Exercise 1

In this exercise, we will be referring the dataset `Darts.csv`.

(a) We were asked to find the best linear regression model for the response variable `Width`. First, we will fit two different linear models, which are obtained from forward step and backward step.

```
model0<-lm(Width ~ Length) # fit a minimal model
modelstep0<-step(model0, scope=formula(Width~Length*Thickness*Name))  # fit a forward step model
model1<-lm(Width~Length*Thickness*Name)  # fit a full model
modelstep1 <- step(model1)   # fit a (backward) step model from a full model
```

where the formula for each model are given as follows

```
> modelstep0
Call:
lm(formula = Width ~ Length + Name)
> modelstep1
Call:
lm(formula = Width ~ Length * Thickness * Name)
```

To choose the best model, we compare the AIC component of the models.

```
> AIC(modelstep0)
[1] 458.9517
> AIC(modelstep1)
[1] 432.4108
```

It follows that we have evidence to choose `modelstep1` over `modelstep0`. Now, we will check the diagnostic plot of the model that we chose.

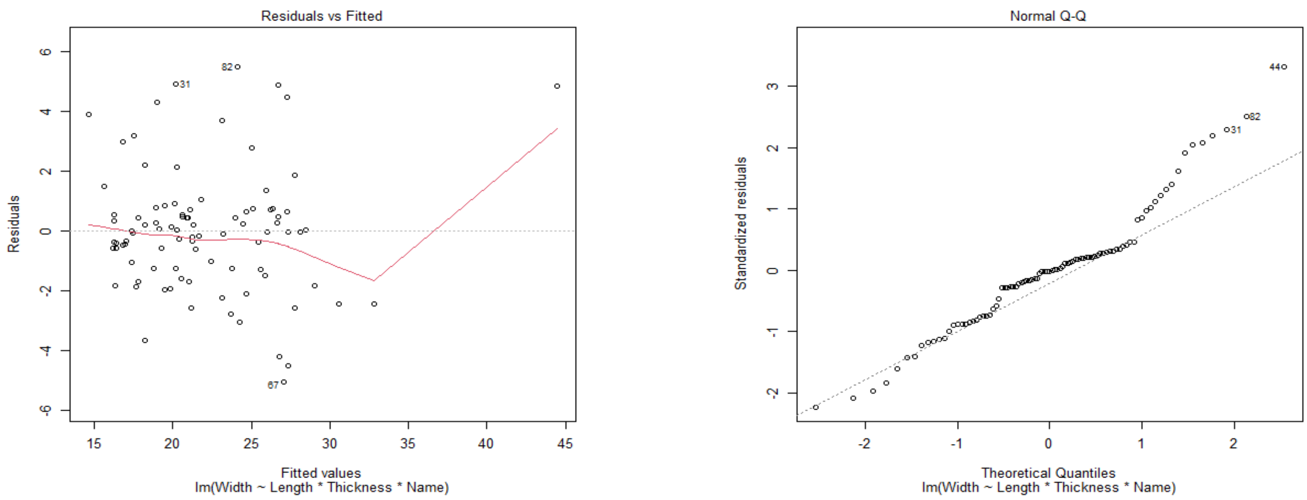

Figure 1: Two of the diagnostic plots of modelstep1

One of the obvious issue highlighted here is the violation of the normality of the errors. We also perform a Box-Cox transformation (as shown in Figure 2) and it suggests taking the logarithmic value of the response variable. Thus, we fit a new linear model.

```
> modelfix <- lm(log(Width) ~ Length * Thickness * Name)
```

From the diagnostic plots of `modelfix` which are shown in Figure 3 below, we see some improvements on the scatter of Residuals vs Fitted, however, it is still **not** obvious from the Normal Q-Q Plot that this model follow our assumption on the normality of the errors.

(b) We will now predict the maximum width in milimetres for the designated input as follows

```
> # predicting a given input form our model fix
> new_dart <- data.frame(Name = "Travis", Thickness = 8, Length = 50)
> width_pred <- predict(modelfix, new_dart)
> exp(width_pred)   # obtain the value for width from log(Width)
```

This returns the value **20.25064 mm**.

