# *STATISTICAL ANALYSIS USING R STUDIO*

Introduction To Statistical Methods For Data Science (Course Work)

*Subject Code : 7089CEM*

*Student Name : Roshan Dhanasekaran*

*Student ID : 12135637*

*Roshan Dhanasekaran*

## SUMMARY

This report focuses on choosing the best regression model which describes the appropriate relationship between the input MEG and output MEG. The first part of this report focuses on preliminary data analysis where we intend to plot the data and find correlations and outliers with the help of correlation and box plots. We also find the distribution of the data by plotting a histogram with the density curve and time series plot to find the correlation against time.

In the second part of the report, we intend to find the best model out of the 5. The given models are tested by using various techniques like model estimation, Sum of square errors, log-likelihood, Akaike information criterion (AIC) and Bayesian information criterion(BIC) and plotting a Q-Q plot to find the error distribution. This process can determine the best model out of 5. Finally, the models are tested with testing data and the model prediction to determine the errors and plot the confidence interval with error bars.

In the final part of the report, we use approximate Bayesian computation(ABC) to compute the posterior distribution of the best regression model and plot the marginal and joint distribution of the two parameters from the selected model. The results have shown that emotional narration does increase brain repones.

NOTE: All the codes used for the assignment are present in the appendix section.

# TABLE OF CONTENTS

## INTRODUCTION

This report was written as a part of statistics module coursework. The project focuses on statistics and how statistics can help in evaluating a machine learning model and also analyse statistical information from the data itself in this coursework we evaluate 5 different polynomial models and using the right techniques we find the best model. All the above-mentioned testing and plotting were done in the R studio. Packages used (base, GGPlot )

The main aim of the project is to find whether there is a change in brain response when there is a voice change from normal to emotional, this was finally achieved after finding the best regression model. To achieve this the following objective was set :

1) Using R studio to load and process the data.
2) Explore and analyse the data using appropriate techniques.
3) Finding the best model using model estimation
4) Using approximate Bayesian computation to estimate the parameters
5) Visualise the outcomes

## BACKGROUND & DATASET

The MEG signal is obtained from the neuronal activity in the brain the MEG signal can show the absolute neuronal activity in the brain with the help of the data we can determine the brain activity from the levels of the signal.

The dataset is separated by 'X', 'Y' and time. The 4 major components of the dataset are x1 input MEG, and x2 define the voice. Y data is the output signal and the time has the time recorded.

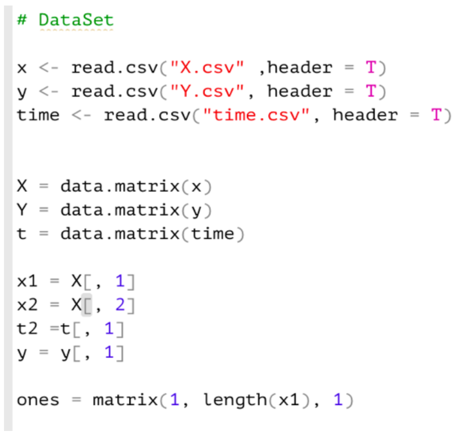
| Variable name | Data Type | Description |
|---|---|---|
| X1 | Double | Input Signal |
| X2 | NUM | Voice |
| Y | Double | Output Signal |
| Time | Date/Time | Time interval |

*Table 1. Dataset*

Since we have basic information about the data and the dataset we move on the plot the data and do EDA to understand the data better which brings us to the next part of the report.

## IMPORTING DATA

Although there are several ways to get the data inside R studio, we have used the read.csv function to import the data inside the environment, we do this with the X, Y, and Time data separately and convert the data into a matrix with the *data.matrix()* function with which we can access the columns of the dataset separately. Refer appendix 6.1



```
# DataSet

x <- read.csv("X.csv" ,header = T)
y <- read.csv("Y.csv", header = T)
time <- read.csv("time.csv", header = T)


X = data.matrix(x)
Y = data.matrix(y)
t = data.matrix(time)

x1 = X[, 1]
x2 = X[, 2]
t2 =t[, 1]
y = y[, 1]

ones = matrix(1, length(x1), 1)
```

*Figure 1. Importing dataset*

As we have imported our data into our R environment we go ahead and perform preliminary data analysis which brings us to the first part of our report.

## PART -1

## 1. Preliminary Data Analysis

### 1.1 Time Series Plot

The time series plot is used to compare the input MSG and output MSG with time therefore we can see if there are any significant changes in the output signal after the voice changes to an emotional voice after 10 seconds. Refer to appendix 6.2

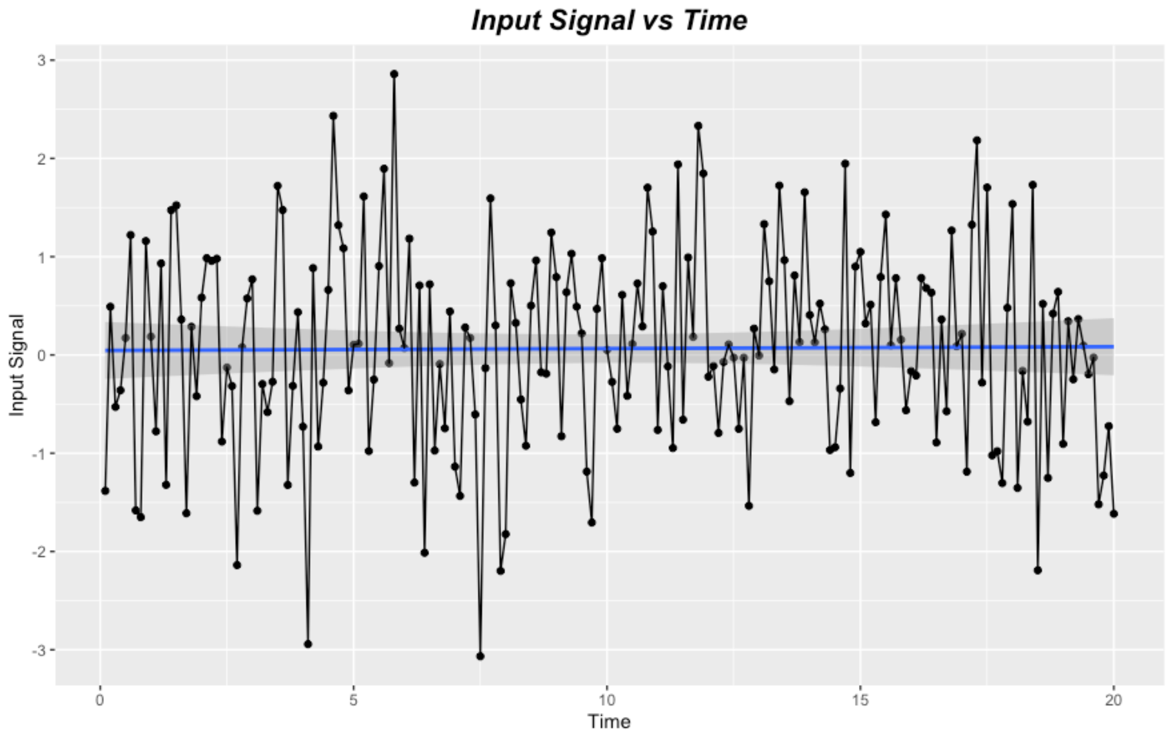


*Figure 2 . Time series plot for input signal against time*

In figure 2 there is not much we can take as inference since there is no differentiation between the voice change and we also need to know when the voice change occurred.

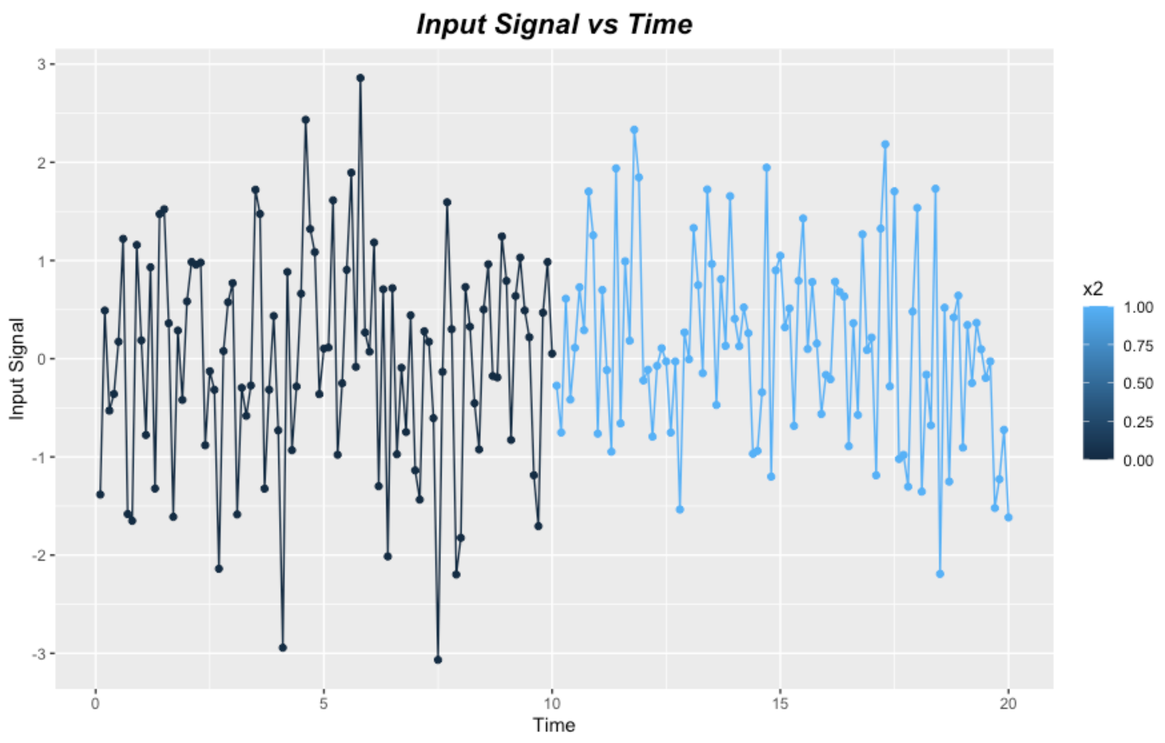


*Figure 3. Time series plot for the input signal and representing the voice change(x2)*

As shown in figure 3 we can see the change in the colour of the lines as the voice change by doing this we can have a better idea of how the data is separated and this plot also shows that there is some minor change when the voice is the change from neutral to emotional. Now let's go ahead and plot the output signal and time

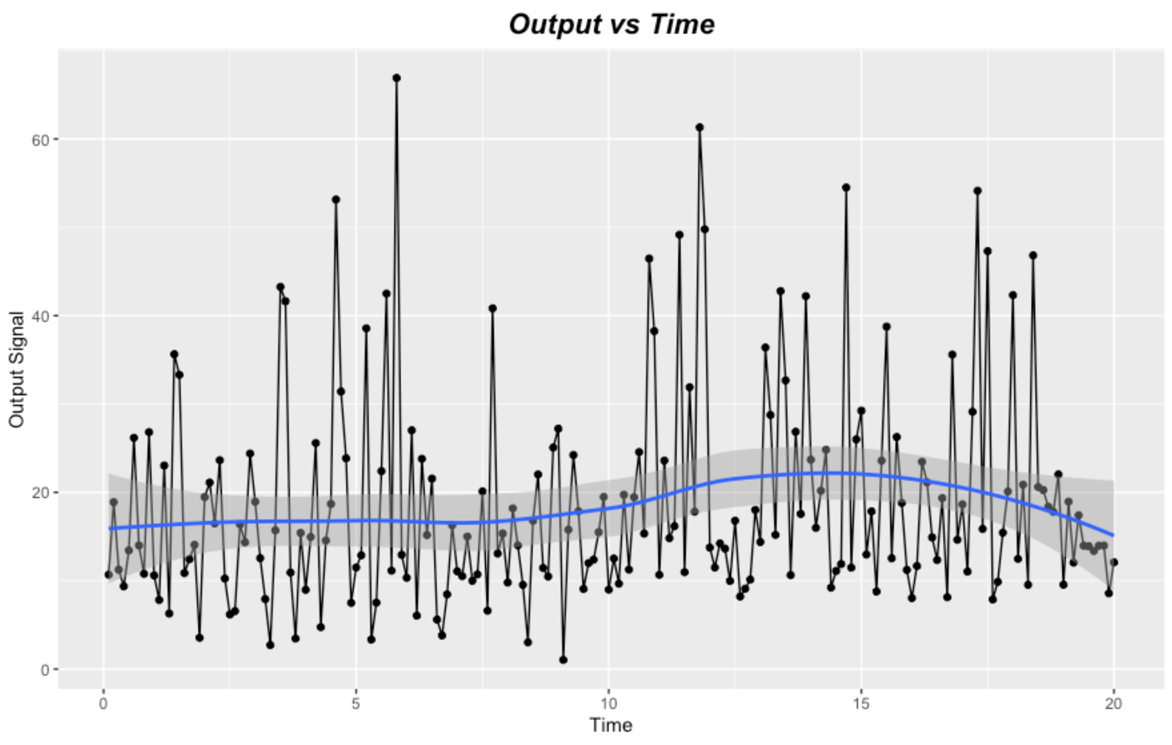


*Figure 4. Time series plot for output signal and time.*

Figure 4 shows the readings of the output signal against time now let’s plot an output signal plot against time separating the voices.

