
## F-statistic: 100.1 on 6 and 55 DF,  p-value: < 2.2e-16
```

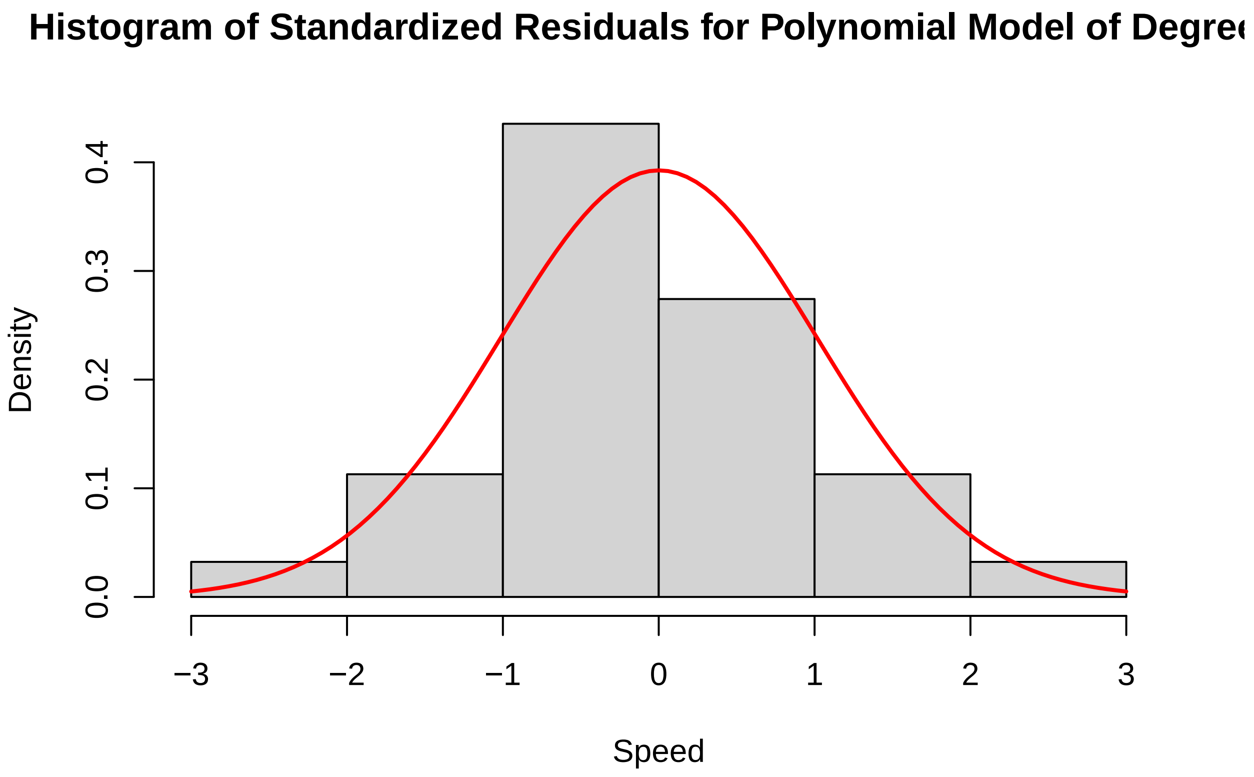
```
poly_model <- lm( distance ~ poly(speed,2), data=brake)
summary(poly_model)
```

```
##
## Call:
## lm(formula = distance ~ poly(speed, 2), data = brake)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -22.5192  -5.4527  -0.5519   3.8442  27.9373
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       39.306      1.261  31.178  < 2e-16 ***
## poly(speed, 2)1  244.211      9.927  24.601  < 2e-16 ***
## poly(speed, 2)2   49.960      9.927   5.033 4.83e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.927 on 59 degrees of freedom
## Multiple R-squared:  0.9144, Adjusted R-squared:  0.9115
## F-statistic: 315.3 on 2 and 59 DF,  p-value: < 2.2e-16
```

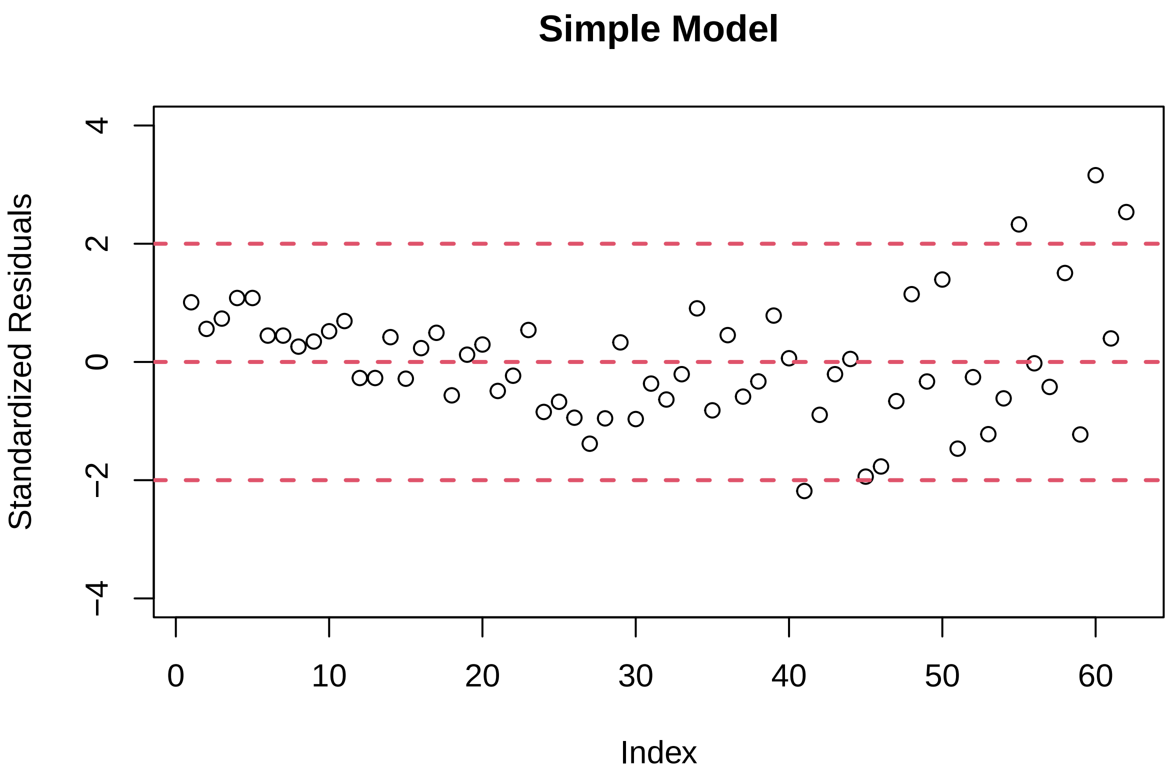
```
plot(rstandard(poly_model), ylab='Standardized Residuals', main='Polynomial Model', ylim=c(-4,4))
abline(h=0,col=2,lwd=2,lty=2)
abline(h=-2,col=2,lwd=2,lty=2)
abline(h=2,col=2,lwd=2,lty=2)
```



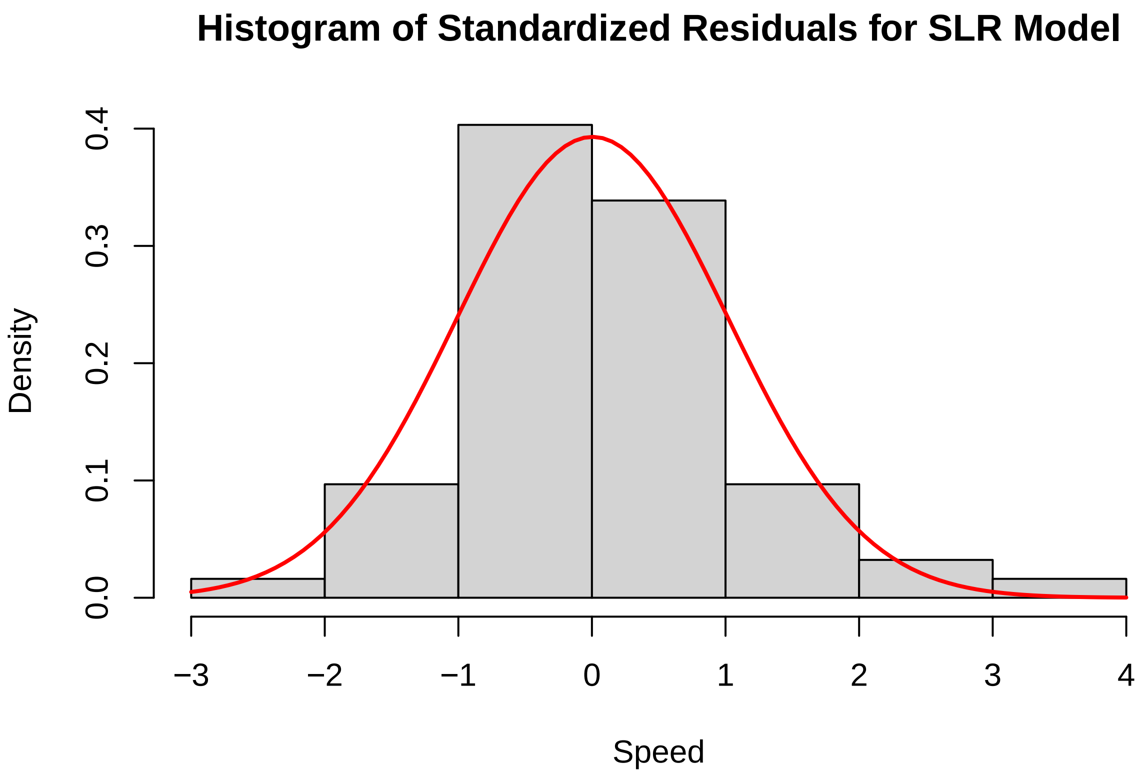
```
