Modeling EEG Signals using Polynomial Regression

Manash Bhele
6th Februray, 2023
Github: https://github.com/manashb21/R-assignment-EEG

Contents

Introduction

Human brain is possibly the most critical part of the human body. From helping humans perform simple mundane tasks to caryying out extraordinary feats throughout history, the brain is what controls everything in humans. The brain works by sending and receiving signals throughout the body with the aid of the nerves. By analyzing such signals or electrical impulses, changes in brain activity and state can be measured. Electroencephalography, or EEG, is such method used to measure the electrical activity of the brain in which small metal electrodes are attached to the scalp surface. Analysing and interpreting such signals provides a wide range of information about the state of brain.

Task 1

Time Series Plot

A time series is a sequence of data which is recorded continuously over time. Time series helps in studying how variables changes or evolves through time. Generally, when plotting a time series graph, time (or date) is plotted on the X axis or the horizontal axis and the variable is plotted on the Y axis.

Plotting the signals against time provides a way of interpreting the data. Here, EEG input/output signals with their respective time in seconds has been provided. We can simply plot those signals against time to interpret the basic overview of the signals.

Below is a plot of the 4 input signals with respect to their times (in millisecond).

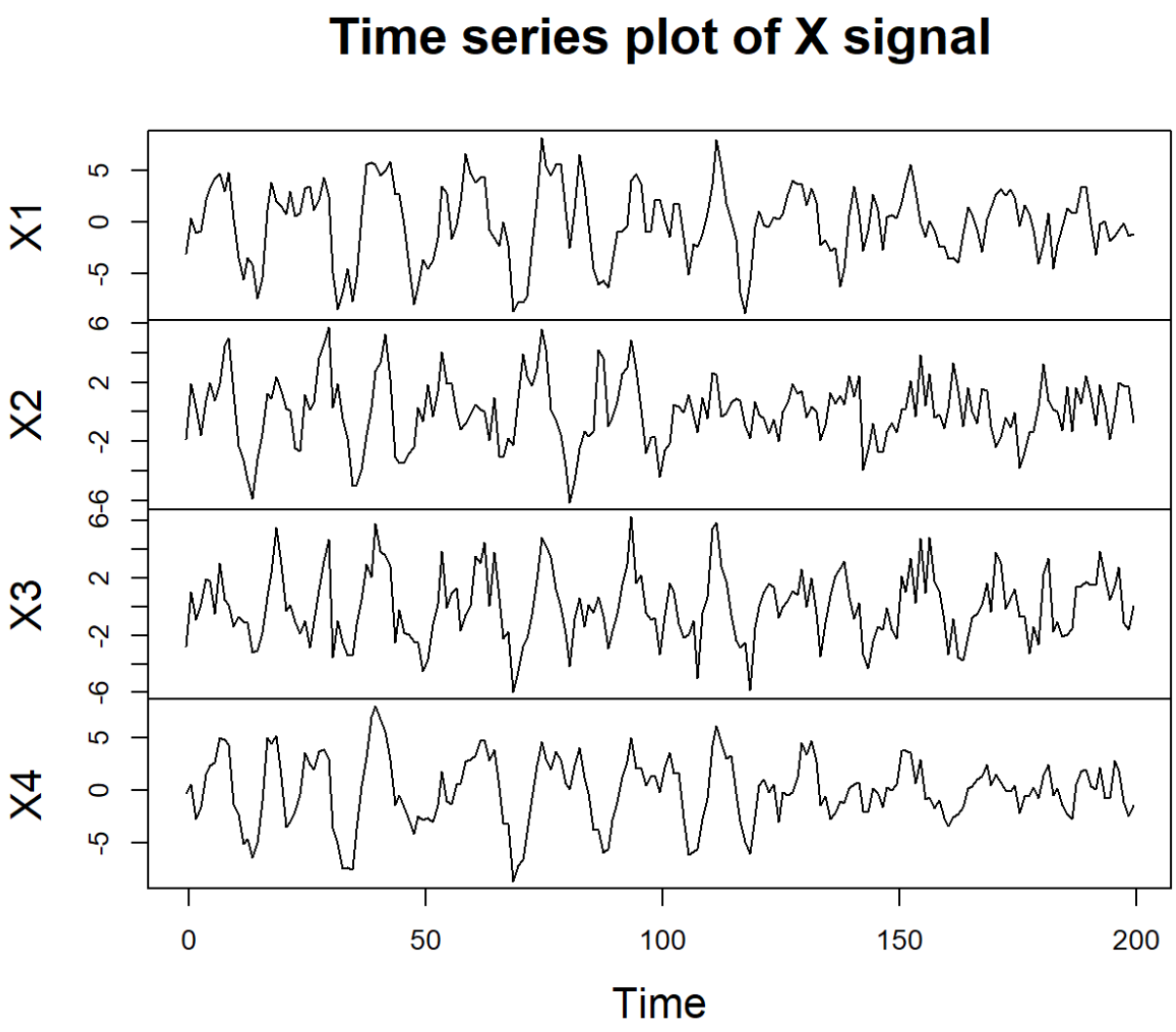


Figure 1 Time Series Plot of X (input) Signal

The above plot shows how the signals change with time. Looking at the plot, it can be seen that there are some values that deviate from the general trend of the signal. Slight spikes can be seen in all the input signals right around the same time frames which gradually transitions into somewhat relaxed state after around 120ms. All the signals have been subjected to noise.

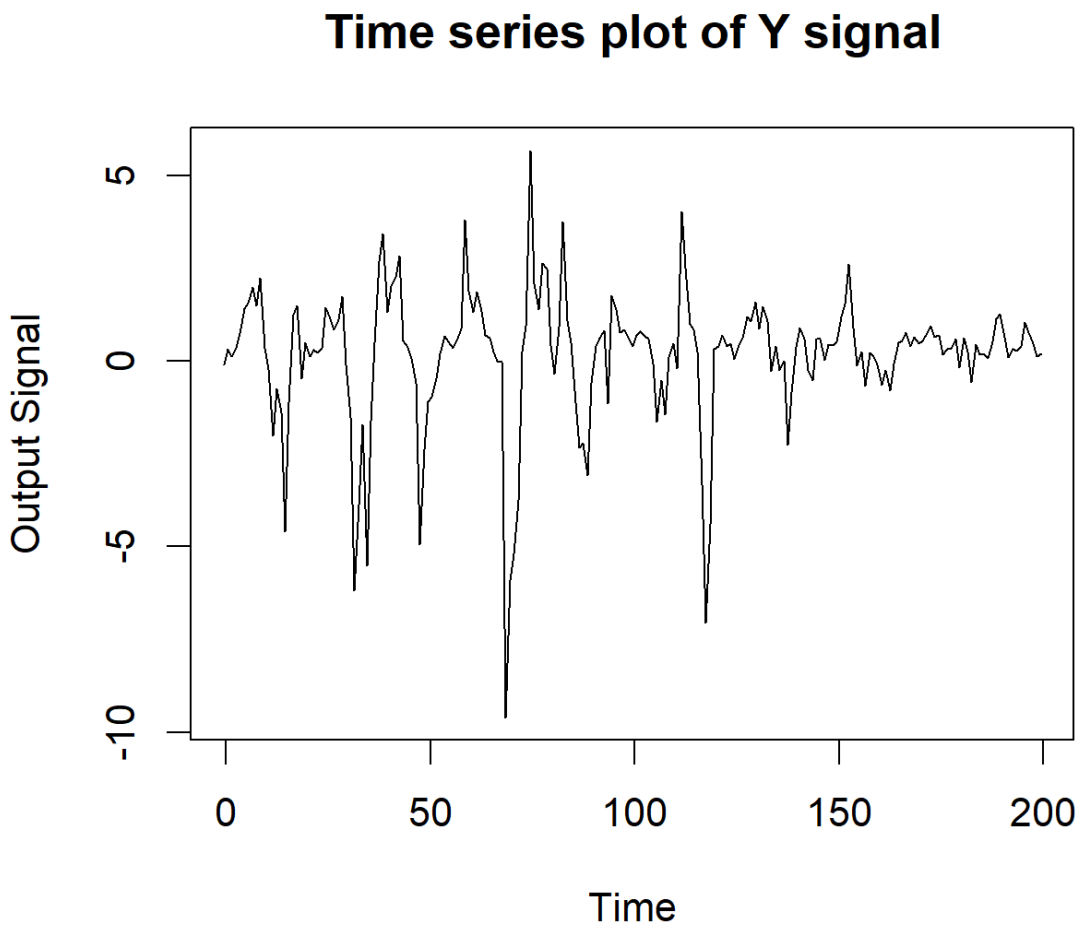


Figure 2 Time Series Plot of Y (Output) Signal

The output signal has also been subjected to some additive noise. We can see certain abrupt spikes in the signal. The abrupt fall of the signal at ~60ms can be explained looking back at ***Error! Reference source not found.*** where the input signals X1, X3 and X4 are seen the have gone down as well at the same time. It is more or less the same at 120ms where the ouput signal has shown abrupt change at the same time where input signals X1, X3 and X4 show similar downward spike.

The general movement of the signal seems quite similar to those of input signals where the spikyness calms down quite significantly after 120ms having seen that all the input signals become significantly steady after 120ms.

Distribution Plots of EEG Signals

On the basis of our data, we can visualize it by plotting histograms and density plots. A histogram is a bar style chart that approximately represents *numerical* datas by classifying them into bins or intervals. Density plots simply shows the density of various points in the data. The distribution shape of a dataset can be determined with the help of distribution plots. The figures below show the histogram and density plot of the input signal.