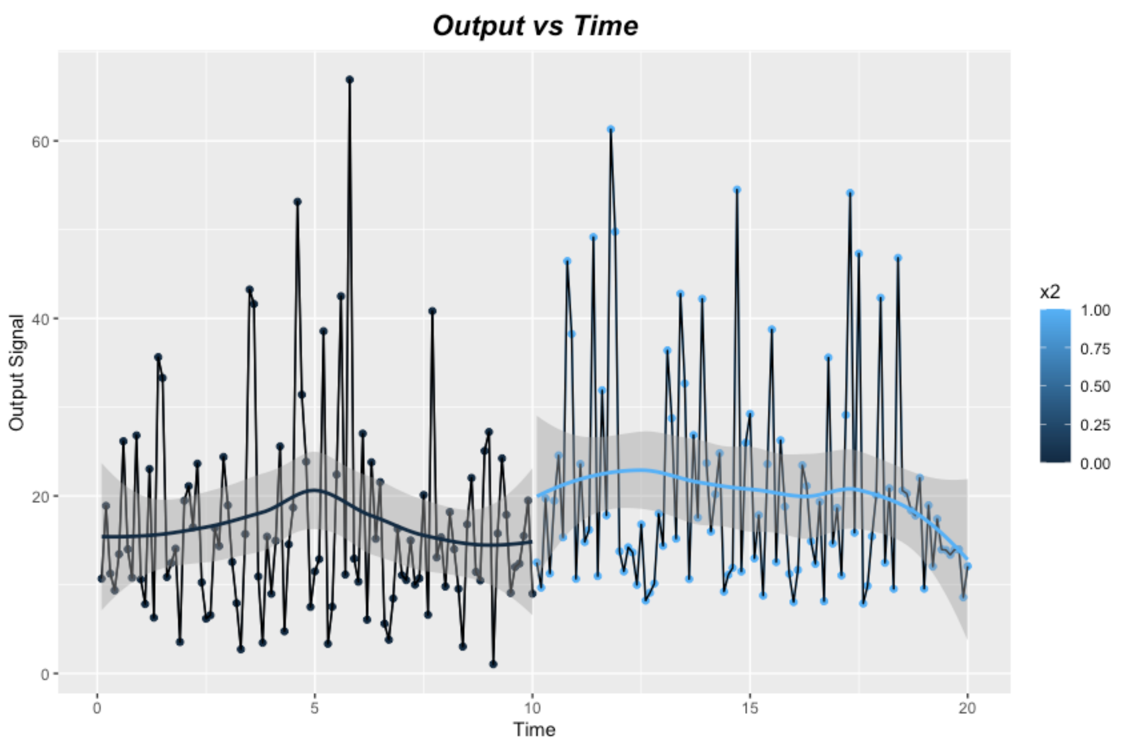


*Figure 5. Time series plot for the output signal and representing the voice change(x2)*

Since we have found the relation of the data against time now let’s find the distribution for each input and output signal.

### 1.2 Distribution

Finding The distribution of the data is very important for several reasons first and foremost we need to find whether the data is normally distributed this can be archived by finding the mean and the median values and plotting the data on a histogram example figure 6. The graph shows that the data is normally distributed and a lot of data points fall in the centre. There is a slight variation between the mean and the median which means there are some outliers. Refer to appendix – 6.3

| Min | Mean | Median | Standard deviation | Variance | MAX |
|---|---|---|---|---|---|
| -3.06664 | 0.06506 | 0.10933 | 1.042515 | 1.086838 | 2.85898 |

*Table 2 . Input signal summary*

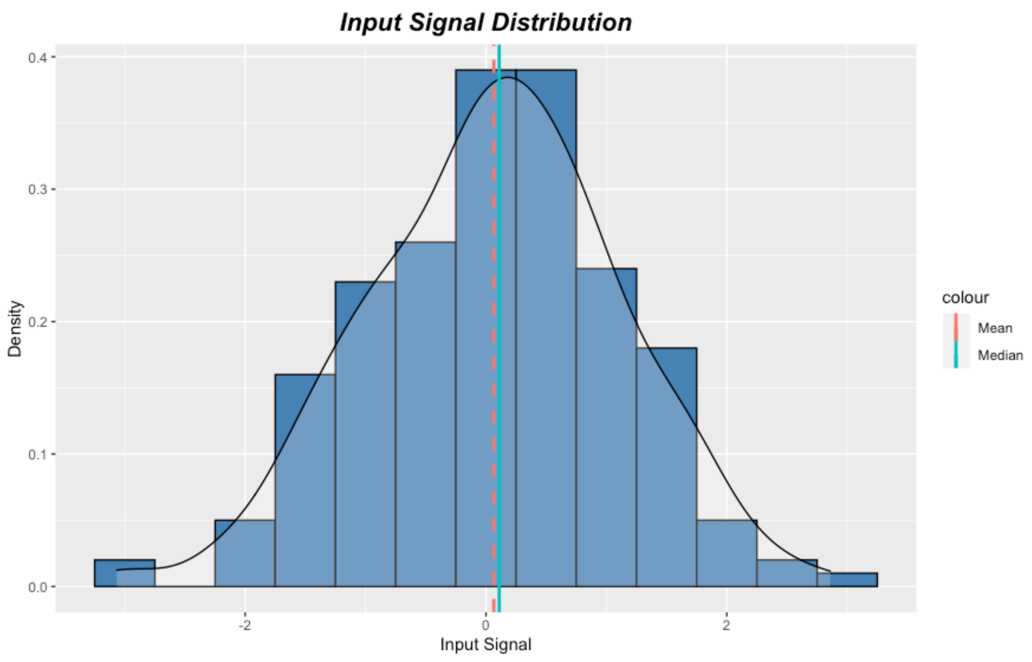


*Figure 6. Distribution of Input Signal*

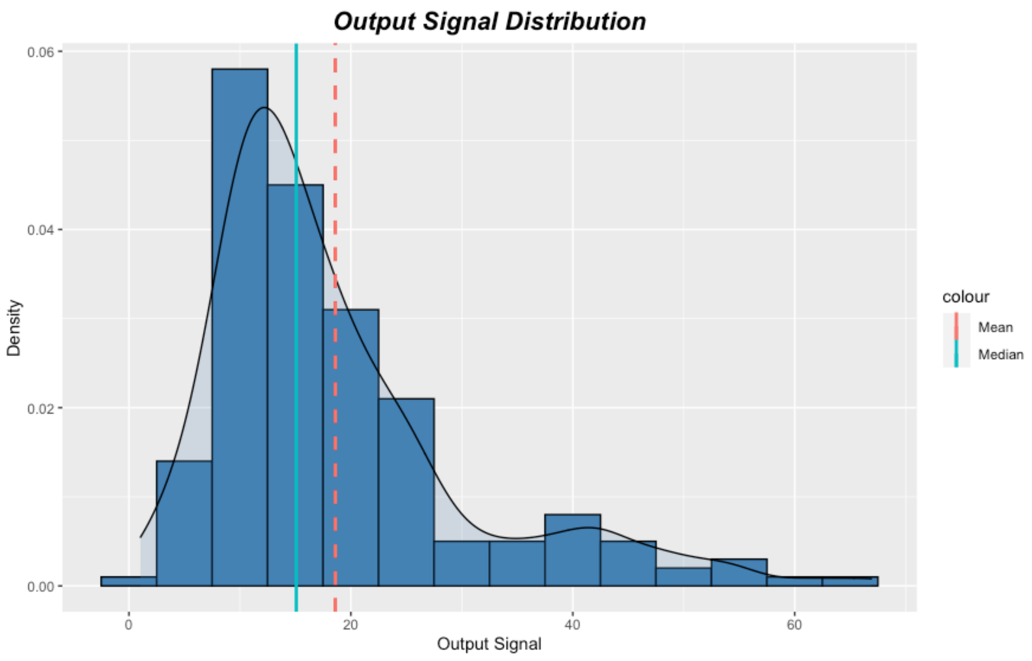


*Figure 7 . Distribution of Output Signal*

| Min | Mean | Median | Standard deviation | Variance | Max |
|---|---|---|---|---|---|
| 1.041 | 18.604 | 15.079 | 11.87971 | 141.1275 | 66.900 |

*Table 3 Output signal summary*

The histogram is right-skewed which means most of the values fall on the left side of the histogram whereas on the right side the graph frequency obtained is less than compared with the right side. Figure 7 shows the distribution of the output signal as the plot clearly shows the distribution is not normal. We can also find that the mean line and the median line are not close to each other this is an indication of outliers in the data. We can find the outliers in the data by plotting a box plot which will give us a better idea of the data.

### 1.3 Box Plot

Using a box plot we can determine the number of outliers which fall away from the upper limit and the lower limit. we plot the box plot by finding the Q1, Q3, mean and the upper bound and the lower bound. Refer to appendix 6.4

| Lower Whisker | Q1 | Q2 (Median) | Q3 | Upper Whisker |
|---|---|---|---|---|
| 1.04134 | 10.82458 | 15.07914 | 23.24714 | 41.62941 |

*Table 4 Box Plot Stats output signal*

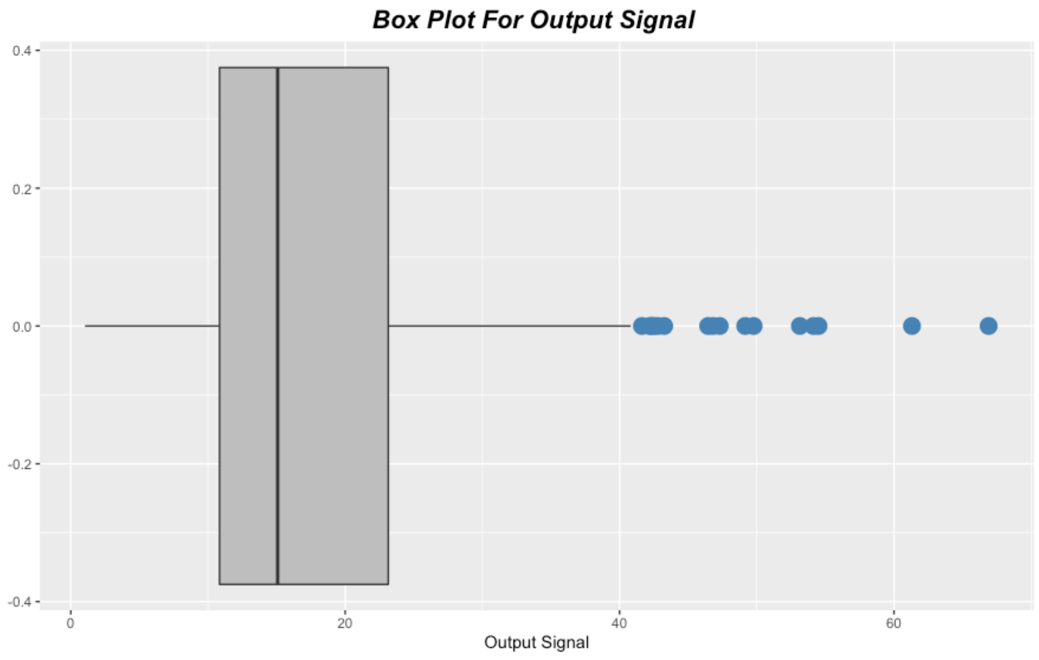


*Figure 8. Box plot for output signal*

The box plot for the output signal shows there are several outliers in the data and that lead to variation between the mean and the median values in the distribution plot. Now let's have a look at the box plot for the input signal.

The below figure 9 shows there are two outliers found in the dataset of the input signal which justifies the mean and median variation in the histogram.

| Lower Whisker | Q1 | Q2 (Median) | Q3 | Upper Whisker |
|---|---|---|---|---|
| -2.1975779 | -0.6817331 | 0.1093263 | 0.7596919 | 2.8589840 |

*Table 3 Box Plot Stats for input signal*

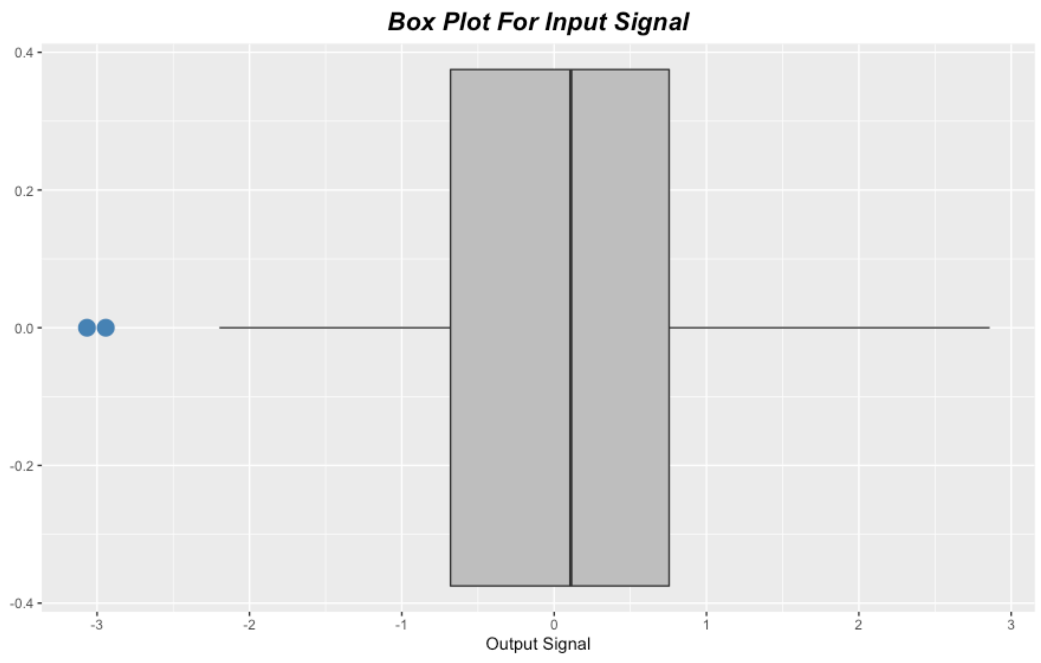


*Figure 9 Box plot for input signal*

Although there are several methods to eliminate the outliers one of the easiest methods is using square root which can eliminate some outliers and then give a normal distribution of the data, in this assignment we tend to keep our outliers and carry on with the operations.