# Plot the distribution of each variable
hist(rstandard(poly_model),probability=TRUE, main = "Histogram of Standardized Residuals for Polynomial

# Add a normal distribution to see if it could fit variable
curve(dnorm(x, mean=mean(rstandard(poly_model)), sd=sd(rstandard(poly_model))), col="red", lwd=2, add=T
```



```r
plot(rstandard(brake_model), ylab="Standardized Residuals", main='Simple Model',ylim=c(-4,4))
abline(h=0,col=2,lwd=2,lty=2)
abline(h=-2,col=2,lwd=2,lty=2)
abline(h=2,col=2,lwd=2,lty=2)
```



```r
hist(rstandard(brake_model),probability=TRUE, main = "Histogram of Standardized Residuals for SLR Model

# Add a normal distribution to see if it could fit variable
curve(dnorm(x, mean=mean(rstandard(brake_model)), sd=sd(rstandard(brake_model))), col="red", lwd=2, add
```



Yes, there is a highly linear relationship between a polynomial of degree 2 for the predictor variable and the response variable.

Based on the analysis of the estimated standard error for the predictor variables in the varying models, the SLR model is better because it provides a tighter confidence interval for the range of values of the beta coefficient for the influence of the predictor on the response.

However, the histograms of the distribution of the standardized residuals show that the polynomial model's standardized residuals more closely follow a normal distribution than the SLR model.

Therefore, the polynomial model of degree 2 was a better model than the simple linear model based on the information criteria, Adjusted R^2. This is derived from the idea that polynomial models are more complex and they can capture more of the relationship between variables, so for the current data, the polynomial model is much better. For predictive purposes, we will need to look at other metrics to determine which model is better.

vii. Try out a suitable transformation of the response variable and fit a simple linear model with speed as the predictor. Create the same plots as parts i, ii and iii. Comment on whether this transformation of the response variable has alleviated any linear regression violations we observed in the previous parts. How does the summary of the simple linear model with the transformed response variable compare to that of the polynomial regression model from the previous part?