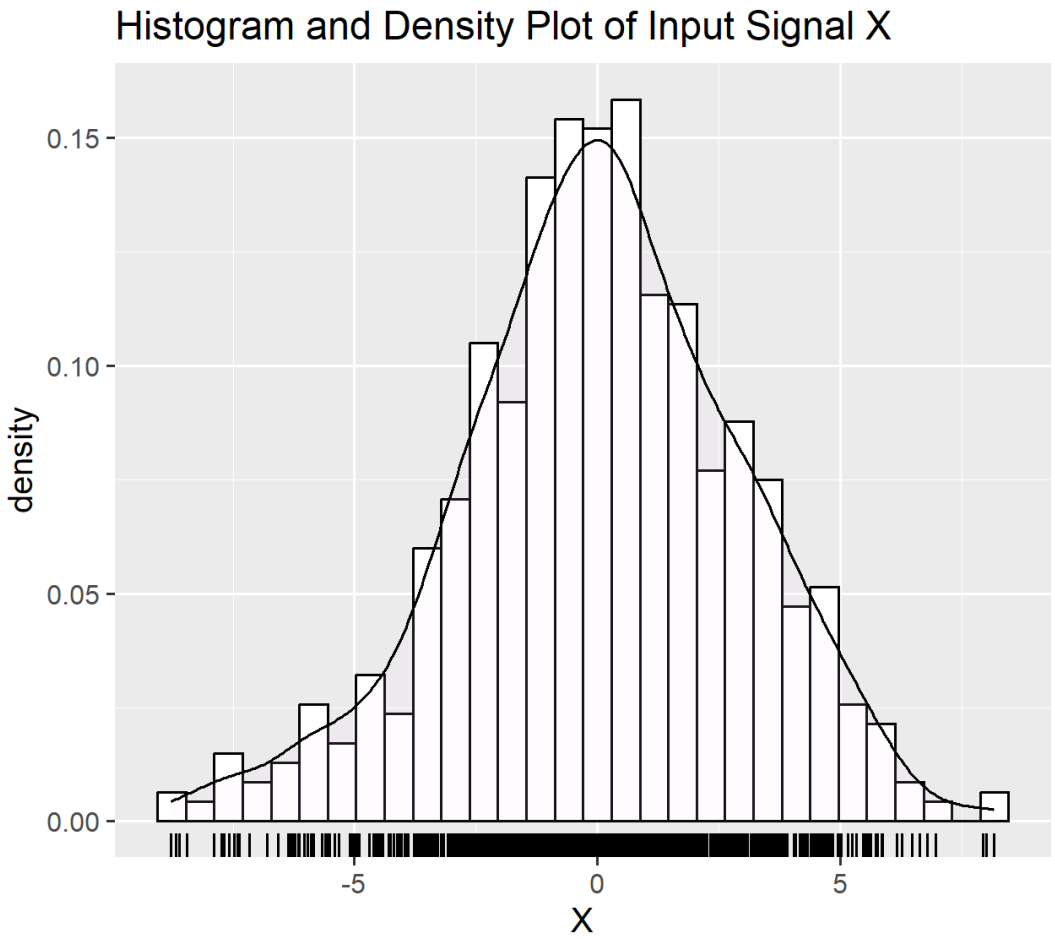
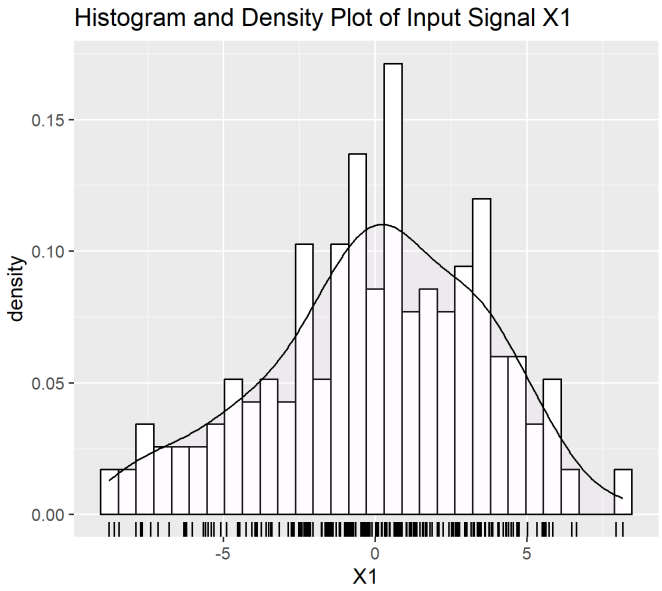
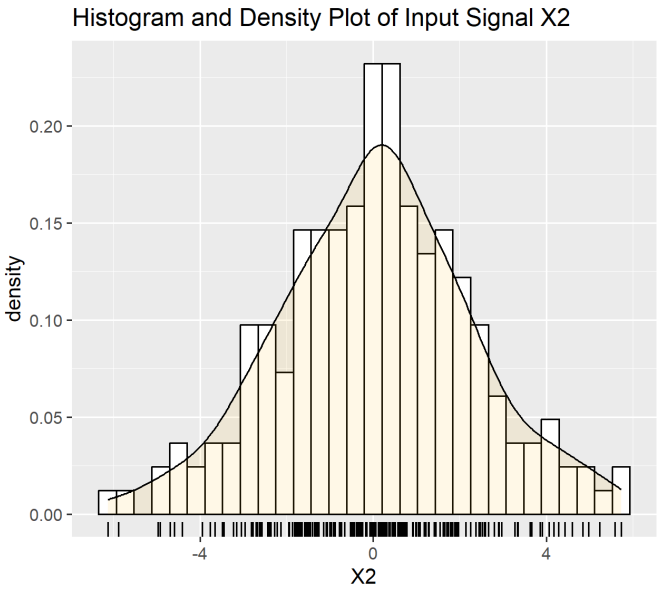
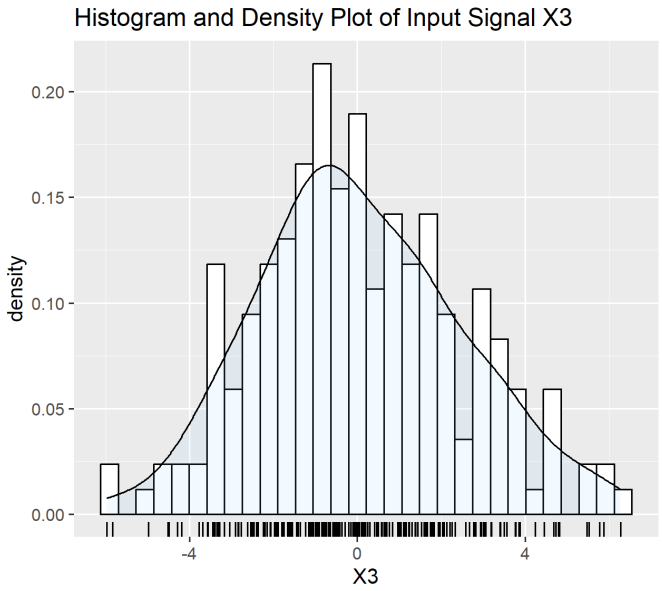


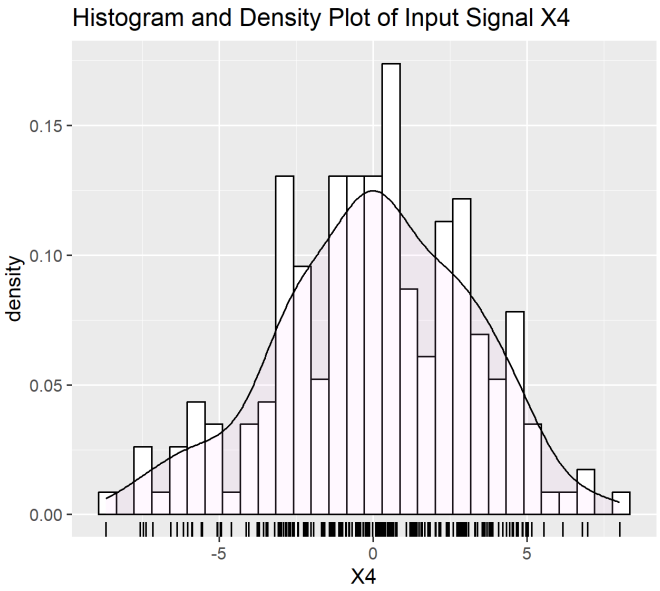
Figure 3 Plot of the Input Signal

The shape of the density plot represents a bell-shaped curve. From the figure we can see that most values are accumulated around the center and there are lesser values at the far end signifying that the signal does not have a lot of extreme values or outliers as the tails of the density plot tapers down.