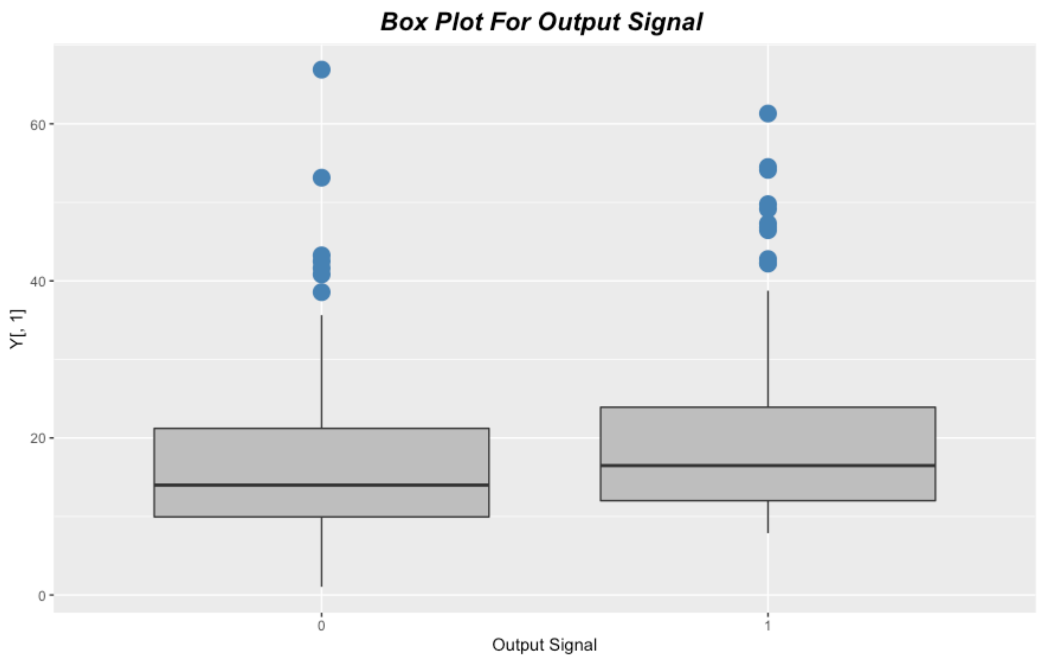


*Figure 9 Box plot for output signa*

The above box plots confirms that there is slight increase in MEG signal, but it may not always be the case. We can confirm this by plotting correlation plots and scatter plots and find weather there is any changes in brain signal as the voice change.

### 1.4 Correlation

We intend to find the correlation between the input signal and the output signal using the parson's correlation. By using the *corr()* function in R we can get the basic information of the correlation values like the p-value t value, confidence interval and the correlation itself. . Refer to appendix 6.5

| Correlation between Input and output signal using Pearson's correlation | | | | | |
|---|---|---|---|---|---|
| T value | P-Value | 95 percent confidence interval | | Correlation Estimate | Df |
| 16.734 | < 2.2e-16 | 0.7010681 | 0.8173168 | 0.7653645 | 198 |

*Table 4 Correlation estimate between input and output signal*

The above table describes the correlation between the input and output signal first and foremost we can see the correlation estimate of 0.7 which is considered a moderate correlation we can also see the p-value is very small according to the rules tells us to reject the null hypostasis. We can also see the 95th percent confidence interval and out p values fall outside the confidence interval which is below 0.701 on the left-hand side.

Let's understand the data by plotting a scatter plot and finding the correlation between the input and output data.

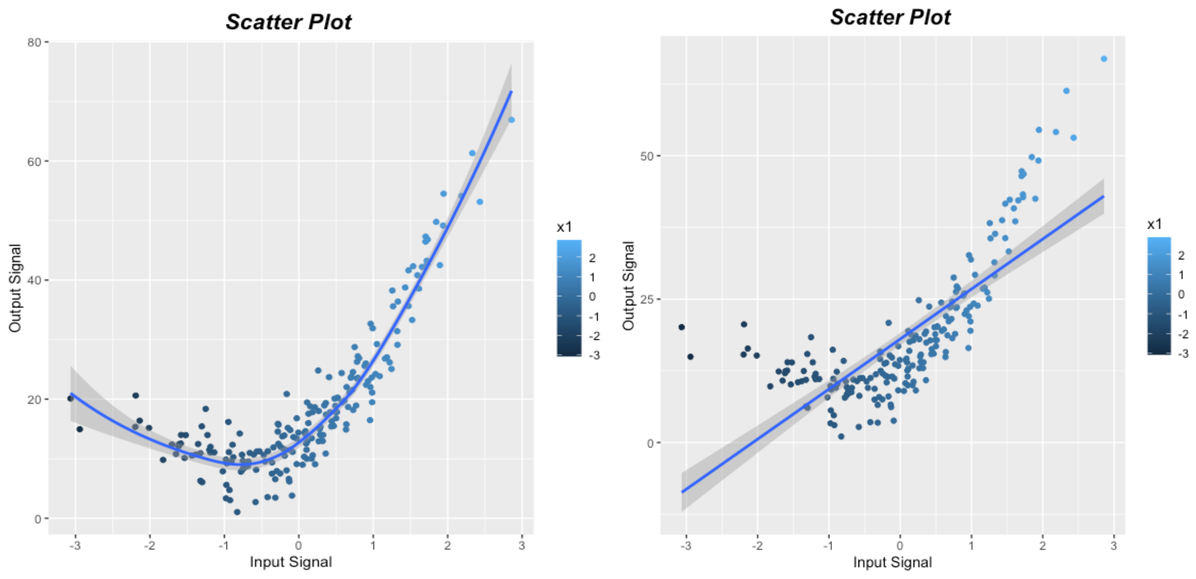


*Figure 10 Scatter plot for input vs output signal*

The above graph was plotted to find the correlation between the input signal and the output signal although there is not much inference that can be taken from the graph. We move on to plot the point when the voice changes therefore the points will have a different colour and let's find out if we can get any inference from the graph.

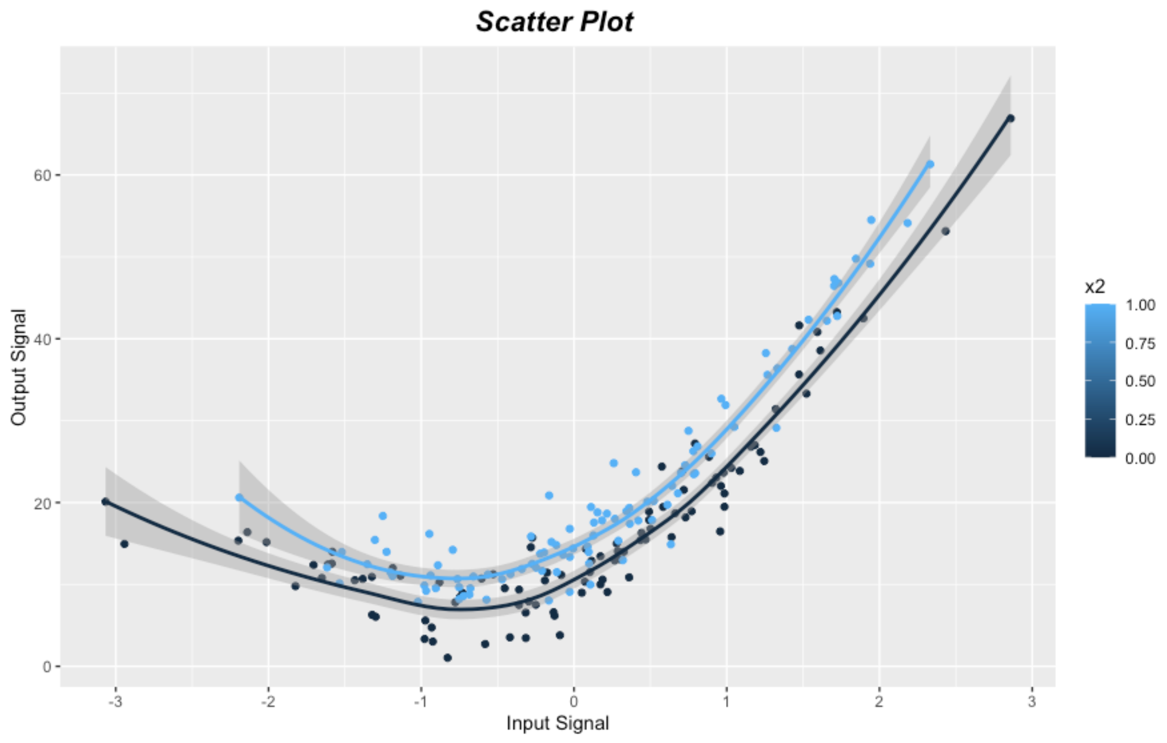


*Figure 11 Scatter plot separating the voices*

This plot gives us a better understanding of the points and the distribution of the points when the voice is changed. The plot indicates the light blue as the emotional voice and the bark blues as the emotional voice. The smooth line is by finding the least square errors and plotting to find the best fit line possible.

## Sub-Conclusion

- Upon doing data analysis on the dataset we can find some correlation between the data especially the input and the output signals this was concluded by the Pearson's correlation
- Certainty there are some outliers in the input and the output signal which was confirmed by the box plot.
- We were also able to check whether the data is normally distributed by plotting the histograms it was found the input signal was normally distributed and the output signal was not.

**PART – 2**

## 2. Regression Modelling

In this part of the report, we test the five candidate models and select the best model based on a few criteria. To be able to do this, we need to estimate the model parameters, find the residual errors of each model, and also compute the log-likelihood, and the AIC and BIC. The best model is the one that has the best scores on all the criteria mentioned above. We show the models below:

1. $y = \theta_{bias} + \theta_1 x_1^3 + \theta_2 x_1^5 + \theta_3 x_2 + \varepsilon$
2. $y = \theta_{bias} + \theta_1 x_1 + \theta_2 x_2 + \varepsilon$
3. $y = \theta_{bias} + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^4 + \theta_4 x_2 + \varepsilon$
4. $y = \theta_{bias} + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^3 + \theta_4 x_1^5 + \theta_5 x_2 + \varepsilon$
5. $y = \theta_{bias} + \theta_1 x_1 + \theta_2 x_1^3 + \theta_3 x_1^4 + \theta_4 x_2 + \varepsilon$

### 2.1 Estimating Model Parameters

For the given five parameters we intend to do parameter estimation. First and foremost we need to assign the theta($\theta$) for x1 = input signal, x2= voice change, theta bias is a row of 1's we intend to create 200 values, y = output signal. After assigning the values we use the cbind function to combine the values since we are using polynomial regression we tend to take the power of the theta values. With our model created we further move to estimate the parameters by finding the thetahat($\boldsymbol{\theta}$) by deploying our model in the formula $\boldsymbol{\theta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$. X denotes the model and y denotes the output signal. Refer appendix 6.6

| Theta Hat Values For Each Candidate Model | | | | |
|---|---|---|---|---|
| **Model - 1** | **Model - 2** | **Model - 3** | **Model - 4** | **Model - 5** |
| 16.9357507 | 16.635590 | 10.1736961 | 10.85328261 | 14.1317254 |
| 5.1712728 | 8.629107 | 8.5521613 | 9.35397322 | 8.0794581 |
| -0.5373253 | 2.813987 | 6.2469171 | 4.64696205 | 0.3144501 |
| 2.2020179 | N/A | -0.2829631 | -0.57792162 | 0.5592050 |
| N/A | N/A | 4.1598833 | 0.07479626 | 3.9681455 |
| N/A | N/A | N/A | 4.34321644 | N/A |

*Table 5 Estimated values*

These are values of each parameter it also shows how much the parameter is contributing to the model this information will be useful later on in the CW. Now we need to find the RSS(Sum of square errors ) for each model value, which brings us to the next task of the CW

### 2.2 Finding the Residual Error

In this task, we are finding the sum of square errors for each model for which we first find the difference between the actual values and the predicted values and then sum the values and square them all together to get rid of the negative value the formula for RSS is given as

$$rss = \sum_{1}^{n} (y - y_{hat})^2$$

| **Models** | **RSS** | **Least RSS Order** |
|---|---|---|
| Model 1 | 11825.42 | 5 |
| Model 2 | 11238.95 | 4 |
| Model 3 | 1636.168 | 1 |
| Model 4 | 1902.063 | 2 |
| Model 5 | 4928.312 | 3 |

*Table 6 RSS values for each model*

From the above table, we can see that model 3 has the lowest RSS values which says it is a better model comparatively but before jumping to a conclusion we can further find the log-likelihood of the model and check the values. Refer  appendix 6.7

### 2.3 Finding Log-Likelihood

In this task, we find the log-likelihood for all the variables but why log-likelihood? We can just find the likelihood function but by doing likelihood, when there are several elements the output for each data point will be very less and that can sometimes be very difficult to compute by taking the log of the likelihood for each data point we can product turns into sum and all the value are much bigger and makes more sense. If we plot likelihood and log-likelihood we can eventually see a bell curve on both therefore both methods are correct Refer to appendix 6.8. The formula for log likelihood is given as

$$\log L(\theta|D) = \sum_{i=1}^{n} \log p(x_i|\theta)$$

| **Models** | **Log-Likelihood** |
|---|---|
| Model 1 | -691.7579 |
| Model 2 | -686.6713 |
| Model 3 | -493.9684 |
| Model 4 | -509.0267 |
| Model 5 | -604.2324 |

*Table 7 Log likelihood*

## 2.4 Akaike information criterion (AIC) and Bayesian information criterion (BIC)

The outputs after calculating AIC and BIC are as shown below. The results of the output are discussed in detail in section 2.6. Ref to appendix 6.9

| **Model** | **AIC** | **BIC** | **Order** |
|---|---|---|---|
| Model 1 | 1391.516 | 1404.709 | 5 |
| Model 2 | 1379.343 | 1389.238 | 4 |
| Model 3 | 997.9368 | 1014.428 | 1 |
| Model 4 | 1030.053 | 1049.843 | 2 |
| Model 5 | 1218.465 | 1234.956 | 3 |

*Table 8 AIC and BIC*

The table indicates that model number 3 is the better model. We can also plot the error distribution and see if the model predictions are good enough.