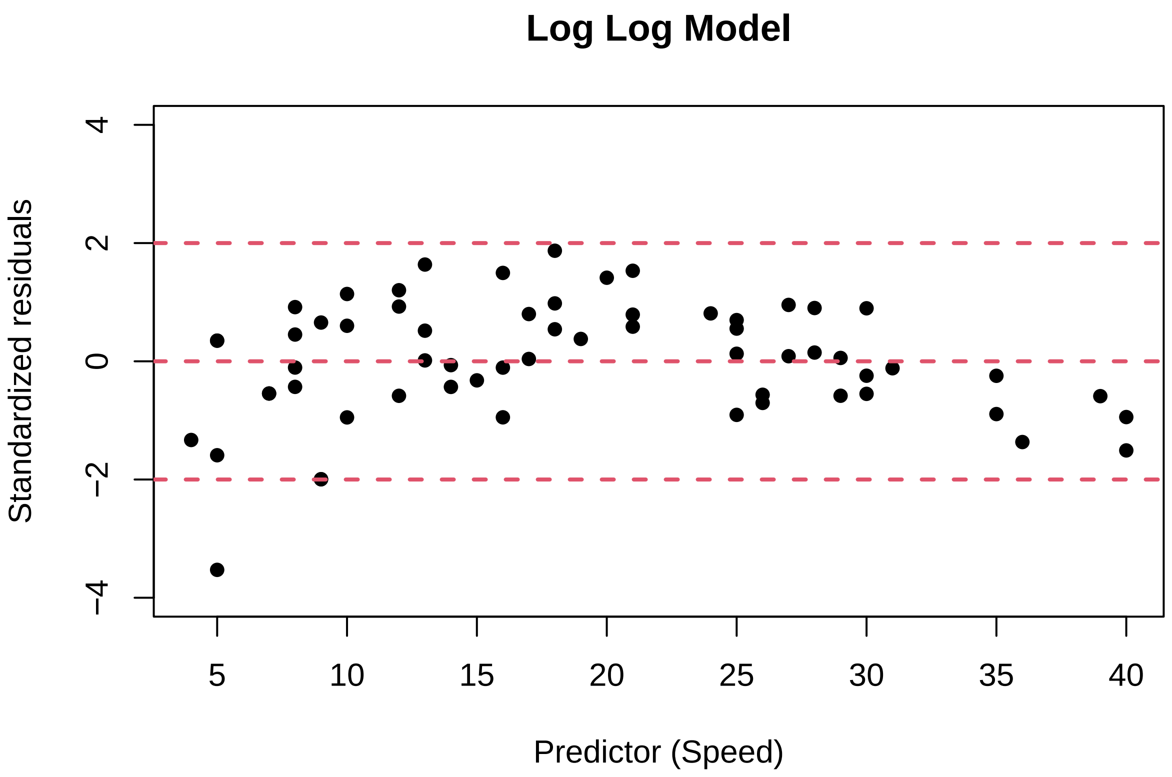
```r
# Create log-transformed variables and add them to the brake dataset
brake[, log_distance := log(brake$distance)]
brake[, log_speed := log(brake$speed)]

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

```
##    distance speed log_distance log_speed
##       <int> <int>        <num>     <num>
## 1:        4     4    1.3862944  1.386294
## 2:        2     5    0.6931472  1.609438
## 3:        4     5    1.3862944  1.609438
## 4:        8     5    2.0794415  1.609438
## 5:        8     5    2.0794415  1.609438
## 6:        7     7    1.9459101  1.945910
```

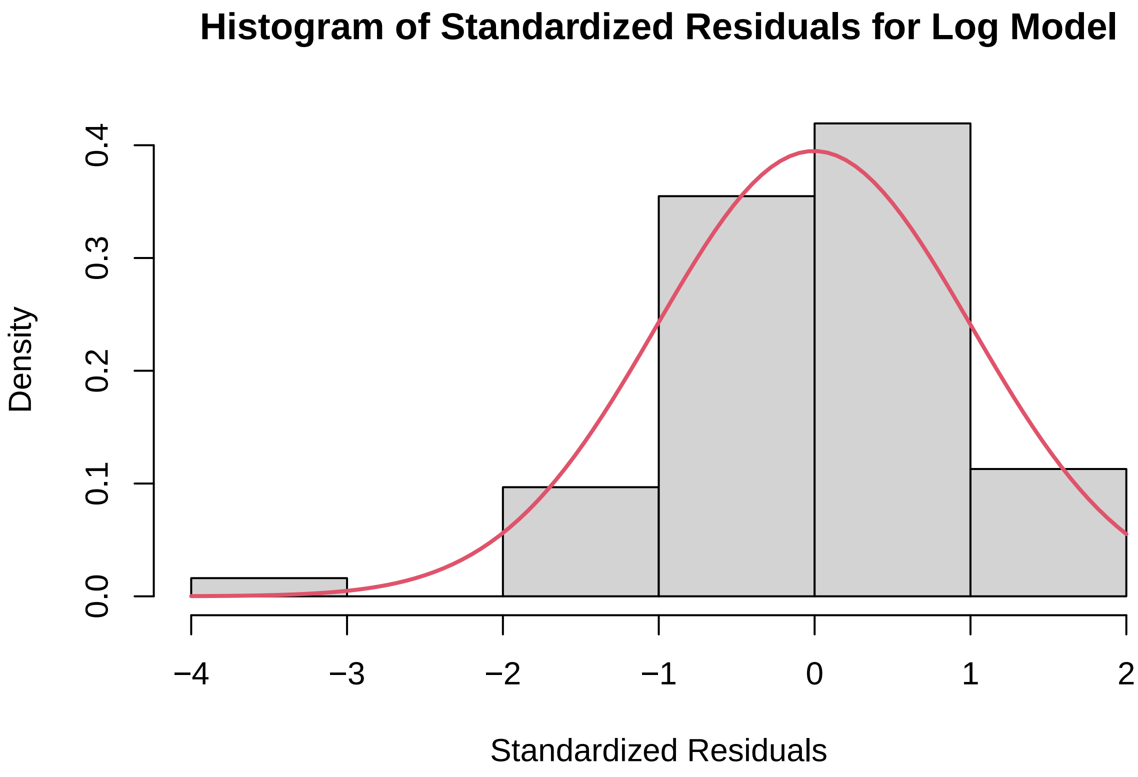
```
# Fit the model using the log-transformed variables
log_model <- lm(log_distance ~ speed, data = brake)

# Summary of the log-log model
summary(log_model)
```

```
##
## Call:
## lm(formula = log_distance ~ speed, data = brake)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.26143 -0.20640  0.01739  0.27826  0.67984
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.486995   0.100604   14.78   <2e-16 ***