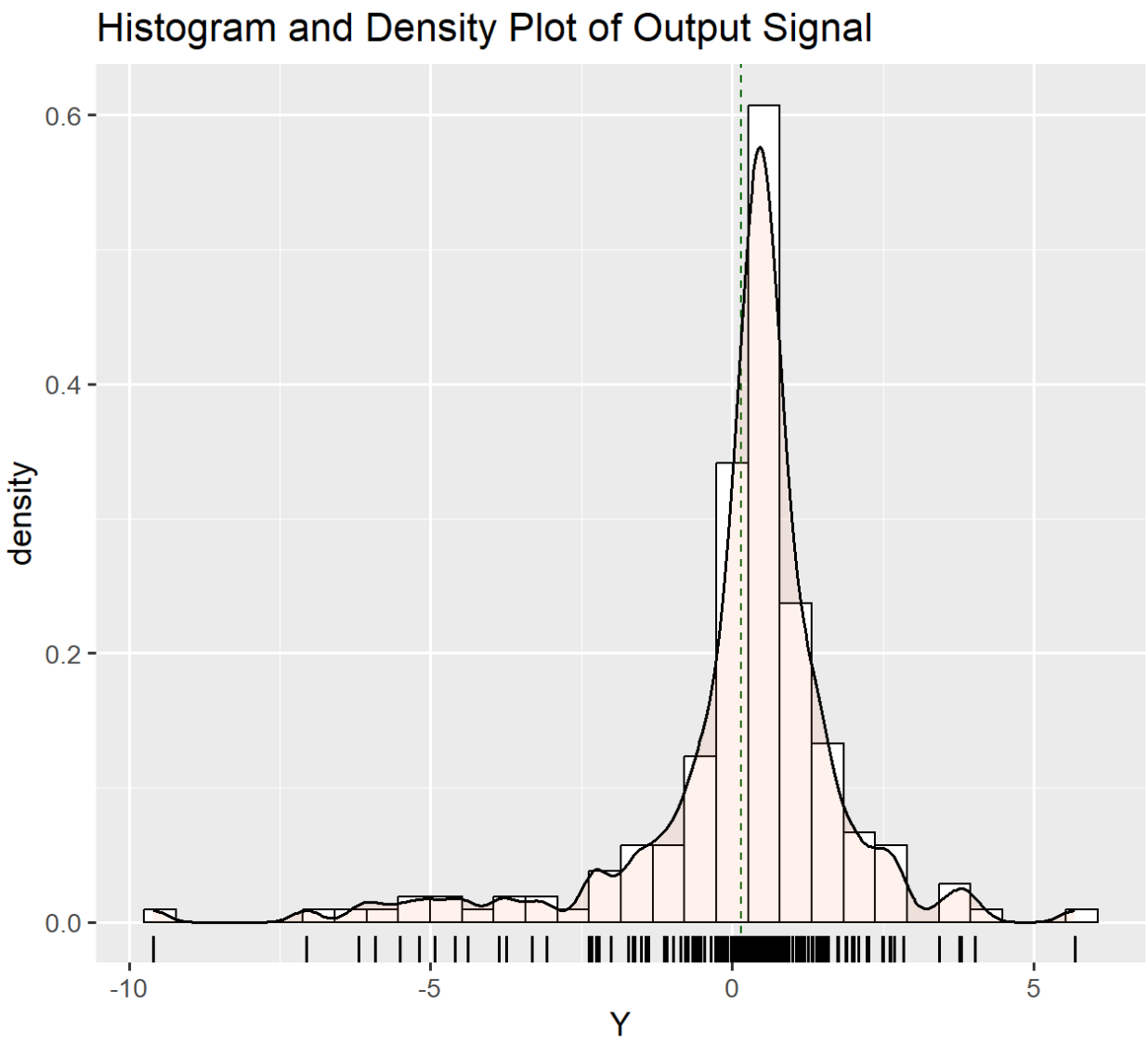








The distribution shapes of all the input signals are fairly close to a bell shaped curve. In plots of all the datasets, it can be seen that the data does not have extreme outliers. Out of all four input values, X2 seems to have the distribution most closest to a Normal Distribution. Some values at the extremes of the plots also verify the spikes seen earlier in the time series plots.



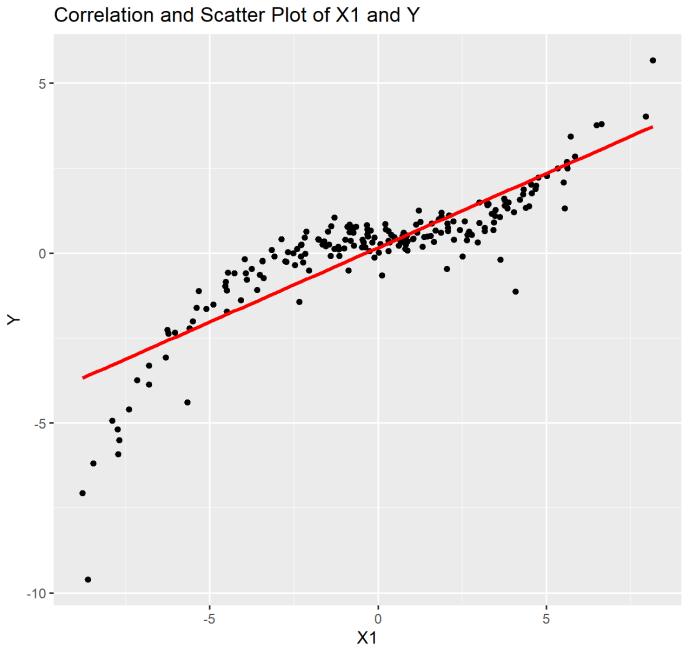
From the plot above, we can see that the left side tail of the signal is longer. Most of the values are concentrated on the right side of the distribution. This shows that the output signal is left skewed or also called negative-skewed. The values can also be seen on the extreme left, far from the mean, which are also called outliers. This also shows that the distribution of the output signal is not symmetrical.

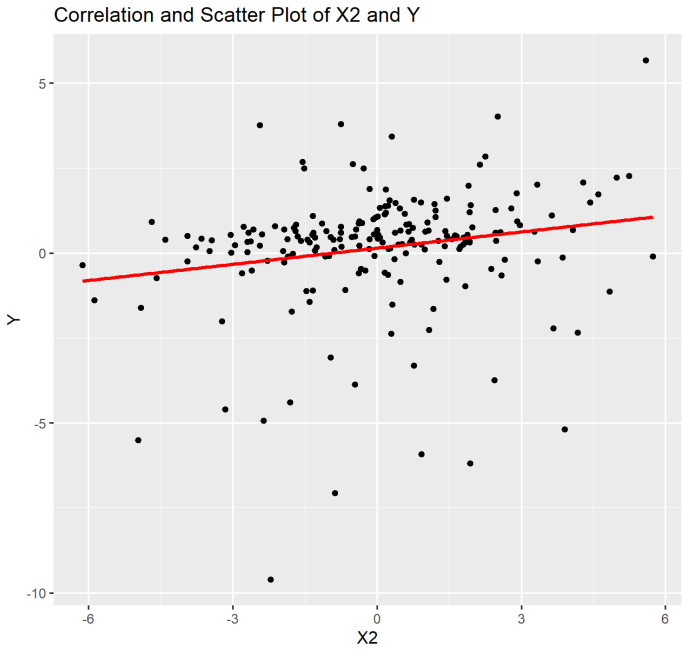
Correlation and Scatter Plots

Scatter Plot is a kind of plot where data points of two variables are plotted in a two-dimensional plane. Generally, the dependent variable is plotted in the Y-axis whereas the independent variable is plotted in the X-axis.

When an individual set of data point is plotted on the chart, it can be represented as a (x, y) coordinate as well. Plotting all the sets of data points on the chart, we can generally see a kind of pattern or a trend in which it either looks like rising, falling or just neutral. This kind of relation which can be seen when plotting scatter plots illustrates the degree of *Correlation* between the variables. Correlation is a common tool for describing simple relationships without making a statement about cause and effect and it also cannot accurately describe curvilinear relationships. (JMP Statitistical Discovery n.d.)

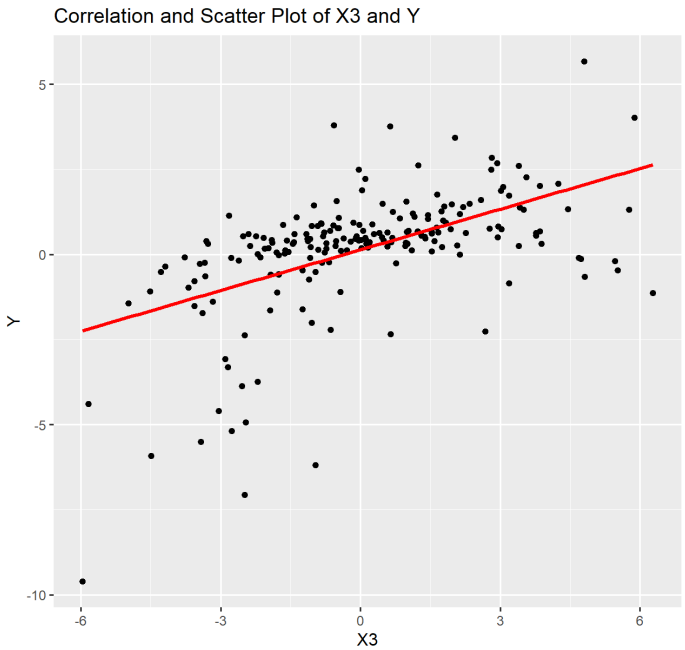
In our given dataset, the X input signals are independent variables and the Y ouput signal is a dependent variable. The scatter plot below shows the data points and the line (line of best fit) shows the correlation between the variables.

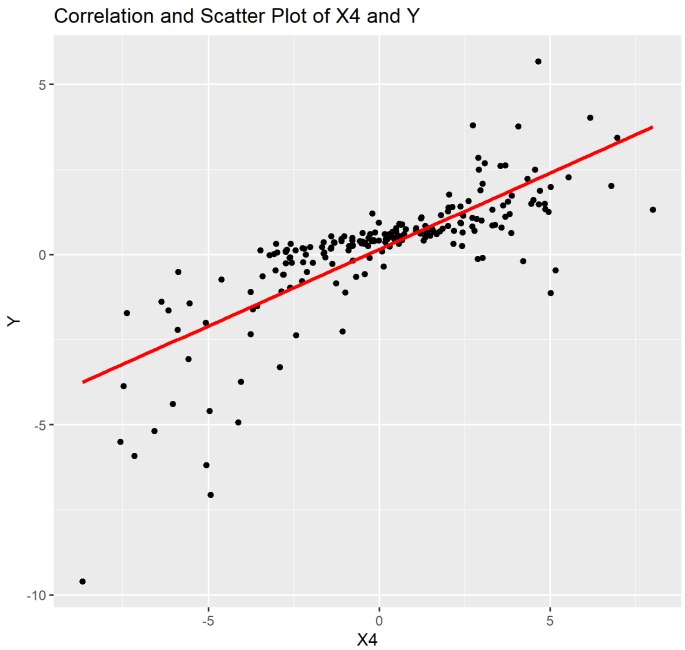




The scatter plot on the left shows the relation between the X1 or the input variable and Y or the output variable respectively. The line of best fit drawn has been drawn to to study the relationship between the variables more clearly. (Wikipedia 2023)

Both the scatter plots for X1 and X2 with respect to Y, show a positive correlation. Comparing the two, X1 and Y seem to have *higher positive correlation* than X2 and Y as the points in plot for X2 and Y are more *scattered* than in the first plot.





The points in the plot for X3 and Y are more scattered than that of X4 and Y. Both the plots show a positive correlation, whereas X4 and Y seem to have higher positive correlation.