The formula for AIC and BIC are given as :

$AIC = 2k - 2\ln(L_{max})$

$BIC = kln(n) - 2\ln\ (Lmax)$

## 2.5 Distribution Of Model Prediction Errors

In this task, we plot the distribution of the error with a QQ plot and histogram. We intend to find the error distribution to be a normal/gaussian distribution. This is to mainly ensure our noise added into our dataset is a gaussian noise and if that is true our model predictions are good. Refer  appendix 6.10

### Model 1

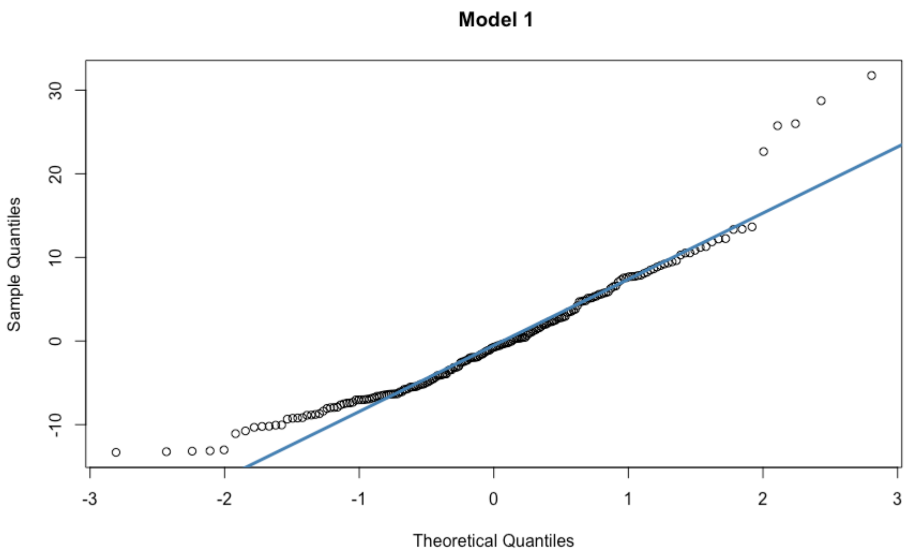


*Figure 12 QQ plot for model 1predicted errors*

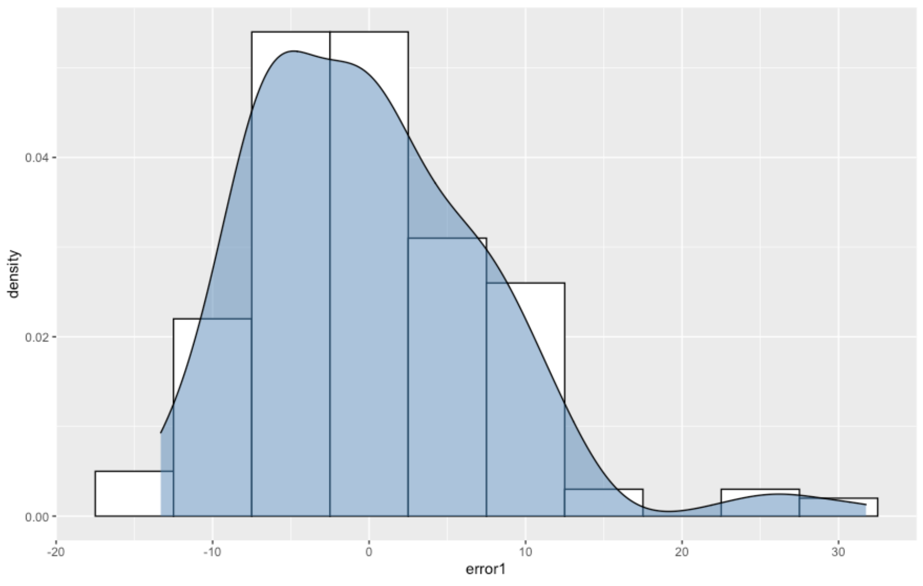


*Figure 13 histogram for model 1 predicted errors*

**Model 2**

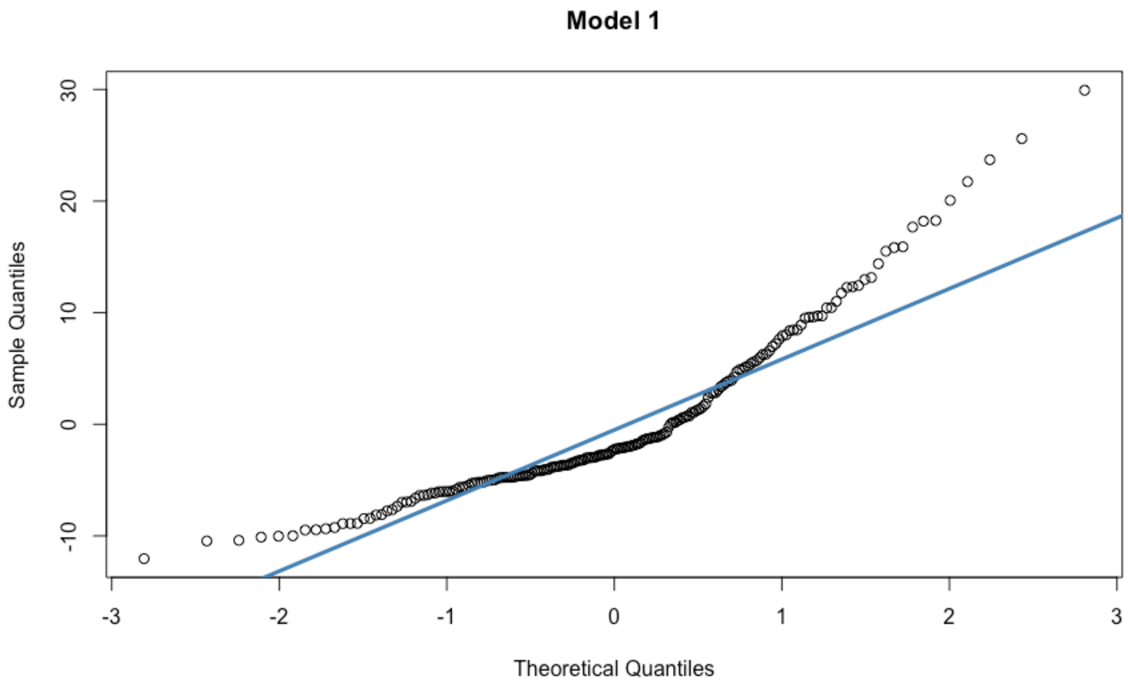


*Figure 14 QQ plot for model 2predicted errors*

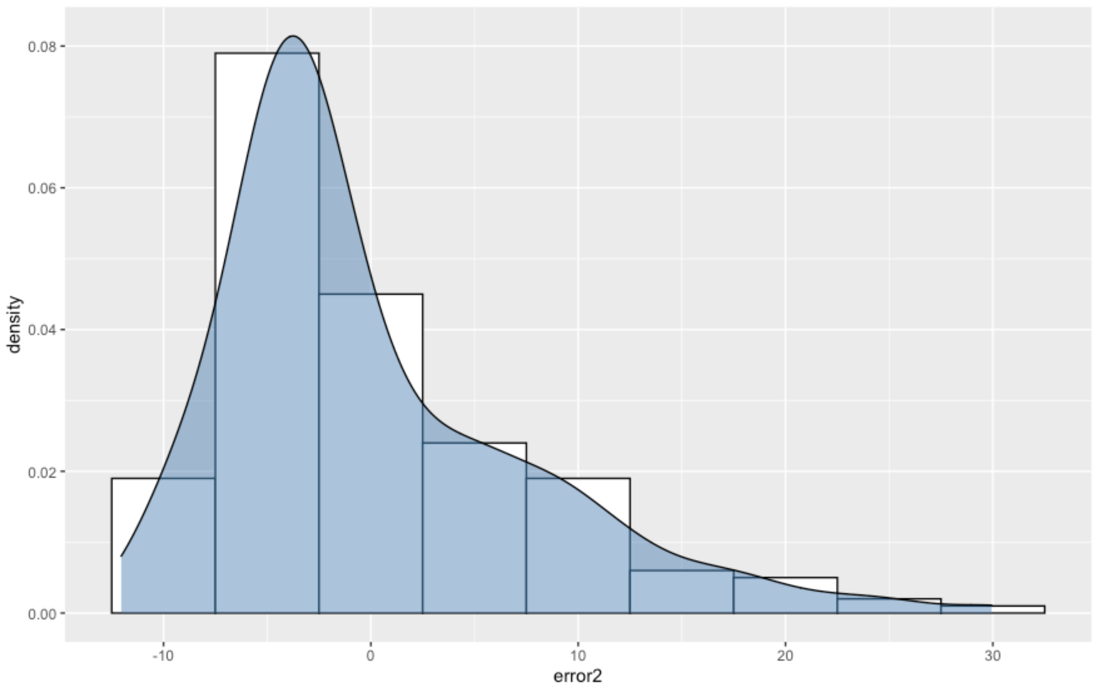


*Figure 15 histogram for model 2 predicted errors*

## Model 3

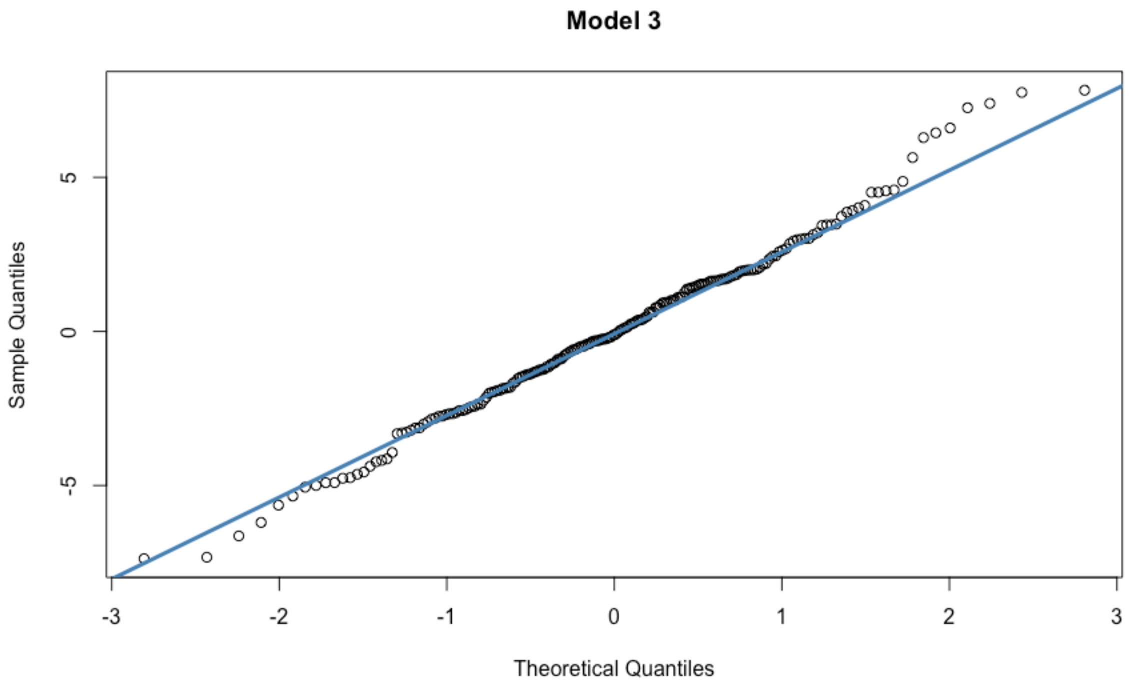


*Figure 16 QQ plot for model 3 predicted errors*

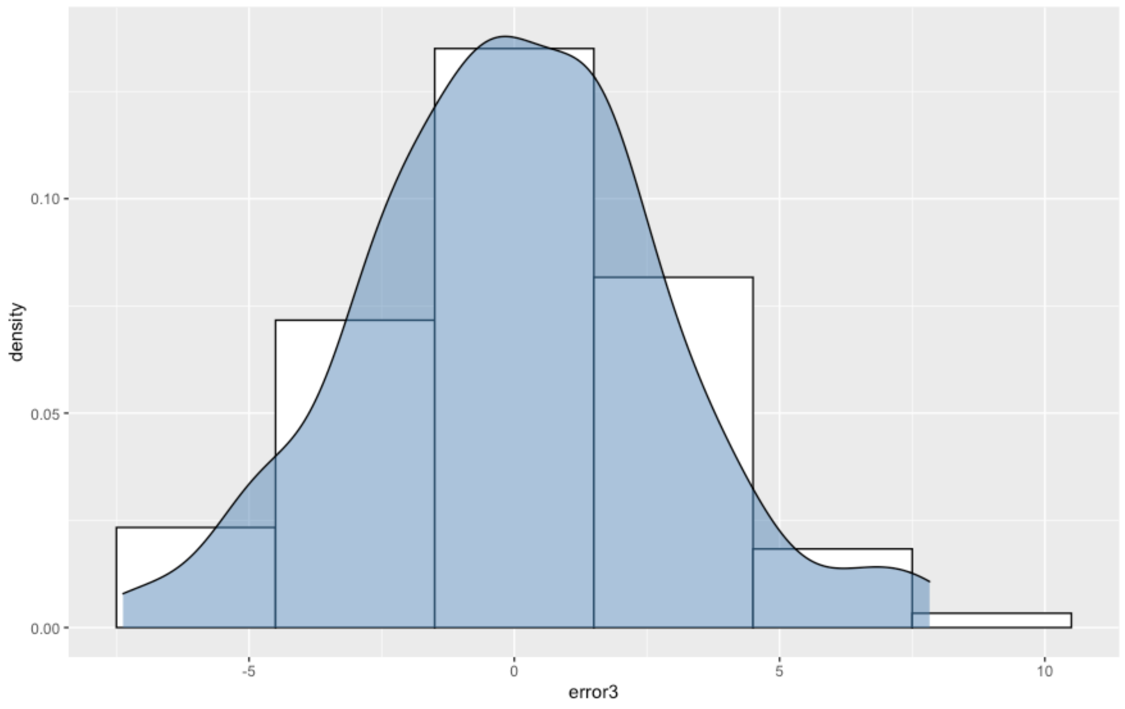


*Figure 17 histogram for model 3 predicted errors*

As we can see the errors in model 3 are more normally distributed this means that the errors in this model can be accepted.