## speed       0.093517   0.004714   19.84   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3665 on 60 degrees of freedom
## Multiple R-squared:  0.8677, Adjusted R-squared:  0.8655
## F-statistic: 393.5 on 1 and 60 DF,  p-value: < 2.2e-16
```

```
plot(rstandard(log_model) ~ speed, brake, main= "Log Log Model", ylim=c(-4,4), xlab = 'Predictor (Speed)
abline(h=0,col=2,lwd=2,lty=2)
abline(h=2,col=2,lwd=2,lty=2)
abline(h=-2,col=2,lwd=2,lty=2)
```

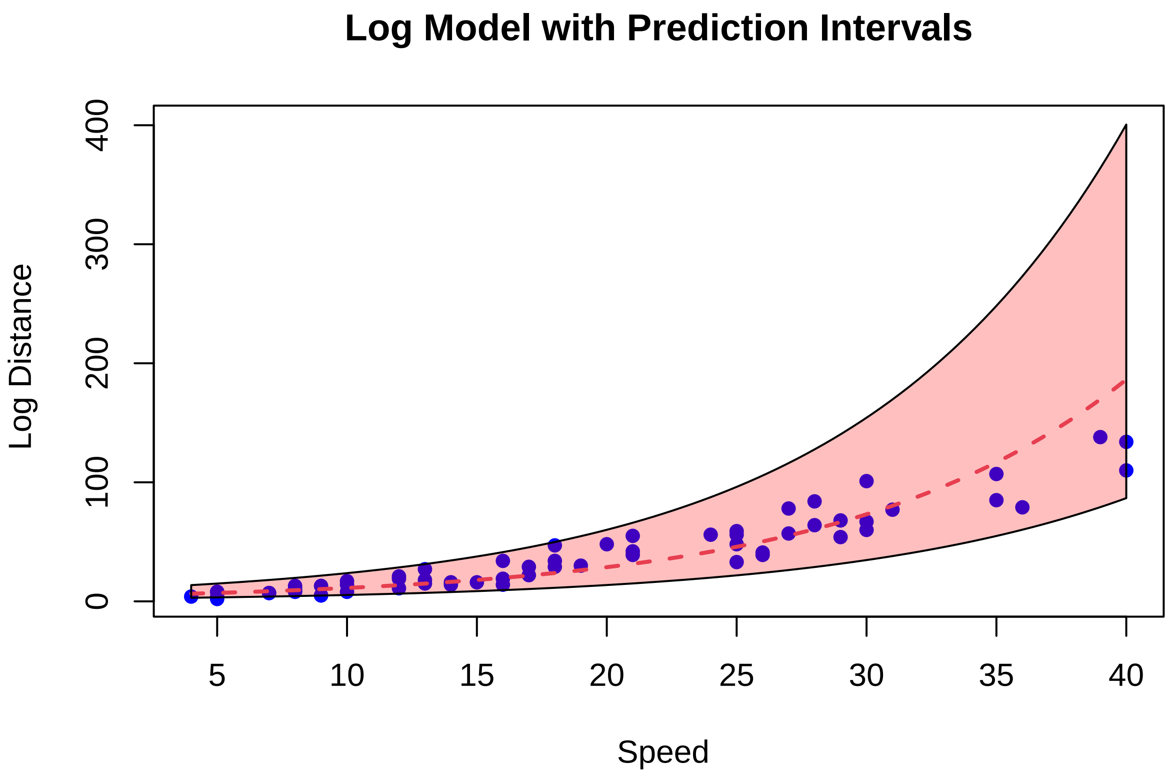


```r
hist(rstandard(log_model), probability = TRUE,main="Histogram of Standardized Residuals for Log Model",
curve(dnorm(x, mean=mean(rstandard(log_model)), sd=sd(rstandard(log_model))), col=2, lwd=2, add=TRUE)
```



```r
# Create a plot with prediction interval

# Define the sequence of speed values for prediction
```

```
x <- seq(min(brake$speed), max(brake$speed), length.out = 100)

# Create a new data frame for predictions, with log-transformed speed
newdata <- data.frame(speed = x)

# Predict the log-transformed distance, along with prediction intervals
predictions_log <- predict(log_model, newdata = newdata, interval = "prediction")

# Convert predictions from log-scale back to the original scale
predictions <- exp(predictions_log)