Analysing all four scatter plots, the signals X1 and X4 seem to be have a high correlation with Y whereas for X3 and X2, the data points are a lot more scattered and even though it has a positive correlation, it is lower than that of X1 and X4.

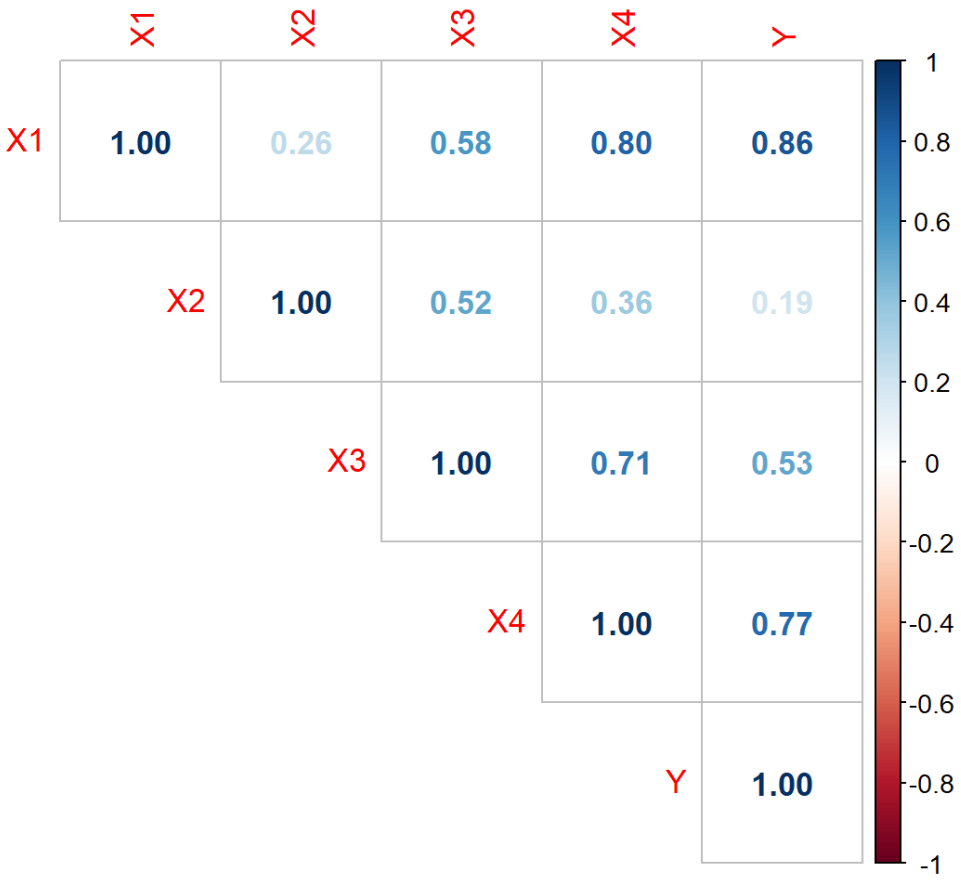


Figure 4 Correlation Plot

The correlation plot also sheds some light on the correlation between the input and output variables. The above plot has been plotten with the help of 'corrplot' library on R and the values in the plots are Pearson's Correlation Coefficients.

Scale of correlation coefficient	Value
$0 < r \leq 0.19$	Very Low Correlation
$0.2 \leq r \leq 0.39$	Low Correlation
$0.4 \leq r \leq 0.59$	Moderate Correlation
$0.6 \leq r \leq 0.79$	High Correlation
$0.8 \leq r \leq 1.0$	Very High Correlation

Figure 5 Scale of Pearson's Correlation Coefficient (Selvanathan 2020)

Looking at the correlation plot and comparing the values, X1 shows very high correlation with Y. X4 shows high and X3 shows moderate correlation. X2 shows very low correlation with Y.

Task 2

Regression – modelling the relationship between EEG signals

As we have already seen above in the scatter plots, drawing a simple linear regression line of best fit does not help much to define a model that can properly describe the relationship between the datas.

For such nonlinear data, polynomial regression model may help in defining a relationship better. Here five different nonlinear polynomial regression models have been provided. Out of the five models, the goal is the find the model which can best define the relationship.

Task 2.1

The method of Least Squares can be used to estimate parameters by minimizing the sum of squares of the difference between the observed data and the value provided by a model.

All the provided models have some parameters $\theta = \{\theta_1, \theta_2, \theta_3, \ldots, \theta_{bias}\}$ which needs to be estimated. These parameters can be estimated by using the Least squares method. Least Squares method is denoted by $\hat{\theta}$ and it can be calculated as,

$$\hat{\theta} = (X^T X)^{-1} X^T Y$$

where, X = input signal and Y = output signal data of the EEG signals.

For calculation of the $\hat{\theta}$ in R, the above equation can be formulated as,

$$\hat{\theta} = solve(t(X)\% * \%X)\% * \%t(X)\% * \%Y$$

The first non-linear polynomial regression model (model 1) is,

$$y = \theta_1 x_4 + \theta_2 {x_1}^2 + \theta_3 {x_1}^3 + \theta_4 {x_2}^4 + \theta_5 {x_1}^4 + \theta_{bias} + \varepsilon$$

The model above, in R would look like,

```
Xmodel1 <- cbind(ones,(X[,'X4']),(X[,'X1'])^2,(X[,'X1'])^3,(X[,'X2'])^4,(X[,'X1'])^4)
```

After creating a model, the value of $\hat{\theta}$ for the model is calculated using the formula shown above which, in R would look like:

```
model1_thetahat = solve(t(Xmodel1)%*%Xmodel1)%*%t(Xmodel1)%*%Y
```

The table given below is the value of ‘model1_thetahat’ calculated in R.

0.401580779	0.12771011	-0.000290217	0.009668813	-0.000409892	-0.000154337

$\hat{\theta}$ values for model 2,

0.483065688	0.143578928	0.010038614	-0.001912836

$\hat{\theta}$ values for model 3,

0.340561975	0.021330543	-0.002857744

$\hat{\theta}$ values for model 4,

0.509013488	0.053322048	0.012067145	-0.001855997

$\hat{\theta}$ values for model 5,

0.479828463	0.14344607	0.000325464	0.010056276	-0.001919875

After calculating the $\hat{\theta}$ values for all the models, those values are used to calculated the $\hat{Y}$ of all the models.

Task 2.2

Computing the Model Residual Sum of Squared Errors (RSS)

The $\hat{Y}$ values calculated in the previous step are all estimated using the models. The actual values of Y are already present in the provided dataset. All five models have their own estimated $\hat{Y}$. The difference between the actual value and the value predicted by the model is called error or residual. The error can be represented as,

$$error = Y - \hat{Y}$$

The sum of squares of all those errors is called RSS. It measures the variance in the value of the observed data when compared to its predicted value as per the regression model as also seen in the figure. (wallstreetmojo.com)

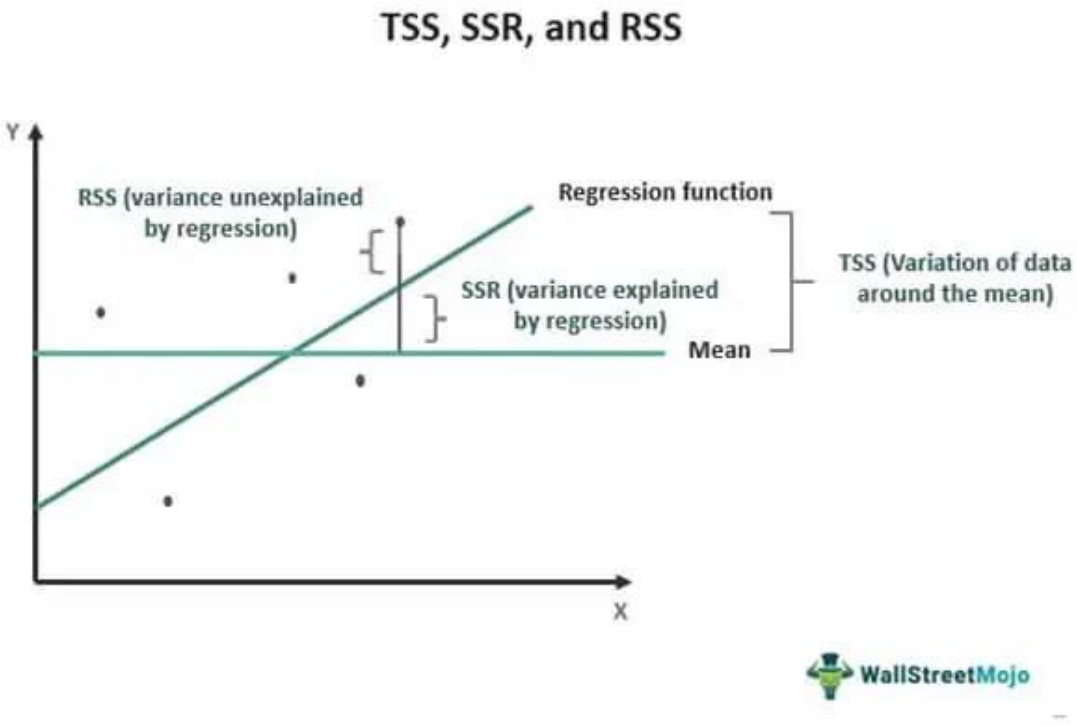


Figure 6 Residual sum of squares (wallstreetmojo.com)

It can be denoted as:

$$RSS = \sum_{i=1}^{n} (Y - \hat{Y})^2$$

where, $\hat{Y} = x_i\hat{\theta}$.

In R, the $\hat{Y}$ of model 1 is calculated as,

```
Y_hat_m1 = Xmodel1 %*% model1_thetahat
```

Similarly, the $\hat{Y}$ of all other remaining models is also calculated and then used to further calculate the RSS of all the models.

The RSS is calculated in R as,

```
RSS_Model_1=sum((Y-Y_hat_m1)^2)
```

The above equation is for calculating RSS of model 1. The RSS of remaining models is calculated in similar way.