**Model 4**

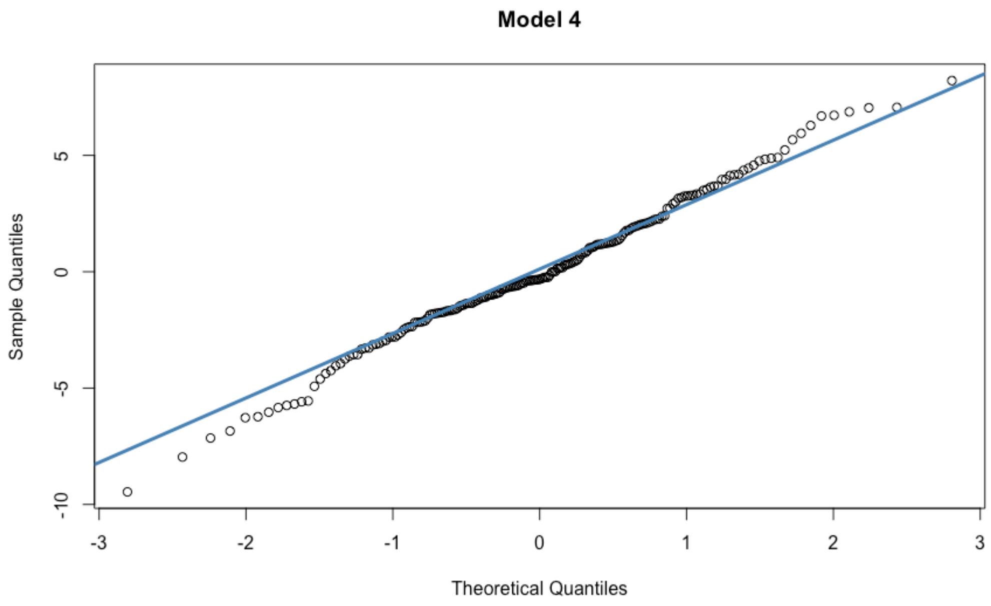
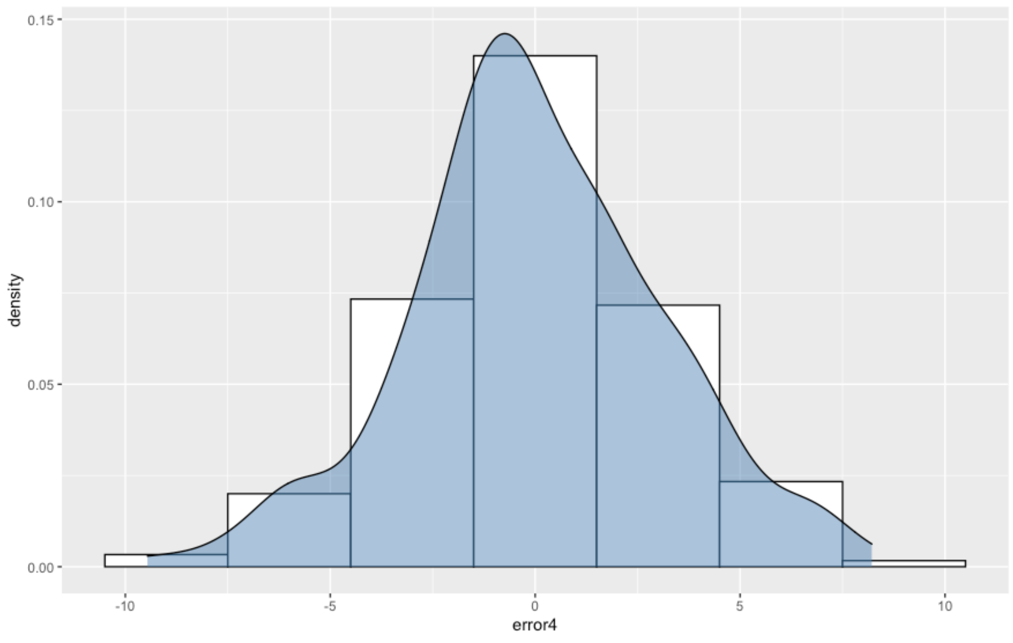


*Figure 18 QQ plot for model 4 predicted errors*



`*Figure 19  histogram for model 4 predicted errors*

## Model 5

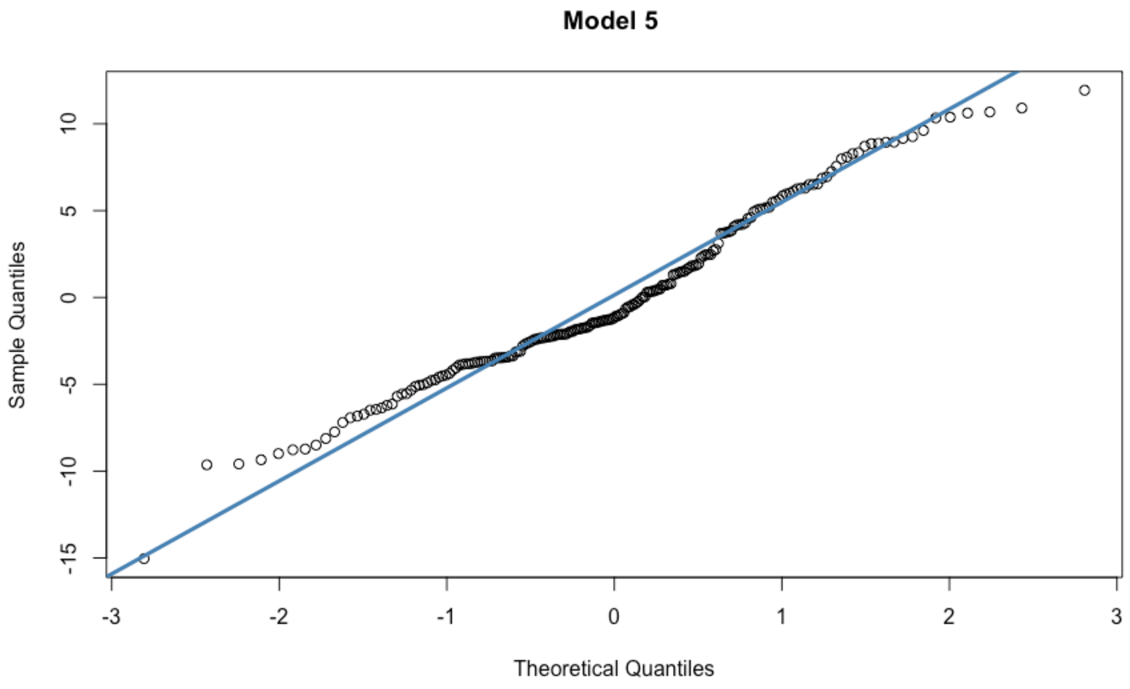
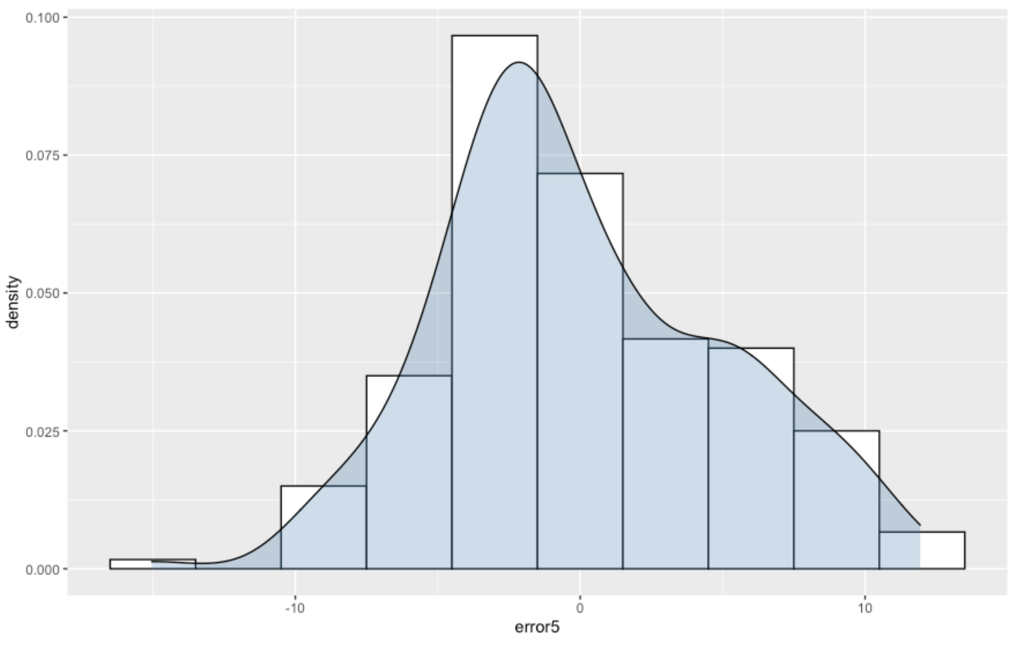


*Figure 20 QQ plot for model 5 predicted errors*



*`Figure 21  histogram for model 4 predicted errors*

## Sub-conclusion

As we can see in our model 3 and model 4 have normally distributed errors whereas others don't this means our model 3 and 4 prediction values are good but according to our AIC and BIC values, model three is less complex and has a high likelihood.

### 2.6 Best Regression Model

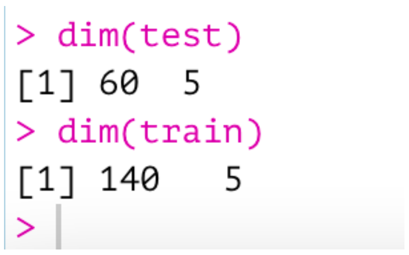
In this task, we intend to get inference out of the value AIC and BIC values for each candidate model. The lesser the AIC and BIC value better the model we can also say the better the model better the AIC we can break AIC into two parts 2k increases as we increase the parameters in the model this penalises us for building models that are complicated the second part 2log(log-likelihood) decreases as our model better explains the data, this will reward us for building models that fit our data well. BIC can also be split into two parts here k.log(n) increases as we increase the number of parameters in the model but also as the number of observations increases this again penalises us for building models which are complicated whereas 2 logs (log-likelihood) will compliment us for building model that fits our model well. Both AIC and BIC strike a balance between the model that is complex and models that are better at explaining the data. The AIC and BIC values shown in table 8 indicate that model 3 is better and help us in predicting the output signal with better accuracy since it has the lowest RSS value and it has the highest log-likelihood value and the AIC and BIC values are the least compared with other models, therefore, we can conclude that our third model to be the best. We can further test our model practically by doing regression with the selected model which gets us to the next task.

Refer  appendix 6.11 for the code

### 2.7  Model Testing

#### 2.7.1 Train Test Split

In this task we combine the given data and make it into one dataset this dataset is again randomly split. 70% of the data for training the model and 30 % for testing the predicted values with the actual values.



```
> dim(test)
[1] 60  5
> dim(train)
[1] 140   5
> 
```

*Figure 22 dimensions of train test split*

All the supporting codes for the train test split are in the appendix. Refer appendix 6.10

### 2.7.2 Estimating best model parameters

Like we have done in task 2.1 we again estimate the selected model parameters and find the theta hat value. But this time we are only going to do it for the testing set the outputs are shown in the below table. The supporting code is in the appendix section

### 2.7.3 Plotting the errors and 95% Confidence interval

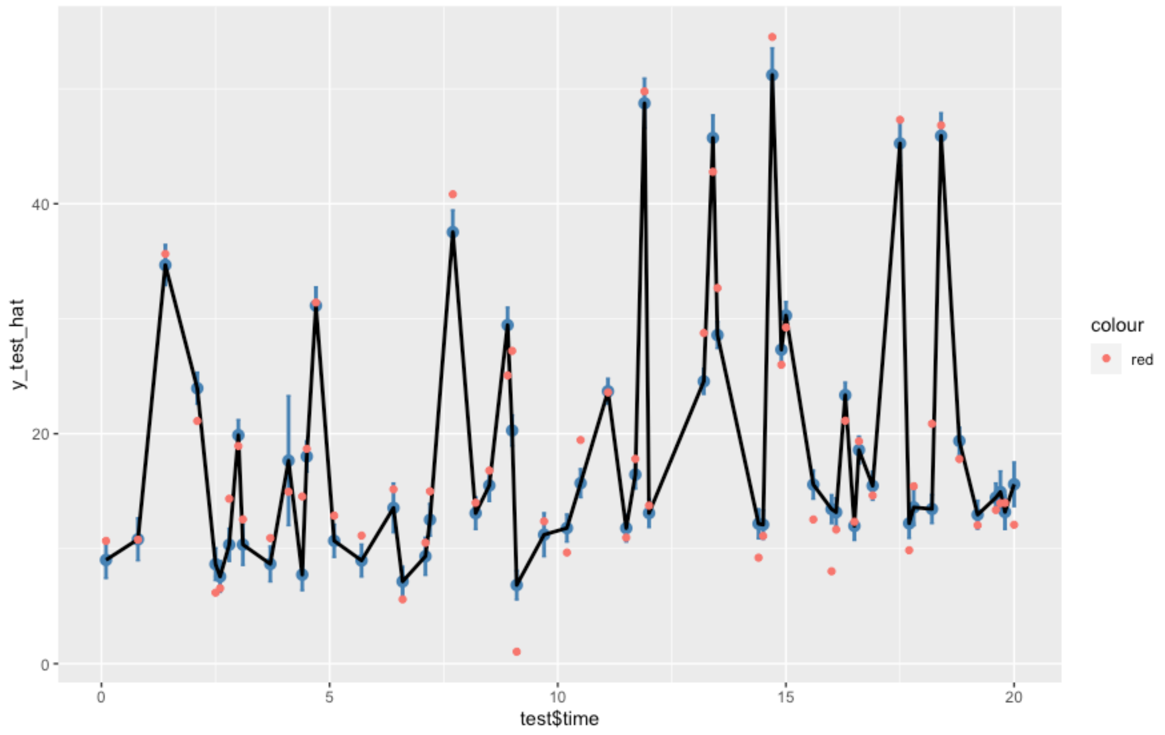


*Figure 23 plotting the errors and the confidence interval of the errors*

The figure shown above demonstrates the error points in the test dataset which is plotted against time plotting confidence interval is useful because it is a statistical test performed visually because the interval covers 95% of the mean we know that anything outside of it occurs less than 5% of the time. In the graph for each error point, the error bars are calculated and used to bracket the errors.