# Plot the original data (in original scale)
plot(distance ~ speed, data=brake,
     pch = 16, col = "blue", ylim=range(predictions),
     xlab = " Speed", ylab = " Log Distance",
     main = "Log Model with Prediction Intervals")

# Add the fitted line (back-transformed from log to original scale)
lines(x, predictions[, 1], col = 2, lwd = 2,lty=2)

# Add the prediction intervals (back-transformed from log to original scale)
polygon(c(x, rev(x)),
        c(predictions[, 2], rev(predictions[, 3])),
        col = rgb(1, 0, 0, 0.25))
```



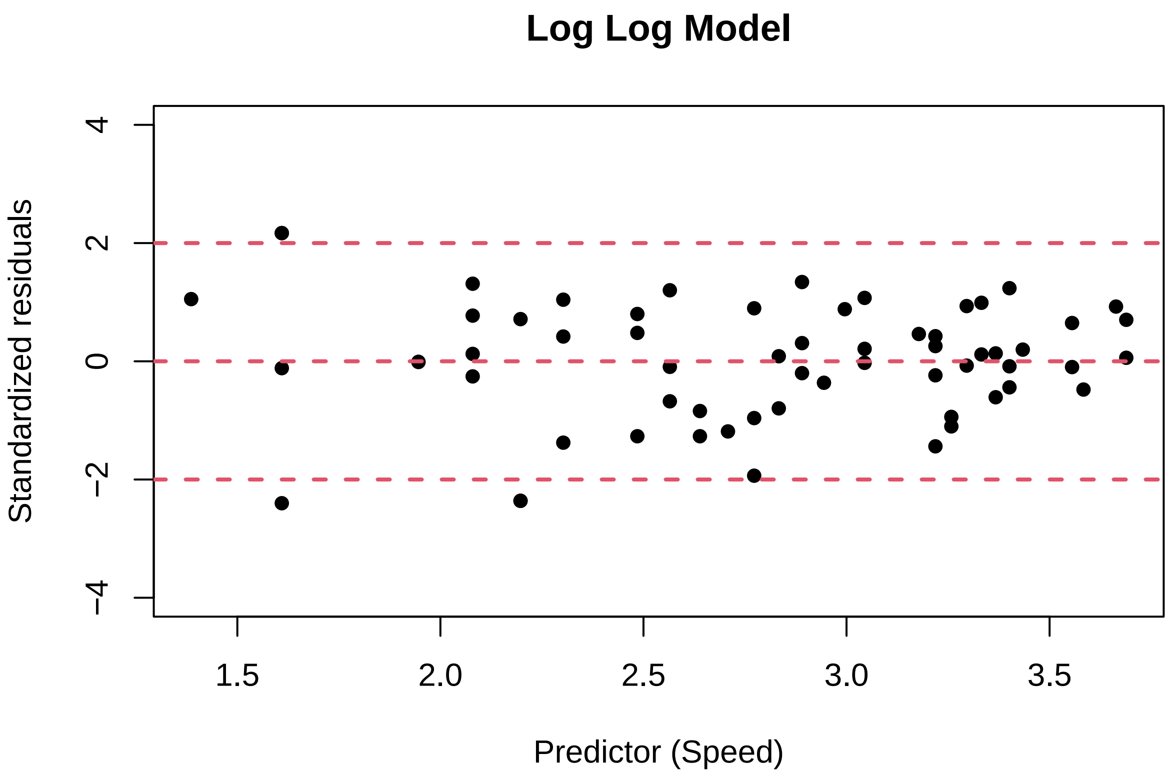
The log model prediction interval is flexible and includes all data points. The standardized residuals are now uniformly and randomly distributed and the distribution of the standardized residuals is relatively normal. The log model is better than the simple linear model because the distribution of the standardized residuals is more normal than the SLR and they all fall within the range $(-2, 2)$ aside from 1 point, which is not the case for the SLR model. The Adjusted R^2 is better than the SLR model without any transformations, which agrees with the results from the analysis of the standardized residuals histograms and plots.