Model	RSS Value
Model 1	35.39663
Model 2	2.139762
Model 3	463.3124
Model 4	20.259
Model 5	2.135503

The table shows the RSS values obtained in R for all the models. Generally, when calculating RSS of a model with reference to provided potential models, the model which has the minimum RSS is said to be the model which fits the actual dataset better than other models.

Taking a look at our table, the models 2 and 5 have the most minimum values of RSS comparing to all other models. So, the regression models 2 and 5 must fit the actual dataset better than any other model present.

Task 2.3

Computing the Log-Likelihood Function

The parameters are a part of the model and has a great role in the output or the actual data. Likelihood method is a measure that explains how well those parameters describes the data. It is a measure of the likeliness of getting or estimating data that resembles the actual data, given the parameters and the model.

Log-likelihood is simply a likelihood function but it uses the logs of likelihood. The log is taken generally because it is usually computationally simpler and easier to optimize. (StatisticsHowTo n.d.)

The equation for log-likelihood function is,

$$\ln p(D \mid \hat{\theta}) = -\frac{n}{2}\ln(2\pi) - \frac{n}{2}\ln(\hat{\sigma}^2) - \frac{1}{2\hat{\sigma}^2}RSS$$

Where,

$\ln p(D \mid \hat{\theta})$ is the log-likelihood.

n is the total number of Y signals.

$\hat{\sigma}^2$ is the variance, which can be calcuated using RSS from previous task as,

$$\hat{\sigma}^2 = \frac{RSS}{n-1}$$

The equation of log-likelihood can be represented and used in R as,

likelihood_1=(N/2)*(log(2*pi))(N/2)*(log(variance_model1))(1/(2*variance_model1))*RSS_Model_1

Similary calculating the log-likelihood of all the models gives the following values.

Model	Likelihood Value
1	-110.6707
2	171.3245
3	-369.1351
4	-54.58995
5	171.5247

The log-likelihood values above show that for models 2 and 5, the values are close and highest. The high values of log-likelihood in those models show that they fit the dataset better than the other models.

Taks 2.4

Computing the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC)

Selecting a good model that is not too complex but can still describe the dataset well is crucial. As adding or having more parameters in a model makes it complex whereas it might not be as explanatory as other models with lesser parameters or lesser complexity.

Akaike Information Criterion (AIC) is one of the methods which helps in choosing a better model with the help of number of prameters used in a model and the log-likelihood function. It entertains parsimoniousness in the use of number of parameters in a model as it penalizes the model more if it has more parameters and also prevents overfitting of model. (Kutz, 2020)

AIC can be represented as:

$$AIC = 2k - 2\ln(\hat{L})$$

where,

k is the number of parameters used in a model.

$\hat{L}$ is the log-likelihood function.

Bayesian Information Criterion (BIC) is a method similar to AIC but it has a different penalty factor. BIC can be represented as:

$$BIC = \ln(n)\, k - 2\ln(\hat{L})$$

As both AIC and BIC provide scores based on how complex a model is and how better the model explains the data, the model which has a smaller score than other models are considered to be performing well.

The 5 regression models have different number of parameters as already seen in previous tasks. Using it and the log-likelihood calculated in previous taks, the table below shows the values of AIC for different models.

Model	AIC Value
1	233.3414
2	-334.6489
3	744.2702
4	117.1799
5	-333.0493

The table below lists the calculated values of BIC for different models.

Model	BIC Value
1	253.1613
2	-321.4357
3	754.1801
4	130.3931
5	-316.5328

Looking at the scores from the tables above, it can be said that out of all models, model 2 and model 5 have the lowest AIC and BIC scores. Out of models 2 and 5, model 2 has even better (lower) AIC and BIC scores.

So, model 2 might be a slightly better model than model 5 based on these criterions.

Task 2.5

Distribution of Model Prediction Errors and Q-Q Plots

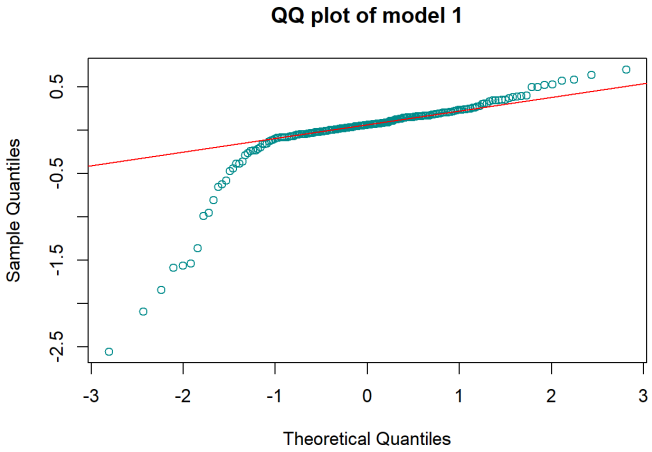
The quantile-quantile plot or generally known as Q-Q plot is a kind of tool used to determine if two datasets come from a population with same distribution.

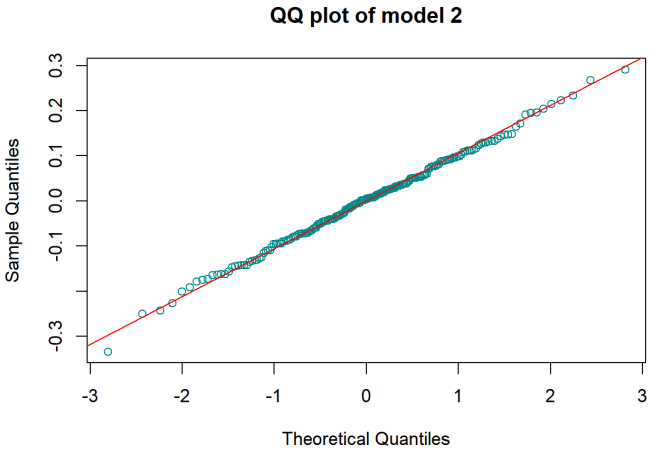
From above tasks, the error of the predicted models has been calculated. To evaluate if the distribution of the model prediction errors is close to a Normal Distribution, Q-Q plots can be used. In this case, the model prediction error data needs to be plotted in the Normal Q-Q plot against the theoritical quantiles. The theoritical quantiles are plotted in the horizontal axis and the sample quantiles are plotted on the vertical axis.

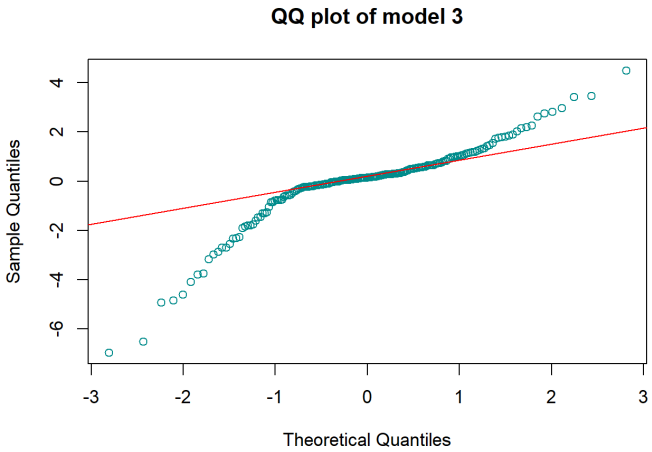
Here, we are comparing the model prediction errors data and theoritical data. If they are of the same type of distribution, the datas will align in a somewhat straight line.

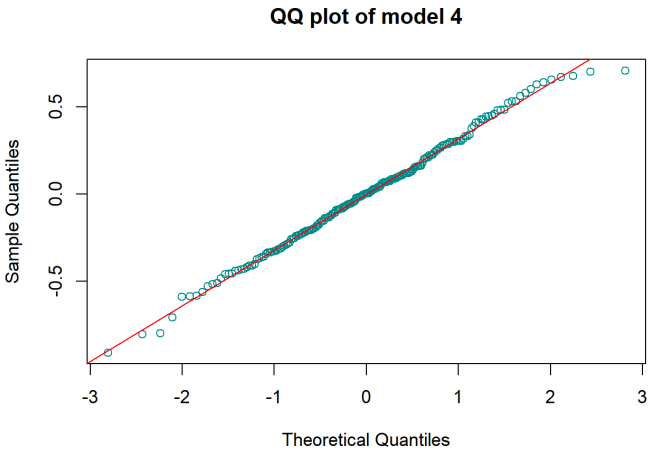
As we need to evaluate if the model prediction error follows a normal distribution, qqnorm can be used.

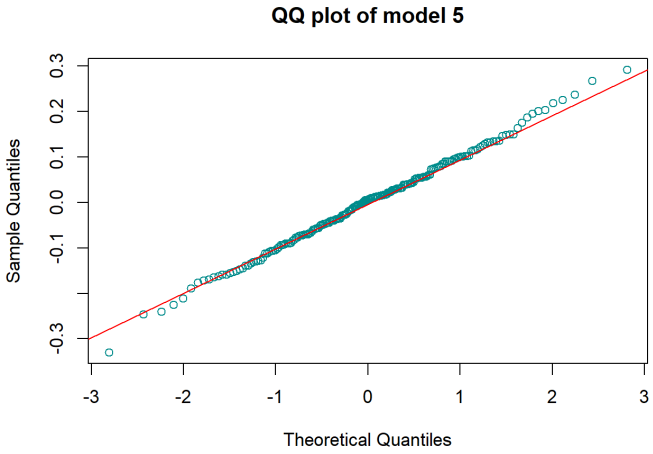
Below we can see the Q-Q plots for all the models.











As we can clearly see, the q-q plot of the models 2, 4 and 5 have points which align almost fully to a straight line with just some points that are off the line. It means that, these models' quantiles and the theoritical quantiles have the same distribution, that is a Normal Distribution.

Whereas from the plots of model 1 and model 3, we can clearly see that there are many points off the straight line and is skewed, showing they don't follow a Normal Distribution.