**PART -3**

## 3 Approximate Bayesian computation (ABC)

In the task, we are using rejection ABC to compute the posterior distribution for our best model which is model 3. The process is to find the 2 parameter value of given 5 parameters, we use the 2 highly contributing parameters from our selected model then we give the two parameters some range and plot an empty plot which looks something like this check figure 24. Refer appendix 6.12

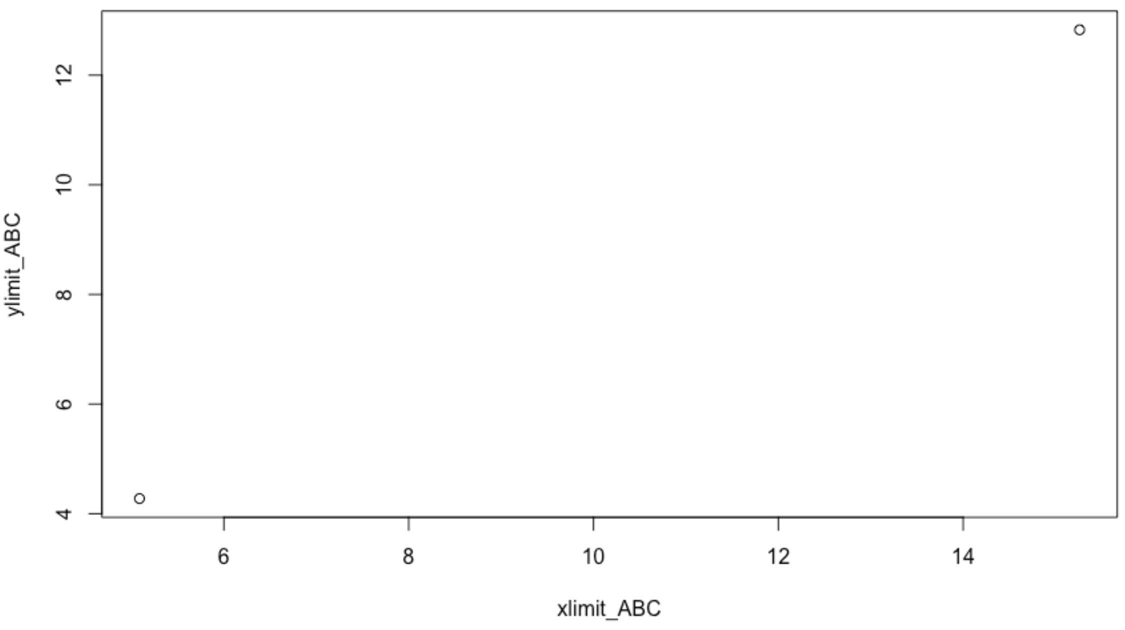


*Figure 24 setting up the range for the two parameters*

This plot addresses our task 3.1. As we finish plotting our empty plot we now intend to plot uniformly distributed points in the space. But why uniformly distributed instead of randomly distributed. When we plot randomly distributed points the values are in a certain range and that will contradict what we are trying to achieve in this task.

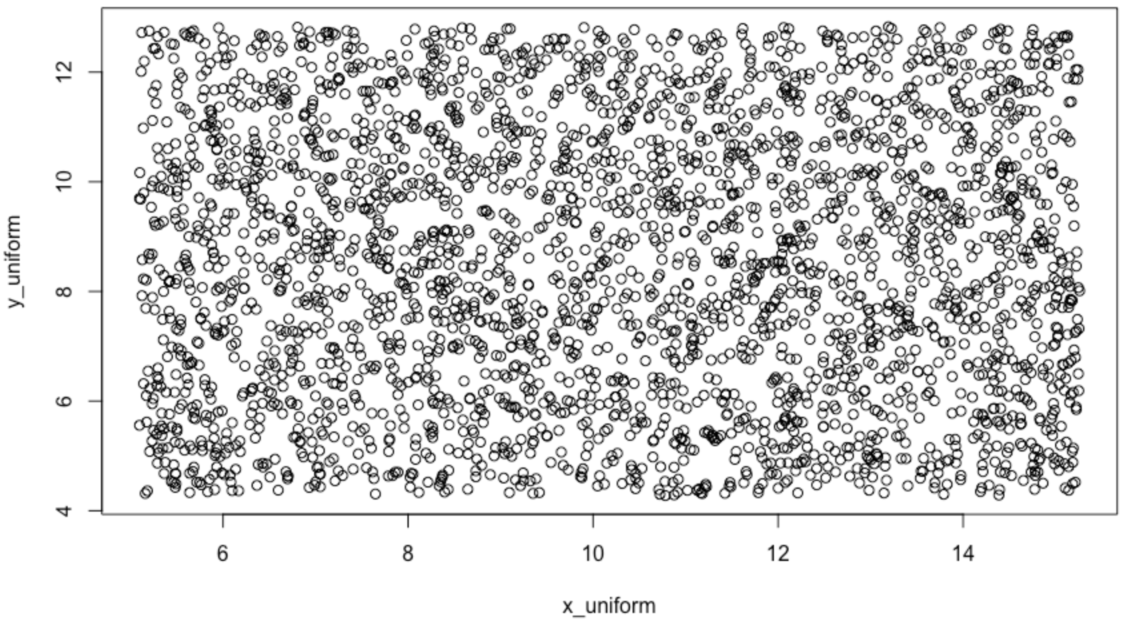


*Figure 25 uniformly distributed plot inside the range*

This plot addresses the 3.2 task in part 3. After this we plan to select every single point inside the plot and perform the rejection ABC for this we will be using a for loop to iterate every single point in the distribution. In that loop, we are going to find the prediction errors and compare the errors with the output signal (y). Now we will have errors for every single point but we need more meaningful values to do rejection ABC so we find the sum of squared errors and construct an if statement to accept the values which provide a value of the sum of squared errors that is below a selected threshold. The accepted values are assigned to an empty variable where all the values are stored and used to plot them later. The sigma value was adjusted based on the number of inputs.

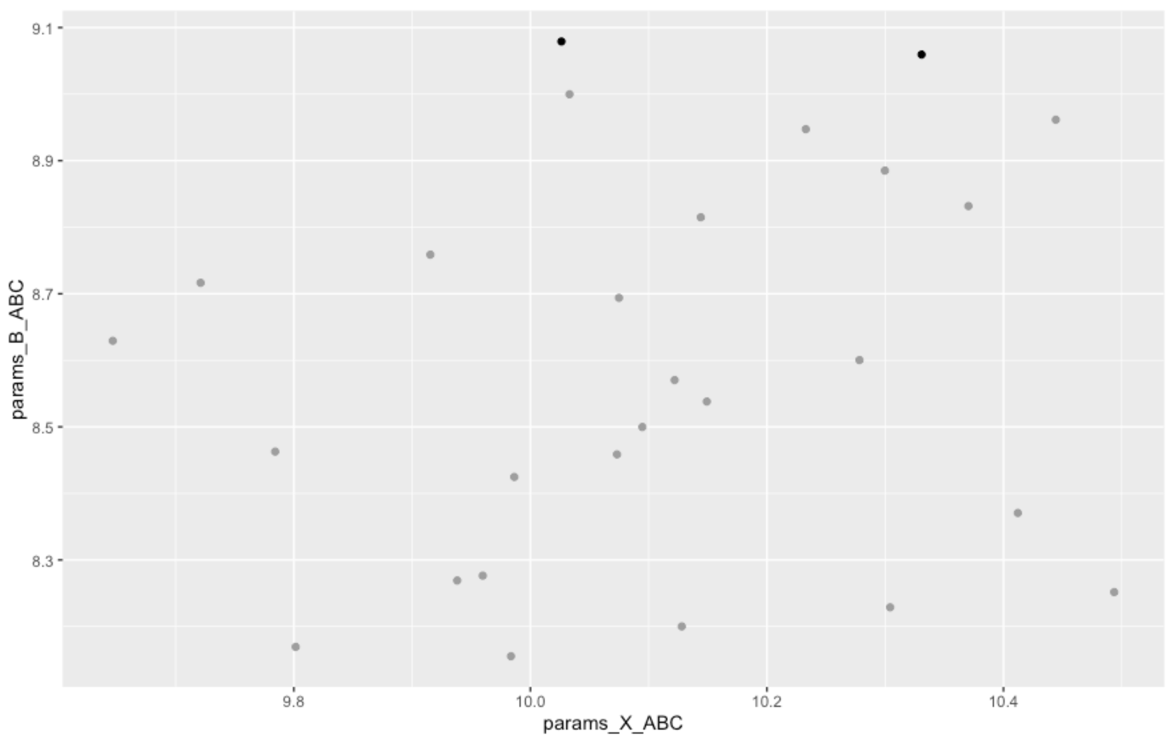


*Figure 26 post-rejection ABC*

This experiment was done with only 3000 points we can increase the points to get many closely related points. Since we have estimated the points which closely resemble our model 3 parameters now let's go ahead and plot and joint and marginal distribution on the points we have found and we can get an inference out of the plot

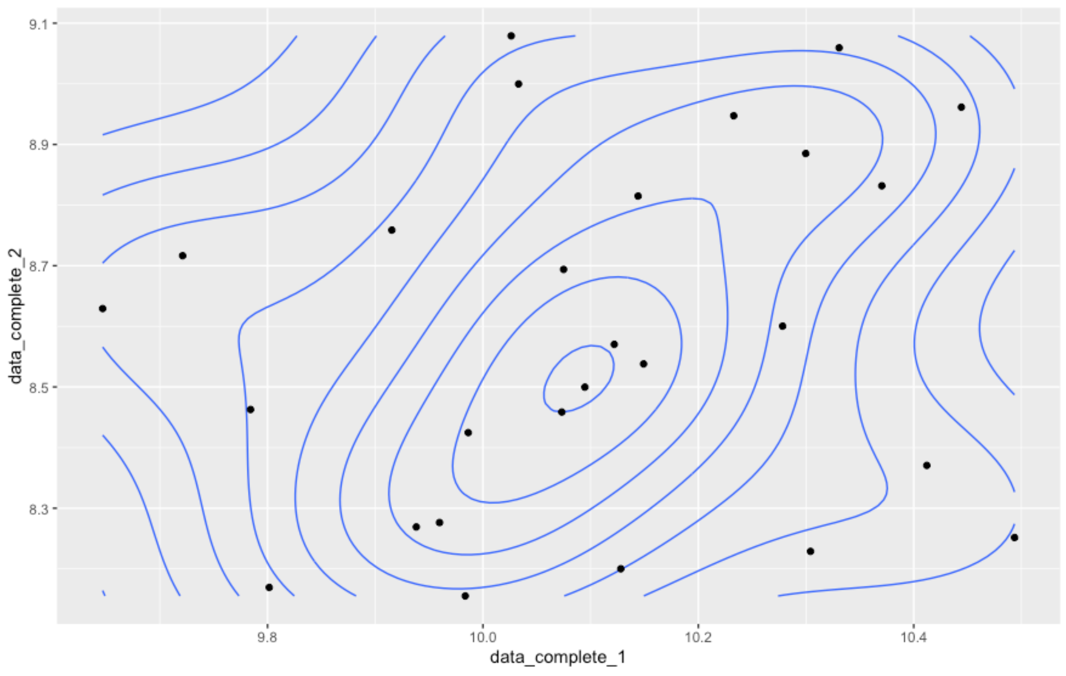


*Figure 27 joint distribution for the selected two parameters*

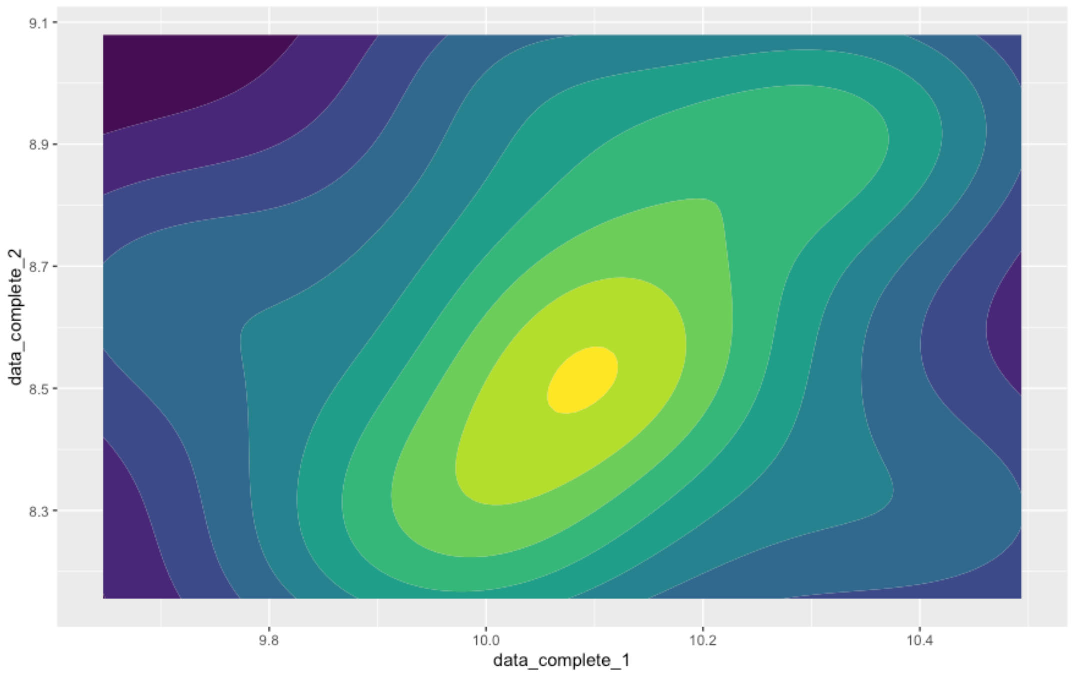


*Figure 28 Joint distribution for the selected model with contour*

The above two graphs indicate where the points which closely resemble our model 3 parameter lie we can also plot a marginal distribution plot to determine the range of the parameters we have accepted.