The log model is not better than the polynomial model at inference because of model complexity. The polynomial model is better fit for current data, but for predictive purposes, because of the higher SE of the beta hat coefficient for the predictor, the log model could be better for prediction if verified with other metrics used for measuring predictive power.

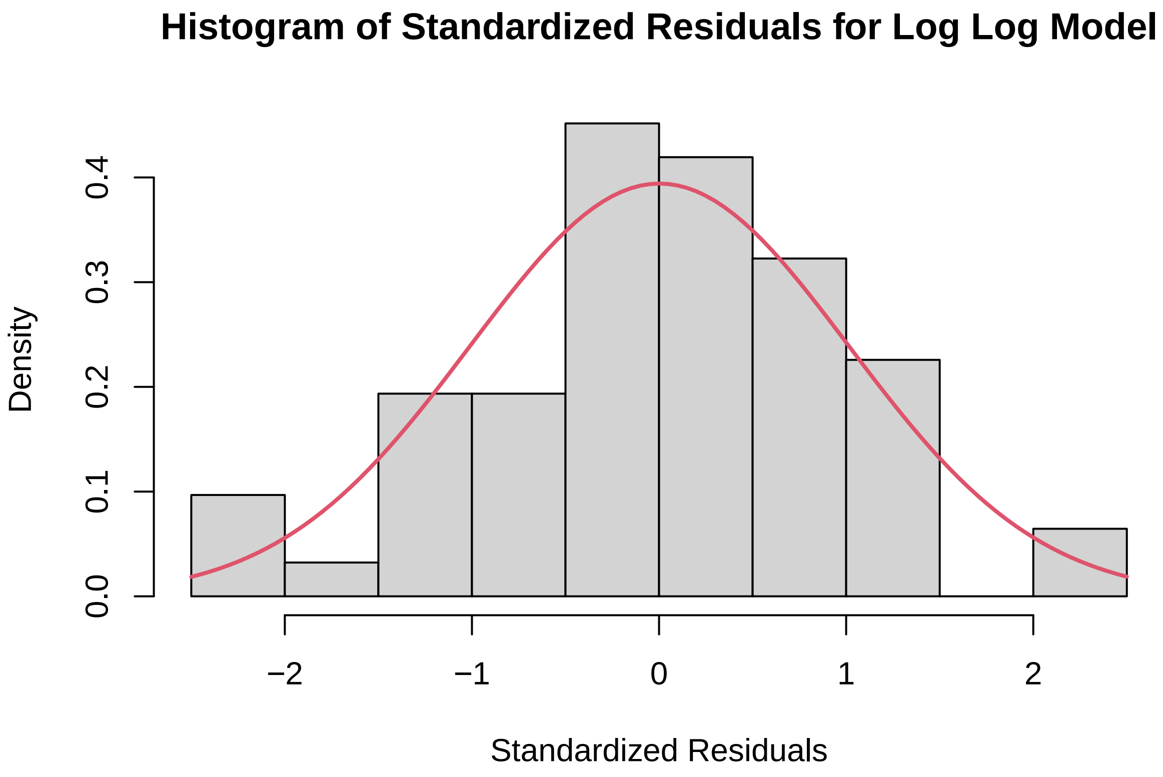
```r
# Fit the model using the log-transformed variables
log_log_model <- lm(log_distance ~ log_speed, data = brake)

# Summary of the log-log model
summary(log_log_model)
```

```
##
## Call:
## lm(formula = log_distance ~ log_speed, data = brake)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.73374 -0.17858  0.02258  0.23500  0.65795
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.10221    0.19003   -5.80 2.64e-07 ***
## log_speed    1.56806    0.06683   23.46  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3158 on 60 degrees of freedom
## Multiple R-squared:  0.9017, Adjusted R-squared:  0.9001
## F-statistic: 550.6 on 1 and 60 DF,  p-value: < 2.2e-16
```

```r
plot(rstandard(log_log_model) ~ log_speed, brake, main= "Log Log Model", ylim=c(-4,4), xlab = 'Predicto
abline(h=0,col=2,lwd=2,lty=2)
abline(h=2,col=2,lwd=2,lty=2)
abline(h=-2,col=2,lwd=2,lty=2)
```

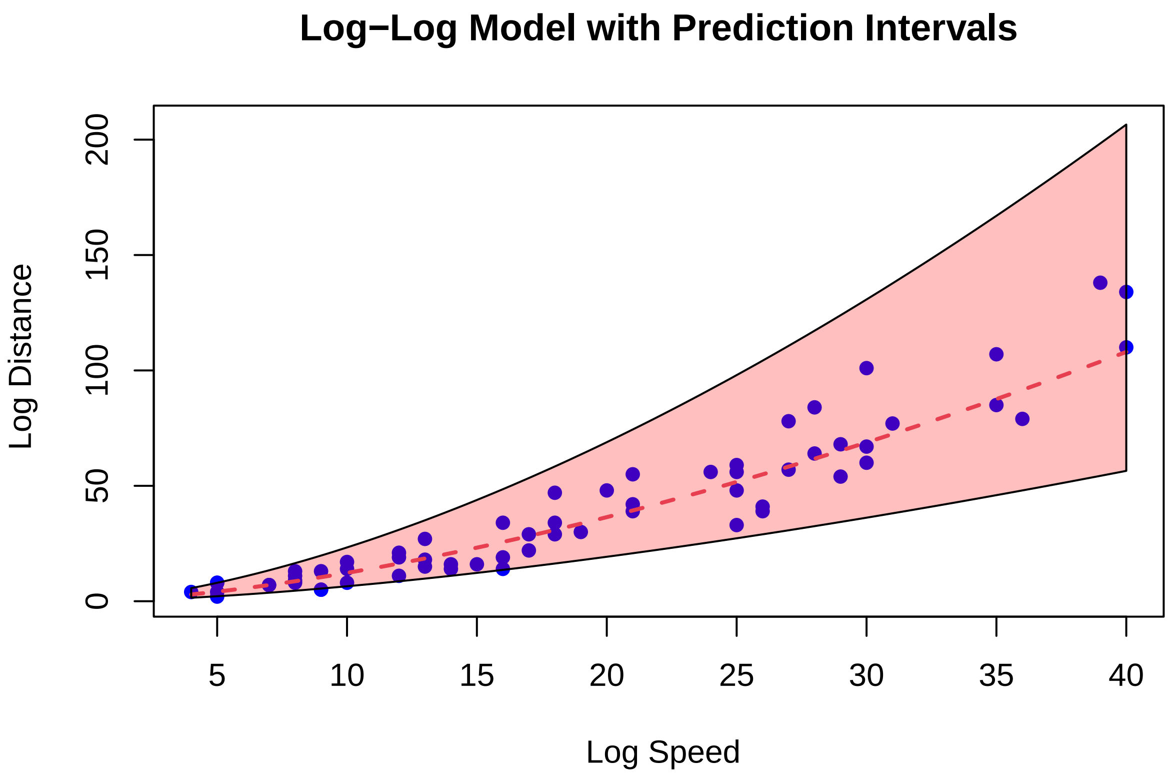


```