Task 2.6

Selecting the best regression model based on AIC, BIC and distribution of model residuals

The AIC and BIC scores are important as it helps in selecting a model which strikes a perfect balance between the model's complexity and how well the model fits the data or is able to explain the dataset. Revisiting the values of AIC and BIC of the models, the models 2 and 5 have the best (lowest) scores among all the models.

AIC OF MODEL 2	-334.6489
AIC OF MODEL 5	-333.0493

From the table, Model 2 has lower AIC than model 5. As lower is better, selection of model 2 is a better choice.

BIC OF MODEL 2	-321.4357
BIC OF MODEL 5	-316.5328

From the table it is clear that model 2 has a lower BIC score. So, model 2 is a better choice in this regard as well.

Also considering the RSS value, models 2 and 5 have the best values.

Looking at the results from Task 2.5, even though models 2, 4 and 5, all follow Normal Distribution, model 2 shows the best fit. Taking all the results into consideration, model 2 seems to be the best regression model. So, model 2 is to be used for later tasks and calculation.

Task 2.7

Splitting the input and output datasets into train and test sets

Here the original dataset has been splitted to training and testing sets such that random 70% of the data goes to the training set and remaining 30% are left for the testing set. For consistent results in several runs seed can be set as well which would ensure the same random values every run.

Task 2.7.(1-2)

Estimating the model parameters using the Training Dataset and computing the model's prediction on Testing Data

The training data split has been used to estimate the parameters of the model 2. The parameters estimated using the training set is then used to compute the model's prediction on the test data which in turn produces a predicted Y for the model 2.

The predicted Y value comes from the use of the test set, comparing it to the actual values of Y from the test set provides an error value. This error value helps in determining how well the model works when the output is unknown. The lesser the error, the better the model is working.

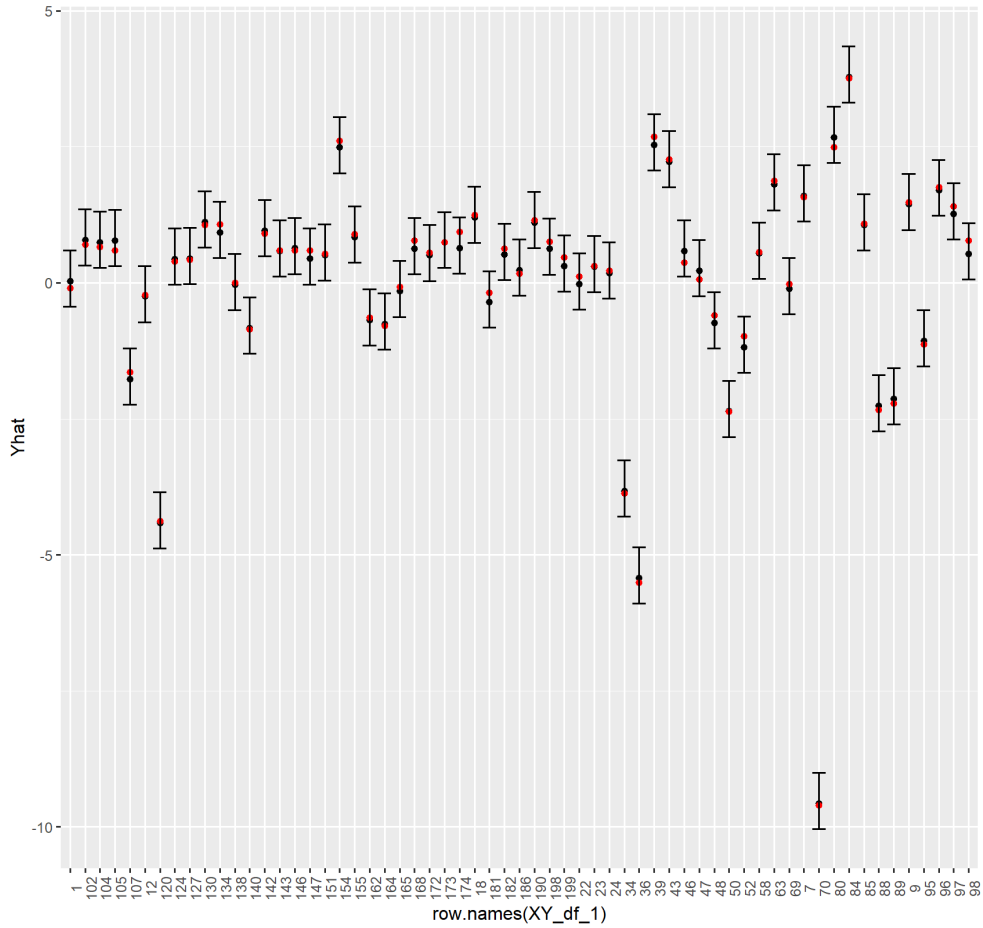
RSS (Residual Sum of Squares) calculation comes out to be 0.5940319, which is a very good value as it indicates that the model has performed well on testing values.

For calculating the 95% confidence intervals of the model prediction, t-values has been used using the equation:

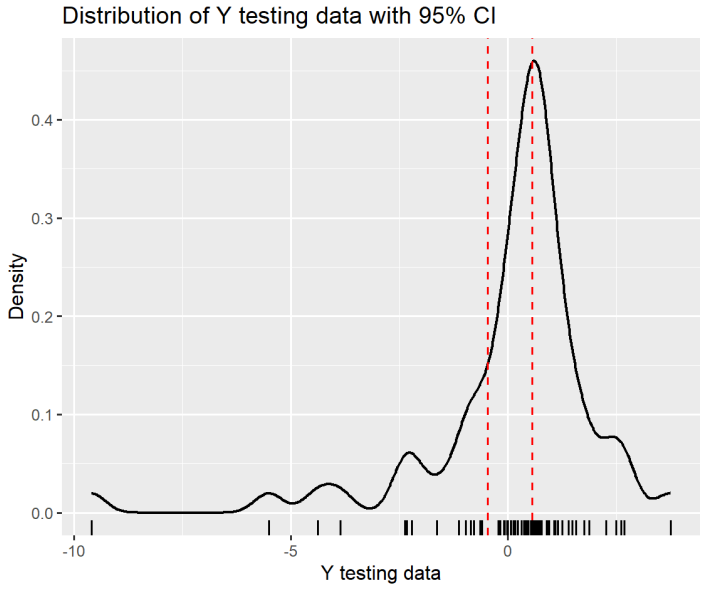
$$\bar{x} \pm t - value \times \frac{s}{\sqrt{n}}$$

Where,

$\bar{x}$ is the sample mean, t-value comes form the t-table for 95% confidence, s is the standard deviation of the sample and n is the no. of samples. The calculated value of intervals is (-0.4701331, 0.5636087). Using these values, the model prediction values with error plots can be plotted using the test data samples as shown in the figure below.



The plot above shows the values of the model prediction plotted along with their confidence intervals. The black points represent the predicted values whereas the red points represent the acutal test values. The x-labels shows the label of the test sample. Analysing the above plot, it can be seen that values are well predicted in relation to the actual test values. The plot below shows the distribution of Y values from testing set with the confidence intervals.



Task 3

Computing Posterior Distribution of Model 2 using rejection Approximate Bayesian Computation (ABC)

Approximate Bayesian Computation is a statistical framework also referred to as 'likelihood-free inteference' which is a tool used when the likelihood is not known or cannot be computed.

From the selected model 2, the parameters with the highest values are selected and used for forming a uniform stochastic distribution as a prior distribution. The values theta 1 and theta 2 are chosen.

	Theta 1 (Highest)	**Theta 2**	**Theta 3**	**Theta 4 (bias)**
Values	0.483065688 (selected)	0.143578928 (selected)	0.010038614 (constant)	-0.001912836 (constant)

Figure 7 Parameters selection for uniform distribution

A threshold value ε (Epsilon, here computed as RSS X 2) also needs to be set. Using these values and remaining parameters produces a new output. The difference between the new predicted value and the actual observed value is calculated and compared to ε. This process is repeated for a selected number of iterations. If the error is greater than ε, the value is rejected. The distribution of the accepted values gives the posterior distribution. (ModelAssist n.d.) Performing 100 iterations, the function produces the posterior values which has been plotted below in the form of joint and marginal posterior distribution.