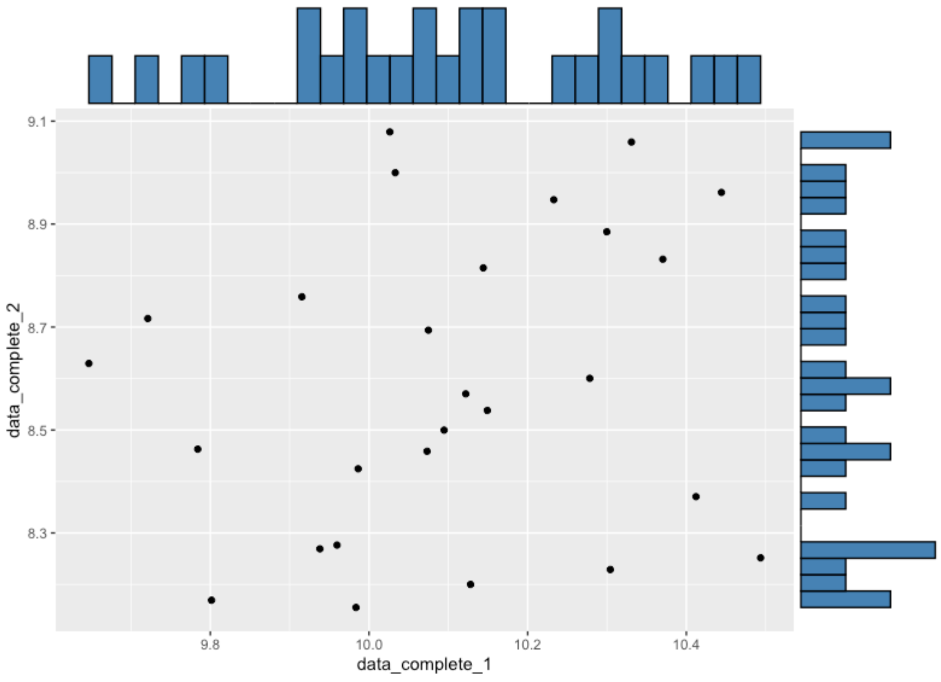


*Figure 29 Marginal distribution*

The above plot can be separated into 3 parts in the middle we have a relationship plot on how x and y are related this can give us a sense of the joint distribution of our data if we want to know the more about the x variable of the data, we push the data variables to the top and using bin width we can plot a histogram based on that we can get a sense of our distribution of the x variable the same can be done with y variable. These two plots(histogram) can give us a sense of what our marginal distribution looks like for both x and y.

## 4 DISCUSSION

In the first part of the report, we have understood the data and the errors in the data by plotting different kinds of plots this helped us to move forward with the data, and the time series gave us an understanding of the change in signal when there was a voice change. The histogram helped us to determine the distribution of the data and scatter plots to find the relation between the input signal and the output signal. In the second, part we were able to determine the best model out of 5 models. It was found that the third model has less RSS and low AIC & BIC values and the model also had a high likelihood from which we were able to determine the best model. From the QQ plot, It was found that the model 3 errors had a normal distribution. In the final part of our report, we were able to estimate our model parameters by using rejection ABC. We were also able to achieve the best possible output from the rejection process

## 5 CONCLUSION

In conclusion, we have found the best regression model which can predict the output with fewer errors. With the above tests, we have also found that the emotional narration does increase the brain repones minimally.

## 6 APPENDIX

### 6.1 Appendix A– Importing the dataset

```r
# DataSet

x <- read.csv("X.csv" ,header = T)
y <- read.csv("Y.csv", header = T)
time <- read.csv("time.csv", header = T)


X = data.matrix(x)
Y = data.matrix(y)
t = data.matrix(time)

x1 = X[, 1]
x2 = X[, 2]
t2 =t[, 1]
y = y[, 1]

ones = matrix(1, length(x1), 1)
```

### 6.2 Appendix B– Time-series Plot

```r
#Task 1
#Draw a time series plot

# plot for time and x1
plot(t2,x1)
lines(t2,x1, type="l", col="brown")
```

```
time_x1 <- ggplot(mapping = aes(x = t, y = x1, z =x2==0))
time_x1 +  geom_point() + geom_line(col="brown")
?ggplot
ggplot(data=x[x$x2>=0,], aes(x= t , y= x1)) + geom_point()+
geom_smooth(method = "lm") + geom_line() +ggtitle("Input
Signal vs Time") + xlab("Time") + ylab("Input Signal")
+theme(plot.title = element_text(color="BLACK", size=16,
face="bold.italic", hjust = 0.5))

axis.title.x = element_text(color="black", size=14,
face="bold")

axis.title.y = element_text(color="black", size=14,
face="bold")

time_series1 <-  ts(total2, start = 0.1, frequency = 10)
plot(time_series1, col = 2)
# plotting time series seprately
ggplot(data=total2[total2$x2],aes(x= t, y= x1, z =x2==1,
colour = x2))+ geom_point()+ geom_line()
ggplot(mapping = aes(x = t , y = x1, z = x2==1, colour = x2))
+ geom_point() + geom_line() +ggtitle("Input Signal vs Time")
+ xlab("Time") + ylab("Input Signal") +theme(plot.title =
element_text(color="BLACK", size=16, face="bold.italic", hjust
= 0.5))
axis.title.x = element_text(color="black", size=14,
face="bold")
axis.title.y = element_text(color="black", size=14,
face="bold")
#Plot for time and y


plot(t2,y)
```

```
lines(t2,y, type= "c" ,col = "blue")



time_y1 <- ggplot(mapping = aes(x = t2, y = y))
time_y1+geom_point()+ geom_line(col="black")+ geom_smooth()
+ggtitle("Output vs Time") + xlab("Time") + ylab("Output
Signal") +theme(plot.title = element_text(color="BLACK",
size=16, face="bold.italic", hjust = 0.5))
axis.title.x = element_text(color="black", size=14,
face="bold")
axis.title.y = element_text(color="black", size=14,
face="bold")

# plotting time series seprately
ggplot(data=total2,aes(x= t, y= y, z =x2==1,   colour = x2))+
geom_point()+ geom_line()+ geom_line(col="black")+
geom_smooth() +ggtitle("Output vs Time") + xlab("Time") +
ylab("Output Signal") +theme(plot.title =
element_text(color="BLACK", size=16, face="bold.italic", hjust
= 0.5))
axis.title.x = element_text(color="black", size=14,
face="bold")
axis.title.y = element_text(color="black", size=14,
face="bold")
```

## 6.3 Appendix C– Distribution Plot

```
# Distribution for each input and output
#input distribution
```

```
summary(x1)
median(x1)
mode(x1)
sd(x1)
var(x1)
dist_plot_input <-  ggplot(mapping = aes(x1, y = ..density..))
dist_plot_input + geom_histogram(binwidth = 0.4,col= "black",
fill = "blue")
dist_plot_input + geom_density(alpha = 0.5,fill= "steelblue")
dist_plot_input + geom_histogram(binwidth = 0.5,col= "black",
fill = "steelblue") + geom_density(alpha=.2,
fill="white")+geom_vline(aes(xintercept=mean(x1,
na.rm=T),color= "Mean"), linetype="dashed", size=1)+
geom_vline(aes(xintercept=median(x1, na.rm=T),color =
"Median"), linetype="solid", size=1)+
ggtitle("Input Signal Distribution") + xlab("Input Signal") +
ylab("Density") +theme(plot.title =
element_text(color="BLACK", size=16, face="bold.italic", hjust
= 0.5))
axis.title.x = element_text(color="black", size=14,
face="bold")
axis.title.y = element_text(color="black", size=14,
face="bold")


dist_plot_input+geom_histogram(binwidth=0.5,
colour="steelblue", fill="white")
+geom_vline(aes(xintercept=mean(x1, na.rm=T)),color="red",
linetype="dashed", size=1)

# expo distribution
curve(dexp(t1, rate = .5), from=0, to=10, col='blue')
```

```
dist_plot_input+geom_histogram(binwidth=.4, colour="brown",
fill="white") +geom_vline(aes(xintercept=mean(x1,
na.rm=T)),color="red", linetype="dashed", size=1)+
geom_vline(aes(xintercept=median(x1, na.rm=T)),color="blue",
linetype="dashed", size=1)
# output distribution
summary(y)
median(y)
mode(y)
sd(y)
var(y)
hist(sqrt(y))
dist_plot_output <-  ggplot(mapping = aes(y, y = ..density..))
dist_plot_output + geom_histogram(binwidth = 5,col= "black",
fill = "blue")
dist_plot_output + geom_density(alpha = 0.5,fill= "orange")
dist_plot_output + geom_histogram(binwidth = 5,col= "black",
fill = "steelblue") + geom_density(alpha=.2, fill="steelblue")
+geom_vline(aes(xintercept=mean(y, na.rm=T)),color="red",
linetype="dashed", size=1) +geom_vline(aes(xintercept=mean(y,
na.rm=T),color= "Mean"), linetype="dashed", size=1)+
geom_vline(aes(xintercept=median(y, na.rm=T),color =
"Median"), linetype="solid", size=1)+ggtitle("Output Signal
Distribution") + xlab("Output Signal") + ylab("Density")
+theme(plot.title = element_text(color="BLACK", size=16,
face="bold.italic", hjust = 0.5))
axis.title.x = element_text(color="black", size=14,
face="bold")
axis.title.y = element_text(color="black", size=14,
face="bold")

dist_plot_output+geom_histogram(binwidth=5, colour="brown",
fill="white") +geom_vline(aes(xintercept=mean(y,
na.rm=T)),color="red", linetype="dashed", size=1)
```

```
+geom_vline(aes(xintercept=mean(y, na.rm=T),color= "Mean"),
linetype="dashed", size=1)+
geom_vline(aes(xintercept=median(y, na.rm=T),color =
"Median"), linetype="solid", size=1)+ggtitle("Input Signal
Distribution") + xlab("Input Signal") + ylab("Density")
+theme(plot.title = element_text(color="BLACK", size=16,
face="bold.italic", hjust = 0.5))
axis.title.x = element_text(color="black", size=14,
face="bold")
axis.title.y = element_text(color="black", size=14,
face="bold")
```

## 6.4 Appendix D– Box Plot

```
# Box plot
?geom_boxplot()
# output
boxplot(sqrt(y))
output_box <- boxplot(y)
output_box$stats
box_plot_output <- ggplot(mapping = aes(y))

box_plot_output + geom_boxplot(fill = "yellow")
box_plot_output + geom_boxplot(outlier.colour =
"steelblue",outlier.shape = 16,outlier.size = 5,fill = "gray")
+ggtitle("Box Plot For Output Signal ") + xlab("Output
Signal") +theme(plot.title = element_text(color="BLACK",
size=16, face="bold.italic", hjust = 0.5))
axis.title.x = element_text(color="black", size=14,
face="bold")
```

```
axis.title.y = element_text(color="black", size=14,
face="bold")

# inoput
input_box <- boxplot(x1)
input_box$stats
box_plot_input <- ggplot(mapping = aes(x1))
box_plot_input + geom_boxplot(outlier.colour =
"steelblue",outlier.fill = "steelblue",outlier.shape =
16,outlier.size = 5,fill= "gray") +ggtitle("Box Plot For Input
Signal") + xlab("Output Signal") +theme(plot.title =
element_text(color="BLACK", size=16, face="bold.italic", hjust
= 0.5))
axis.title.x = element_text(color="black", size=14,
face="bold")
axis.title.y = element_text(color="black", size=14,
face="bold")
```

### 6.5 Appendix E– Scatter Plot

```
# Correlation and scatter plot
cor_value <- cor.test(x1,y)
cor_value
# there is a fairly modest positive correlation
#Scatter plot for Input and output signal
?geom_point
# x = input and y = output
Scatter_plot <- ggplot(mapping = aes (x1, y, colour = x1))
Scatter_plot + geom_point() + geom_smooth(method= "lm")
+ggtitle("Scatter Plot ") + xlab("Input Signal") +
ylab("Output Signal") +theme(plot.title =
element_text(color="BLACK", size=16, face="bold.italic", hjust
= 0.5))
```

```r
axis.title.x = element_text(color="black", size=14,
face="bold")
axis.title.y = element_text(color="black", size=14,
face="bold")

Scatter_plot + geom_point(method = "lm") + geom_smooth(method=
"loess")  +ggtitle("Scatter Plot ") + xlab("Input Signal") +
ylab("Output Signal") +theme(plot.title =
element_text(color="BLACK", size=16, face="bold.italic", hjust
= 0.5))
axis.title.x = element_text(color="black", size=14,
face="bold")
axis.title.y = element_text(color="black", size=14,
face="bold")



Scatter_plot
# library(car)
install.packages('car')
library("car")
scatterplot(y ~x[,"x1"])
scatterplot(x[,"x1"] ~y)

# x = output and Y = input
Scatter_plot_2  <-  ggplot(mapping= aes(x = y , y = x1, colour
= y))
Scatter_plot_2 + geom_point() + geom_smooth(method="lm")
Scatter_plot_2 + geom_point() + geom_smooth(method="loess")




ggplot(data=total2[total2$x2.],aes(x= x1, y= y, z =x2==0,
colour = x2))+ geom_point()
+geom_smooth(method="loess")+ggtitle("Scatter Plot ") +
xlab("Input Signal") + ylab("Output Signal") +theme(plot.title
= element_text(color="BLACK", size=16, face="bold.italic",
hjust = 0.5))
axis.title.x = element_text(color="black", size=14,
face="bold")
axis.title.y = element_text(color="black", size=14,
face="bold")
ggplot(data=total2[total2$x2],aes(x= y, y= x1, z =x2==1,
colour = x2))+ geom_point()+ geom_smooth(method="lm")
```

## 6.6 Appendix F- Task 2.1 Estimating model parameters

```
#Task 2 and 2.1
# Model 1

model1_X = cbind(ones, x1^3, x1^5, x2)

thetaHat1 = solve(t(model1_X) %*% model1_X) %*% t(model1_X)
%*% y
thetaHat1

# Model 2

model2_X = cbind(ones,x1,x2)

thetahat2 = solve(t(model2_X)%*% model2_X) %*% t(model2_X) %*%
y
thetahat2

# Model 3

model3_X = cbind(ones, x1, x1^2, x1^4, x2)
```

```
thetaHat3 = solve(t(model3_X)%*% model3_X) %*% t(model3_X) %*%
y
thetaHat3

# Model 4
model4_X = cbind(ones, x1, x1^2,x1^3,x1^5,x2)
thetaHat4 = solve(t(model4_X)%*% model4_X) %*% t(model4_X) %*%
y
thetaHat4

# Model 5
model5_X = cbind(ones,x1,x1^3,x1^4,x2)
thetaHat5 = solve(t(model5_X)%*% model5_X) %*% t(model5_X) %*%
y
thetaHat5
```