hist(rstandard(log_log_model), probability = TRUE,main="Histogram of Standardized Residuals for Log Log
curve(dnorm(x, mean=mean(rstandard(log_log_model)), sd=sd(rstandard(log_log_model))), col=2, lwd=2, add
```



```
# Create a plot with prediction interval

# Define the sequence of speed values for prediction
```

```r
x <- seq(min(brake$speed), max(brake$speed), length.out = 100)

# Create a new data frame for predictions, with log-transformed speed
newdata <- data.frame(log_speed = log(x))

# Predict the log-transformed distance, along with prediction intervals
predictions_log <- predict(log_log_model, newdata = newdata, interval = "prediction")

# Convert predictions from log-scale back to the original scale
predictions <- exp(predictions_log)

# Plot the original data (in original scale)
plot(distance ~ speed, data=brake,
     pch = 16, col = "blue", ylim=range(predictions),
     xlab = " Log Speed", ylab = " Log Distance",
     main = "Log-Log Model with Prediction Intervals")

# Add the fitted line (back-transformed from log to original scale)
lines(x, predictions[, 1], col = 2, lwd = 2,lty=2)

# Add the prediction intervals (back-transformed from log to original scale)
polygon(c(x, rev(x)),
        c(predictions[, 2], rev(predictions[, 3])),
        col = rgb(1, 0, 0, 0.25))
```



Trying out a log-log model as well (not required), gives us a better Adjusted R^2 than just the log model.