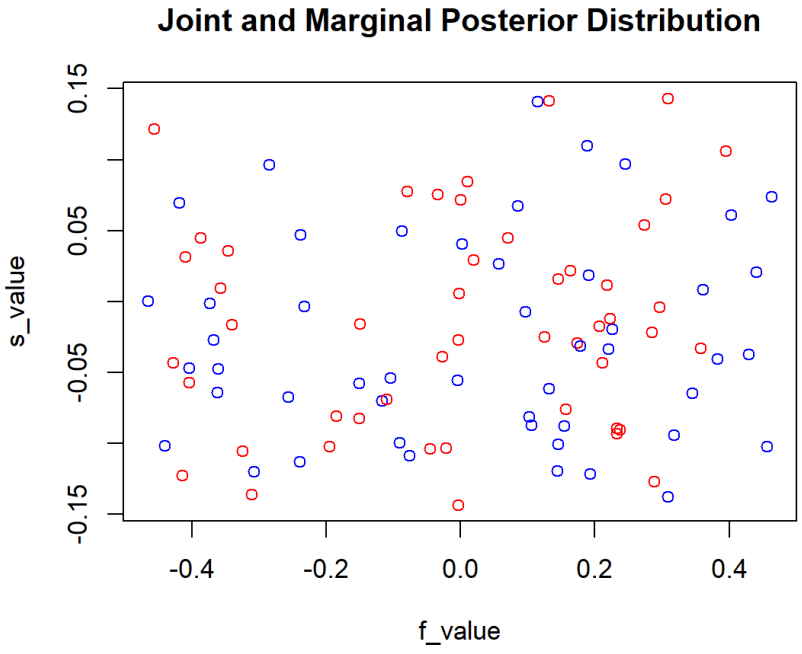
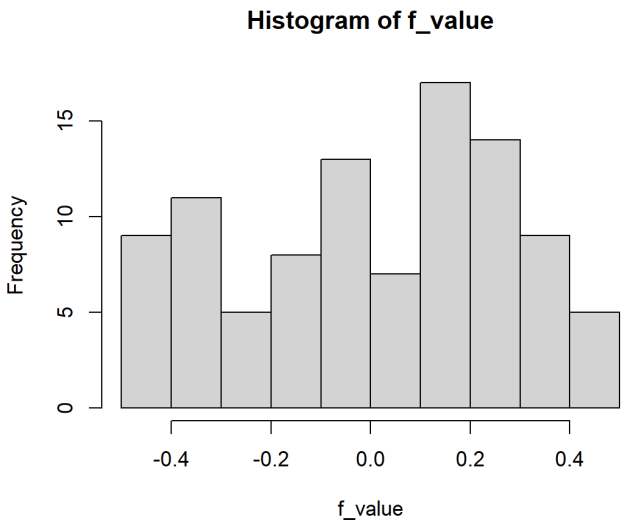
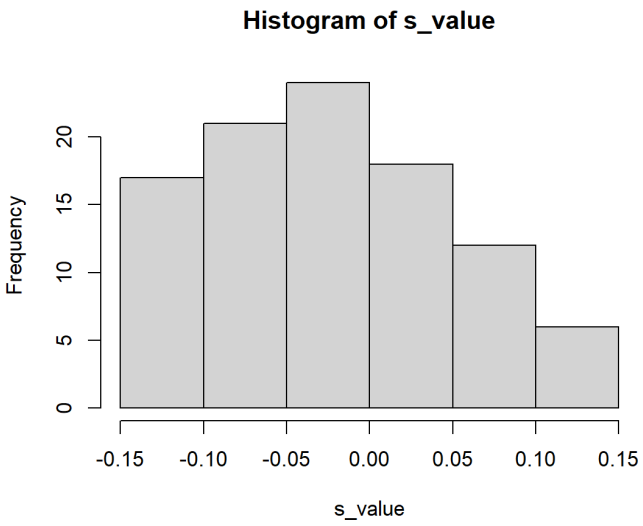


Figure 8 Joint and Marginal Posterior Distribution





Using the rejection ABC algorithm, the posterior of the model was calculated. The parameters of the model two with highets values were used for simulating datas. It can be seen from the results that the value of the parameters explained the data well which is what was needed to be checked.

Conclusion

Analysing the data provided thee model 2 was selected based on the calculation of RSS, AIC, BIC and the rejection ABC method was also performed which showed that the model parameters explained the data well.