### 6.7 Appendix G– Task 2.2 Determining RSS

```
#Task 2.2 (Finding RSS)
# model 1
y_Hat_model1 = model1_X %*% thetaHat1
plot(x1, y, col="blue")
lines(x1,y_Hat_model1, col="red")

error1 = y - y_Hat_model1

error1

SSE_1 = norm(error1, type = "2")^2
SSE1 = sum((y-y_Hat_model1)^2)
print(SSE_1)
```

```
# model 2
y_Hat_model2 = model2_X %*% thetahat2
plot(x1, y, col="blue")
lines(x1,y_Hat_model2, col="red")

error2 = y - y_Hat_model2

error2

SSE_2 = norm(error2, type = "2")^2
SSE2 = sum((y-y_Hat_model2)^2)
print(SSE_2)

# model 3

y_hat_model3 = model3_X %*% thetaHat3
plot(x1, y, col="blue")
lines(x1,y_hat_model3, col="red")

error3 = y - y_hat_model3
error3

SSE_3 = norm(error3, type = "2")^2
SSE2 = sum((y-y_hat_model3)^2)
print(SSE_3)

# model 4

y_hat_model4 = model4_X %*% thetaHat4
plot(x1, y, col="blue")
lines(x1,y_hat_model4, col="red")
```

```
error4 = y - y_hat_model4
error4

SSE_4 = norm(error4, type = "2")^2
SSE2 = sum((y-y_hat_model4)^2)
print(SSE_4)

# model 5

y_hat_model5 = model5_X %*% thetaHat5
plot(x1, y, col="blue")
lines(x1,y_hat_model5, col="red")

error5 = y - y_hat_model5
error5

SSE_5 = norm(error5, type = "2")^2
SSE5 = sum((y-y_hat_model5)^2)
print(SSE_5)
print(SSE5)
```

### 6.8 Appendix H– Computing Log likelihood

```
#Task 2.3 Maximum likeliyhood
#MLE_model_1

varience_sq_1 = SSE_1/(200-1)
```

```
MXL_1 = - (200/2)*log(2*pi)-
(200/2)*log(varience_sq_1)-
(1/(2*varience_sq_1))*SSE_1
MXL_1


#MLE_model_2

varience_sq_2 = SSE_2/(200-1)

MXL_2 = - (200/2)*log(2*pi)-
(200/2)*log(varience_sq_2)-
(1/(2*varience_sq_2))*SSE_2
MXL_2

#MLE_model_3

varience_sq_3 = SSE_3/(200-1)

MXL_3 = - (200/2)*log(2*pi)-
(200/2)*log(varience_sq_3)-
(1/(2*varience_sq_3))*SSE_3
MXL_3

#MLE_model_4

varience_sq_4 = SSE_4/(200-1)
```

```
MXL_4 = - (200/2)*log(2*pi)-
(200/2)*log(varience_sq_4)-
(1/(2*varience_sq_4))*SSE_4
MXL_4

#MLE_model_5

varience_sq_5 = SSE_5/(200-1)

MXL_5 = - (200/2)*log(2*pi)-
(200/2)*log(varience_sq_5)-
(1/(2*varience_sq_5))*SSE_5
MXL_5
```

### 6.9 Appendix I– Calculating AIC and BIC

```
#task 2.4 AIC and BIC

# AIC
# model 1 AIC

AIC_1 <-  2*(4) - 2*(MXL_1)
AIC_1

# model  2 AIC
AIC_2 <-  2*(3)- 2*(MXL_2)
AIC_2

#model 3 AIC
AIC_3 <- 2*(5)- 2 *(MXL_3)
AIC_3
```

```
# model 4 AIC
AIC_4 <-  2*(6)- 2*(MXL_4)
AIC_4

#model 5 AIC
AIC_5  <- 2*(5) - 2*(MXL_5)
AIC_5

#BIC
#model 1 BIC

BIC_1 <-  4*(log(200)) - 2*(MXL_1)
BIC_1

#model 2 BIC
BIC_2 <- 3*(log(200)) - 2*(MXL_2)
BIC_2

# model 3 BIC
BIC_3 <-  5*(log(200)) - 2*(MXL_3)
BIC_3

# model 4 BIC
BIC_4 <-  6*(log(200)) - 2*(MXL_4)
BIC_4

# model 5 BIC
BIC_5 <-  5*(log(200)) - 2*(MXL_5)
BIC_5
```

## 6.10 Appendix J– QQ plot and distribution

```
# Task 2.5
```

```
#hist plot  for model 1 error && QQplot
error_plot_1 <- ggplot(mapping = aes(error1 , y =
..density..))
error_plot_1  + geom_histogram(binwidth = 5 ,fill = "red", col
= 'black' )
error_plot_1 + geom_density(alpha= 0.5, fill = "green ")
error_plot_1 + geom_histogram(binwidth = 5 ,fill = "white",
col = 'black' ) + geom_density(alpha= 0.5, fill = "steelblue")

qqnorm(error1 ,main=' Model 1')
qqline(error1, col = "steelblue", lwd = 3)
# hist_plot  for model 2 error && QQplot
error_plot2 <- ggplot(mapping = aes(error2 , y = ..density..))
error_plot2  + geom_histogram(binwidth = 5,fill = "orange",
col= "black")
error_plot2 + geom_density(alpha = 0.5, fill=  "steelblue")
error_plot2 + geom_histogram(binwidth = 5,fill = "white", col=
"black") + geom_density(alpha = 0.5, fill=  "steelblue")

qqnorm(error2, main= 'Model 2')
qqline(error2, col = "steelblue", lwd = 3, main ="Model 1")
# hist plot  for model 3 error && QQplot
error_plot3 <- ggplot(mapping = aes(error3 , y = ..density..))
error_plot3  + geom_histogram(binwidth = 5,fill = "orange",
col= "black")
error_plot3 + geom_density(alpha = 0.5, fill=  "blue")
error_plot3 + geom_histogram(binwidth = 3,fill = "white", col=
"black") + geom_density(alpha = 0.5, fill=  "steelblue")

qqnorm(error3, main = "Model 3")
qqline(error3, col = "steelblue", lwd = 3)

# hist plot for model 4 QQ plot for model 4
```

```
error_plot4 <- ggplot(mapping = aes(error4 , y = ..density..))
error_plot4  + geom_histogram(binwidth = 5,fill = "orange",
col= "black")
error_plot4 + geom_density(alpha = 0.5, fill=  "blue")
error_plot4 + geom_histogram(binwidth = 3,fill = "White", col=
"black") + geom_density(alpha = 0.5, fill=  "steelblue")

qqnorm(error4, main = "Model 4")
qqline(error4, col = "steelblue", lwd = 3)

## hist plot for model 4 && QQ plot for model 5
error_plot5 <- ggplot(mapping = aes(error5 , y = ..density..))
error_plot5  + geom_histogram(binwidth = 5,fill = "orange",
col= "black")
error_plot5 + geom_density(alpha = 0.5, fill=  "blue")
error_plot5 + geom_histogram(binwidth = 3,fill = "white", col=
"black") + geom_density(alpha = 0.3, fill=  "steelblue")

qqnorm(error5, main = "Model 5 " )
qqline(error5, col = "steelblue", lwd = 3)
```

## 6.11 Appendix K– Task 2.7 Regression modelling

```
#task 2.7
#train test split
total <- cbind(x,y)
total2 <- cbind(total, t)
total3 <-  cbind(total2, ones)
ones
total3
```

```
split2<- sample(c(rep(0, 0.7 * nrow(total3)), rep(1, 0.3 *
nrow(total3))))

split2

table(split2)


train <- total3[split2 == 0, ]
dim(train)
head(train)
train


test  <- total3[split1 == 1, ]
dim(test)
tail(test)
test
?poly
test
# model data
#train data
x1_train<- train$x1
x1_train
head(x1_train)

x2_train <-  train$x2
x2_train

x1_train_SQ <-  (x1_train)^2
x1_train_SQ

x1_train_pow4 <-  (x1_train)^4
x1_train_pow4
```

```
x1_ones_train <- train$ones
x1_ones_train

y_train <-  train$y

# create a model

model_3_poly<- lm(train$y ~ x1_train + x1_train_SQ +
x1_train_pow4 + x2_train, model=TRUE)
model_3_poly_test<- lm(test$y ~ x1_test + x1_test_SQ +
x1_test_pow4 + x2_test)
summary(model_3_poly)


?lm
predict_df <- data.frame(test$x1, test$x2)
df<- predict(model_3_poly, predict_df)
df
?predict

plot(x1_train,train$y)
plot(x1,y)
lines(smooth.spline(x1_train, predict(model_3_poly)), col =
'blue', lwd = 3)
lines(smooth.spline(train$x1, train$y), col = 'red', lwd = 3)
length(df)


## 2.7.1 Estimate the model parameters

model3_X_train = cbind(x1_ones_train, x1_train, x1_train_SQ,
x1_train_pow4, x2_train)
```

```
thetaHat3_train = solve(t(model3_X_train)%*% model3_X_train)
%*% t(model3_X_train) %*% y_train
thetaHat3_train

## 2.7.2 Compute the error
dim(test)
test_matrix <- data.matrix(test)
test_matrix
test_x1 <-  test$x1
test_x1_matrix <- data.matrix(test_x1)
y_test_hat = test_x1_matrix %*% thetaHat3_train
test

ones_test = matrix(1, 60, 1)
X_test_for_prediction = cbind(ones_test, test$x1, (test$x1)^2,
(test$x1)^4, test$x2)
X_test_for_prediction
y_test_hat = X_test_for_prediction %*% thetaHat3_train

dim(y_test_hat)
y_test_hat
test$y

errors = test$y - y_test_hat
errors
## 2.7.3
test_matrix[,4]
#plot(x1, y)
#lines(smooth.spline(y_test_hat), col = 'red', lwd = 3)
#time
#plot(test_matrix[, 4], test$y, col = "black")
#lines(test_matrix[, 4], y_test_hat , col="steelblue")
#segments(test_matrix[, 4], y_test_hat-CI_test,
y_test_hat+CI_test)
```

```
#varience_y_test_hat <-  var(y_test_hat)
#varience_y_test_hat
y_test_hat
dim(test)
summary(test_matrix)
train$time
test$time

n = 60
var_y_hat = matrix(0 , n , 1)
#cov_thetaHat = varience_sq_3 *
(solve(t(X_test_for_prediction) %*% X_test_for_prediction))
#cov_thetaHat_matrix <- data.matrix(cov_thetaHat)
for( i in 1:n){
  X_i = matrix( X_test_for_prediction[i,] , 1 , 5 )
  var_y_hat[i,1] = X_i %% cov_thetaHat_matrix %% t(X_i)
}
var_y_hat
CI_test <- 2 * sqrt(var_y_hat)
CI_test

plot(test_matrix[, 4], y_test_hat)
lines(test_matrix[, 4], y_test_hat , type = "l", lwd= 3)
segments(test_matrix[, 4], y_test_hat-CI_test, test_matrix[,
4], y_test_hat+CI_test,  col="steelblue", lwd= 3)
CI  = 2 * sqrt(var_y_hat) # Confidance interval
plot(test$time, y_test_hat)
lines(test$time, y_test_hat , type = "l", lwd= 3)
segments(test$time, y_test_hat-CI_test, test$time,
y_test_hat+CI_test,  col="steelblue", lwd= 3)
CI_test
```

## 6.12 Appendix L – Task 3 Approximate Bayesian computation (ABC)

```
thetaHat3

xlimit_ABC = c(thetaHat3[1]-0.5 * thetaHat3[1],
thetaHat3[1]+0.5 * thetaHat3[1])
ylimit_ABC = c(thetaHat3[2]-0.5 * thetaHat3[2],
thetaHat3[2]+0.5 * thetaHat3[2])
plot(xlimit_ABC,ylimit_ABC )
x_uniform = runif(3000, min = xlimit_ABC[1], max =
xlimit_ABC[2])
y_uniform = runif(3000, min = ylimit_ABC[1], max =
ylimit_ABC[2])


#x_uniform = rnorm(100, mean = thetaHat3[1], sd = 1)
#y_uniform = rnorm(100, mean = thetaHat3[2], sd = 1)

#x is theta0 and y is theta1
plot(x_uniform, y_uniform)
?rnorm

params_X_ABC <- NULL
params_B_ABC <- NULL

for (i in 1:3000) {
  #the model is y = theta0 + theta1x1 + theta2x1^2 +
theta3x1^4 + theta4x2
  thetaHat_ABC = matrix(c(x_uniform[i], y_uniform[i], 6.247, -
0.283, 4.16))
  y_ABC = model3_X %*% thetaHat_ABC
  error_ABC = y - y_ABC
  rss_ABC =  sum(error_ABC ^ 2)

  if (rss_ABC < 1700) {
    print(i)
    print(rss_ABC)
    print(x_uniform[i])
    print(y_uniform[i])
    params_X_ABC[i] <-(x_uniform[i])
    params_B_ABC[i] <- (y_uniform[i])
  }
}

library(LaplacesDemon)
hist(params_X_ABC)
hist(params_B_ABC)
ggplot(mapping = aes(params_X_ABC,params_B_ABC)) +
geom_point(colour = params_B_ABC, lws= 4)
library(ggExtra)



data_complete_1 <- params_X_ABC[complete.cases(params_X_ABC)]
data_complete_2 <- params_B_ABC[complete.cases(params_B_ABC)]

plot(data_complete_1, data_complete_2)
joint.density.plot(data_complete_1,data_complete_2
,Title="JOINT POSTERIOR PLOT ",contour=TRUE, color= FALSE,
Trace=NULL)
# joint Distribution
joint_dist <- ggplot(mapping = aes(data_complete_1,
data_complete_2),col = "gray")  + geom_point(col="black")
joint_dist+ geom_density2d() + geom_point()
# marginl Distribution
ggMarginal(joint_dist,type = "boxplot")
ggMarginal(joint_dist,type = "histogram", fill = "steelblue")
ggMarginal(joint_dist,type = "density")
```