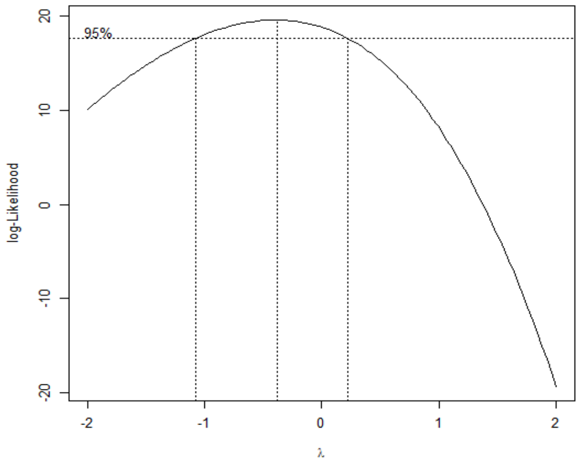

Figure 2: Box-Cox plot for modelstep1

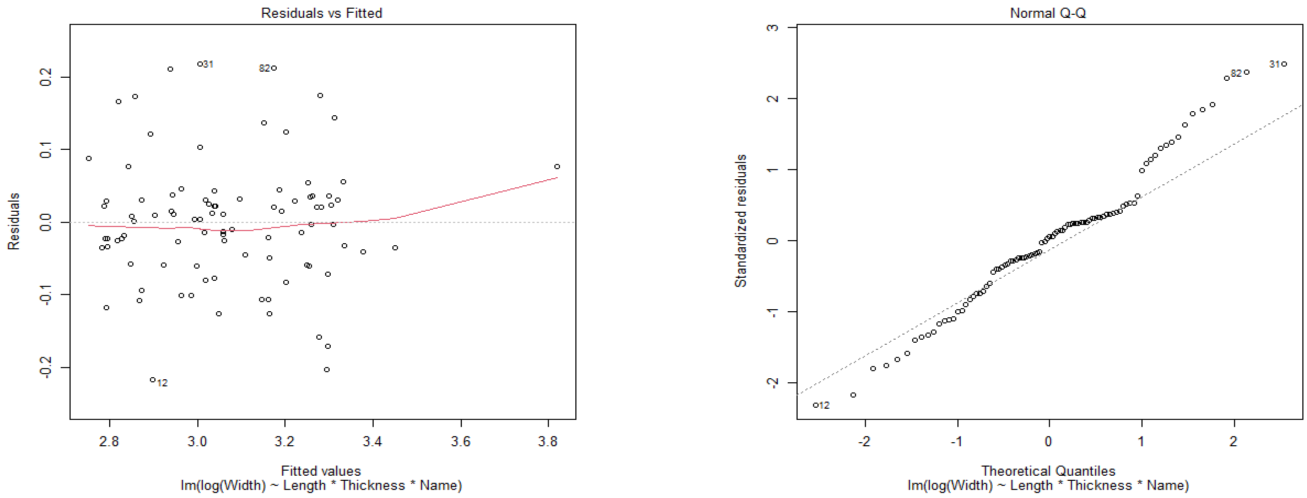

Figure 3: Two of the diagnostic plots of modelfix

Exercise 2

In this exercise we will be referring to the dataset extracted from `optimal.csv`.

(a) We were asked to find the value of `x1` and `x2` that minimise the response variable `y`. First, we fit a model that includes the quadratic terms and also the interaction terms.

```
modelPoly <- lm(y ~ x1*x2 + I(x1^2) + I(x2^2))   # fit a LM with quadratic terms
```

Next, we extract the coefficients of the estimators as follows

```
> beta <- modelPoly$coefficients
> beta
(Intercept)          x1          x2     I(x1^2)     I(x2^2)       x1:x2
    0.86770    -2.81388    -0.08523     1.94370     0.19913    -0.00467
 > b <- beta[2:3]             # vector of coeffiient for linear term.
 > b
          x1          x2
 -2.81387591 -0.08523381
 >
 > B <- matrix(0, 2, 2)    # creating an empty matrix
 > B[1,1] <- beta[4]
 > B[2,2] <- beta[5]
 > B[1,2]<-B[2,1]<-0.5*beta[6]
 > B                                # obtain the quadratic form
             [,1]        [,2]
 [1,]     1.94370    -0.00233
 [2,]    -0.00233     0.19913
```

After computing the vector b and the matrix B, we can now calculate the critical point.

```
x_opt<--0.5*solve(B)%*%b
> x_opt         # minimum point
```

This gives us x1 $= 0.72411$ and x2 $= 0.22250$.

(b) Next, we will provide 95% confindence interval for y when x1 and x2 are as in the previous.

```
> new_opt <- data.frame(x1 = x_opt[1], x2 = x_opt[2])
> x_opt_conf <- predict(modelPoly, new_opt, interval = "confidence", level = 0.95)
> x_opt_conf
        fit        lwr        upr
1  -0.16056   -0.26511   -0.056016
```

Exercise 3

We will load the dataset warpbreaks and fit a generalised linear model for the dataset.

```
> library(datasets)
> data("warpbreaks")
> glm1<- glm(breaks ~ wool*tension,family=poisson, data = warpbreaks)
```

(a) Under the function glm, we construct a generalised linear model to observe the response variable breaks of the dataset warpbreaks. Here, we specify poisson as the probability distribution together with the log function as the default link function.

Let Y be the number of breaks, then the model glm1 follows

$$Y_i \sim \text{Pois}(\mu_i)$$

where Y_i independent for $i = 1, 2, ..., n$, and the Poisson parameter (mean) μ_i is related to the estimators by the link function

$$\boxed{log(\mu_i) = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{3i} + \beta_4 x_{1i} x_{2i} + \beta_5 x_{1i} x_{3i}}$$

where

$$x_{1i} = \begin{cases} 1 & \text{, if wool B} \\ 0 & \text{, otherwise} \end{cases}$$

$$x_{2i} = \begin{cases} 1 & \text{, if tension M} \\ 0 & \text{, otherwise} \end{cases}$$

$$x_{3i} = \begin{cases} 1 & \text{, if tension H} \\ 0 & \text{, otherwise.} \end{cases}$$

The estimates of the parameters β_j for $j = 0, ..., 5$ are given below.

```
> glm1$coefficients
   (Intercept)          woolB        tensionM        tensionH  woolB:tensionM
       3.79674       -0.45663        -0.61868        -0.59580         0.63818
 woolB:tensionH
       0.18836
```

and also **the estimate for the dispersion parameter ϕ is equal to** 1.

(b) We were asked to calculate the expected value for the number of breaks when we have the type of wool A and tension M. Firstly, we store a new data frame as follows.

```
try_data <- data.frame(wool = "A", tension = "M")
```

Next, we use the function predict() to obtain the value of $\log \mu_i$ for try_data and apply the exponential function on the value to yield the expected value of the number of breaks.

```
ypred<-predict(glm1, try_data)
exp(ypred)
```

This returns the value **24**. Note that the result is the same if we calculated manually from the dataset.