Appendix

```
# set working directory first to source file location
library(matlib)
library(ggplot2)
library(rsample)
#import X data
X = as.matrix(read.csv("X.csv", header = F))
colnames(X) <- c("X1", "X2", "X3", "X4")

#import Y data
Y = as.matrix(read.csv("y.csv", header = F))
colnames(Y) <- c("Y")

#import time data
time = read.csv("time.csv", header = F, skip = 1)
time = as.matrix(rbind(0, time))

#Creating Time series Plot
#create time series objects
X.ts <- ts(X, start = c(min(time), max(time)),frequency = 1)
Y.ts <- ts(Y, start = c(min(time), max(time)),frequency = 1)

#plotting the graphs
plot(X.ts, main = "Time series plot of X signal", xlab = "Time", ylab = "Input Signal")
plot(Y.ts, main = "Time series plot of Y signal", xlab = "Time", ylab = "Output Signal")

#Creating density plot
#density plot of X signal
dis = density(X)
#par(mfrow = c(1,2))
#plot(dis, main = "Density Plot of Input Signal")
#histogram of X signal
#hist(X, freq = FALSE, main = "Histogram Plot of Input Signal")

#both density and histogram
par(mfrow = c(1,1))
hist(X, freq = FALSE, main = "Density and Histogram Plot of Input Signal")
lines(dis, lwd = 2, col = "tomato2")
rug(jitter(X))

#density plots using GGPLOT
```

```
Xdf = read.csv("X.csv")
colnames(Xdf) <- c("X1", "X2", "X3", "X4")

#unlist all columns to one for plotting single plot for X
Xdfunlist <- data.frame(unlist(Xdf))
nrow(Xdfunlist)
colnames(Xdfunlist) <- c("X")
ggplot(Xdfunlist, aes(x=X))+
  geom_histogram(aes(y=after_stat(density)), colour="black", fill="white")+
  geom_rug()+
  geom_density(alpha=.1, fill="#FFE1FF")+
  ggtitle("Histogram and Density Plot of Input Signal X")

#density plots and histogram plots of individual signals
ggplot(Xdf, aes(x=X1))+
  geom_histogram(aes(y=after_stat(density)), colour="black", fill="white")+
  #geom_vline(aes(xintercept=mean(Y)), color="#8B3626", linetype="dashed", size=0.4)+
  geom_rug()+
  geom_density(alpha=.1, fill="#FFE1FF")+
  ggtitle("Histogram and Density Plot of Input Signal X1")
ggplot(Xdf, aes(x=X2))+
  geom_histogram(aes(y=after_stat(density)), colour="black", fill="white")+
  geom_rug()+
  geom_density(alpha = .1, fill="#FFB90F")+
  ggtitle("Histogram and Density Plot of Input Signal X2")
ggplot(Xdf, aes(x=X3))+
  geom_histogram(aes(y=after_stat(density)), colour="black", fill="white")+
  geom_rug()+
  geom_density(alpha = .1, fill="#87CEFF")+
  ggtitle("Histogram and Density Plot of Input Signal X3")
ggplot(Xdf, aes(x=X4))+
  geom_histogram(aes(y=after_stat(density)), colour="black", fill="white")+
  geom_rug()+
  geom_density(alpha = .1, fill="#FFBBFF")+
  ggtitle("Histogram and Density Plot of Input Signal X4")

#density plot of X1 signal
dis_X1 = density(X["X1"])
hist(X["X1"], freq = FALSE, main = "Histogram Plot of X1")
lines(dis_X1, lwd = 2, col = "orange")
rug(jitter(X["X1"]))
```

```
#density plot of X2 signal
dis_X2 = density(X[,"X2"])
hist(X[,"X2"], freq = FALSE, main = "Histogram Plot of X2")
lines(dis_X2, lwd = 2, col = "cadetblue")
rug(jitter(X[,"X2"]))

#density plot of X3 signal
dis_X3 = density(X[,"X3"])
hist(X[,"X3"], freq = FALSE, main = "Histogram Plot of X3")
lines(dis_X3, lwd = 2, col = "aquamarine2")
rug(jitter(X[,"X3"]))

#density plot of X4 signal
dis_X4 = density(X[,"X4"])
hist(X[,"X4"], freq = FALSE, main = "Histogram Plot of X4")
lines(dis_X4, lwd = 2, col = "seagreen")
rug(jitter(X[,"X4"]))

#density plot of Y signal
dis_Y = density(Y)
hist(Y, freq = FALSE, main = "Histogram Plot of Y")
lines(dis_Y, lwd = 2, col = "purple")
rug(jitter(Y))

#density plot of Y using GGPLOT
Ydf = read.csv("Y.csv")
colnames(Ydf) <- c("Y")
ggplot(Ydf, aes(x=Y))+
  geom_histogram(aes(y=after_stat(density)), size= 0.3, colour="black", fill="white")+
  geom_vline(aes(xintercept=mean(Y)), color="darkgreen", linetype="dashed", size=0.4) +
  geom_rug()+
  geom_density(size = 0.5, alpha = .1, fill="#FF7F50")+
  ggtitle("Histogram and Density Plot of Output Signal")

#scatter plot between X1 and Y
par(mfrow = c(2,2))
plot(X[,"X1"], Y, main = "Scatter Plot: X1 and Y signal", xlab = "X1 signal",
     ylab = "Output Signal")
abline(lm(X['X1'] ~ Y), col = 'red')
#scatter plot between X2 and Y
```

```
plot(X["X2"], Y, main = "Scatter Plot: X2 and Y signal", xlab = "X2 signal",
    ylab = "Output Signal")
abline(lm(X['X2'] ~ Y), col = 'red')
#scatter plot between X3 and Y
plot(X["X3"], Y, main = "Scatter Plot: X3 and Y signal", xlab = "X3 signal",
    ylab = "Output Signal")
abline(lm(X['X3'] ~ Y), col = 'red')
#scatter plot between X4 and Y
plot(X["X4"], Y, main = "Scatter Plot: X4 and Y signal", xlab = "X4 signal",
    ylab = "Output Signal")
abline(lm(X['X4'] ~ Y), col = 'red')

#using ggplot
XY_table <- as.data.frame(cbind(X,Y))

ggplot(XY_table, aes(x = X1, y=Y))+
  geom_point()+
  geom_smooth(method = lm, se= FALSE, colour = "red")+
  ggtitle("Correlation and Scatter Plot of X1 and Y")
ggplot(XY_table, aes(x = X2, y=Y))+
  geom_point()+
  geom_smooth(method = lm, se= FALSE, colour = "red")+
  ggtitle("Correlation and Scatter Plot of X2 and Y")
ggplot(XY_table, aes(x = X3, y=Y))+
  geom_point()+
  geom_smooth(method = lm, se= FALSE, colour = "red")+
  ggtitle("Correlation and Scatter Plot of X3 and Y")
ggplot(XY_table, aes(x = X4, y=Y))+
  geom_point()+
  geom_smooth(method = lm, se= FALSE, colour = "red")+
  ggtitle("Correlation and Scatter Plot of X4 and Y")

#correlation
#install.packages("corrplot")
par(mfrow = c(1,1))
library(corrplot)
cor_df = cbind(X, Y)
cor_tab = cor(cor_df)
corrplot(cor_tab, method = "number", type = "upper")
```

```
#TASK 2
ones = matrix(1 , nrow(X), 1)
ones
#2.1
#estimating model parameters using Least squares theta-hat
# thetahat <- Inverse(Xtr %*% X) %*% Xtr %*% Y

Xmodel1 <- cbind(ones,(X[,'X4']),(X[,'X1'])^2,(X[,'X1'])^3,(X[,'X2'])^4,(X[,'X1'])^4)
model1_thetahat = solve(t(Xmodel1)%*%Xmodel1)%*%t(Xmodel1)%*%Y
write.csv(model1_thetahat, "results/model1thetahat.csv")

Xmodel2 <- cbind(ones,(X[,'X4']),(X[,'X1'])^3,(X[,'X3'])^4)
model2_thetahat = solve(t(Xmodel2)%*%Xmodel2)%*%t(Xmodel2)%*%Y
write.csv(model2_thetahat, "results/model2thetahat.csv")

Xmodel3 <- cbind(ones,(X[,'X3'])^3,(X[,'X3'])^4)
model3_thetahat = solve(t(Xmodel3)%*%Xmodel3)%*%t(Xmodel3)%*%Y
write.csv(model3_thetahat, "results/model3thetahat.csv")

Xmodel4 <- cbind(ones,(X[,'X2']),(X[,'X1'])^3,(X[,'X3'])^4)
model4_thetahat = solve(t(Xmodel4)%*%Xmodel4)%*%t(Xmodel4)%*%Y
write.csv(model4_thetahat, "results/model4thetahat.csv")

Xmodel5 <- cbind(ones,(X[,'X4']),(X[,'X1'])^2,(X[,'X1'])^3,(X[,'X3'])^4)
model5_thetahat = solve(t(Xmodel5)%*%Xmodel5)%*%t(Xmodel5)%*%Y
write.csv(model5_thetahat, "results/model5thetahat.csv")

#Calculating Y-hat and RSS Model 1
Y_hat_m1 = Xmodel1 %*% model1_thetahat
Y_hat_m1
#Calculating RSS
RSS_Model_1=sum((Y-Y_hat_m1)^2)
RSS_Model_1
#Calculating Y-hat and RSS of model 2
Y_hat_m2 = Xmodel2 %*% model2_thetahat
Y_hat_m2
RSS_Model_2=sum((Y-Y_hat_m2)^2)
RSS_Model_2
#Calculating Y-hat and RSS of model 3
Y_hat_m3 = Xmodel3 %*% model3_thetahat
Y_hat_m3
```

```
RSS_Model_3=sum((Y-Y_hat_m3)^2)
RSS_Model_3
#Calculating Y-hat and RSS of model 4
Y_hat_m4 = Xmodel4 %*% model4_thetahat
Y_hat_m4
RSS_Model_4=sum((Y-Y_hat_m4)^2)
RSS_Model_4
#Calculating Y-hat and RSS of model 5
Y_hat_m5 = Xmodel5 %*% model5_thetahat
Y_hat_m5
RSS_Model_5=sum((Y-Y_hat_m5)^2)
RSS_Model_5


#calculating variances for all models
N = length(Y)
variance_model1 = RSS_Model_1/(N-1)
variance_model2 = RSS_Model_2/(N-1)
variance_model3 = RSS_Model_3/(N-1)
variance_model4 = RSS_Model_4/(N-1)
variance_model5 = RSS_Model_5/(N-1)

#calculation of loglikelihood
#for model 1
likelihood_1 = -(N/2)*(log(2*pi))-(N/2)*(log(variance_model1))-(1/(2*variance_model1))*RSS_Model_1
likelihood_1
#for model 2
likelihood_2 = -(N/2)*(log(2*pi))-(N/2)*(log(variance_model2))-(1/(2*variance_model2))*RSS_Model_2
likelihood_2
#for model 3
likelihood_3 = -(N/2)*(log(2*pi))-(N/2)*(log(variance_model3))-(1/(2*variance_model3))*RSS_Model_3
likelihood_3
#for model 4
likelihood_4 = -(N/2)*(log(2*pi))-(N/2)*(log(variance_model4))-(1/(2*variance_model4))*RSS_Model_4
likelihood_4
#for model 5
likelihood_5 = -(N/2)*(log(2*pi))-(N/2)*(log(variance_model5))-(1/(2*variance_model5))*RSS_Model_5
likelihood_5

#calculation of AIC (Akaike Information Criterion) and BIC
##Calculating AIC and BIC of model 1
```

```
K_model1<-length(model1_thetahat)
K_model1
AIC_model1=2*K_model1-2*likelihood_1
AIC_model1
BIC_model1=K_model1*log(N)-2*likelihood_1
BIC_model1

##Calculating AIC and BIC of model 2
K_model2<-length(model2_thetahat)
K_model2
AIC_model2=2*K_model2-2*likelihood_2
AIC_model2
BIC_model2=K_model2*log(N)-2*likelihood_2
BIC_model2

##Calculating AIC and BIC of model 3
K_model3<-length(model3_thetahat)
K_model3
AIC_model3=2*K_model3-2*likelihood_3
AIC_model3
BIC_model3=K_model3*log(N)-2*likelihood_3
BIC_model3

##Calculating AIC and BIC of model 4
K_model4<-length(model4_thetahat)
K_model4
AIC_model4=2*K_model4-2*likelihood_4
AIC_model4
BIC_model4=K_model4*log(N)-2*likelihood_4
BIC_model4

##Calculating AIC and BIC of model 5
K_model5<-length(model5_thetahat)
K_model5
AIC_model5=2*K_model5-2*likelihood_5
AIC_model5
BIC_model5=K_model5*log(N)-2*likelihood_5
BIC_model5

## Task 2.5
## Error of model1
```

```
par(mfrow = c(1,1))
model1_error <- Y-Y_hat_m1
## Plotting the graph QQplot and QQ line of model 1
qqnorm(model1_error, col = "darkcyan",main = "QQ plot of model 1")
qqline(model1_error, col = "red",lwd=1)

model2_error <- Y-Y_hat_m2
qqnorm(model2_error, col = "darkcyan",main = "QQ plot of model 2")
qqline(model2_error, col = "red",lwd=1)

model3_error <- Y-Y_hat_m3
qqnorm(model3_error, col = "darkcyan",main = "QQ plot of model 3")
qqline(model3_error, col = "red",lwd=1)

model4_error <- Y-Y_hat_m4
qqnorm(model4_error, col = "darkcyan",main = "QQ plot of model 4")
qqline(model4_error, col = "red",lwd=1)

model5_error <- Y-Y_hat_m5
qqnorm(model5_error, col = "darkcyan",main = "QQ plot of model 5")
qqline(model5_error, col = "red",lwd=1)

#Task 2.6
## Chose model 2 based on the values of AIC and BIC, (Lower the better), and low RSS means
## better fit of the regression equation for the data

### Task 2.7 ## Binding X and Y table into one
XY_table <- cbind(X,Y)
set.seed(1353)
split_XY<-initial_split(data = as.data.frame(XY_table),prop=.7)

XY_training_df <- training(split_XY)
XY_testing_df <- testing(split_XY)
XY_training_set<-as.matrix(training(split_XY) )
XY_testing_set<-as.matrix(testing(split_XY))


### Estimating model parameters using Training set
training_ones=matrix(1 , nrow(XY_training_set),1)
#model ->ones,(X[,'X4']),(X[,'X1'])^3,(X[,'X3'])^4
X_training_model<-cbind(training_ones,XY_training_set[,"X4"],(XY_training_set[,"X1"])^3,(XY_training_set[ ,"X3"])^4)
```

```
training_thetahat=solve(t(X_training_model) %*% X_training_model) %*% t(X_training_model) %*%
XY_training_set[,"Y"]

### Model output or model prediction
#creating X testing model
testing_ones = matrix(1, length(XY_testing_set[,"X1"]), 1)
#selecting X_test_set variables needed as per model 2
X_testing_model = cbind(testing_ones,XY_testing_set[,"X4"],(XY_testing_set[,"X1"])^3,(XY_testing_set[ ,"X3"])^4)
#computing model output on testing data model using the training_thetahat parameters
Y_testing_hat = X_testing_model%*% training_thetahat
Y_testing_hat
RSS_testing=sum((XY_testing_set[,"Y"]-Y_testing_hat)^2)
RSS_testing

#calculating 95% confidence intervals of the predicted y_testing_hat
#using the t-distribution
#no. of independent variables in model 2 = 3
t <- 2     # for n-1, 61-1=60 degree of freedom
sample_sd <- sd(Y_testing_hat)
sample_size <- nrow(Y_testing_hat)
sample_mean <- mean(Y_testing_hat)

c_i_pos_t <- sample_mean + t * (sample_sd/sqrt(sample_size))
c_i_neg_t <- sample_mean - t * (sample_sd/sqrt(sample_size))

XY_df_1 <- as.data.frame.matrix(cbind(XY_testing_df[,"Y"],  Y_testing_hat, Y_testing_hat + c_i_neg_t, Y_testing_hat
+ c_i_pos_t))
colnames(XY_df_1) <- c("Y","Yhat","lower","upper")


ggplot(XY_df_1, aes(x=row.names(XY_df_1)))+
  geom_point(aes(y = Yhat))+
  geom_point(aes(y=Y), color = "red")+
  geom_errorbar(aes(ymin = lower, ymax=upper))+
  theme(axis.text.x = element_text(angle = 90))

#density plot of Y testing data with confidence intervals
ggplot(data = data.frame(XY_testing_df[,"Y"]), aes(XY_testing_df[,"Y"])) +
  geom_density(col = "black" , lwd=0.7) +
  geom_vline(xintercept = c_i_neg_t, col = "red", linetype = "dashed") +
  geom_vline(xintercept = c_i_pos_t, col = "red", linetype = "dashed") +
```

```
  geom_rug()+
  ggtitle("Distribution of Y testing data with 95% CI")+
  xlab("Y testing data") +
  ylab("Density")

#task 3
## Model 2 will be used, parameter are selected and kept constant.
arr_1=0
arr_2=0
f_value=0
s_value=0
model2_thetahat

#values from thetahat
thetaone <- 0.483065688 # chosen parameter #largest
thetatwo <- 0.143578928 # chosen parameter #2nd largest
thetathree <- 0.010038614 # constant value
thetabias <- -0.001912836 # constant value

Epsilon <- RSS_Model_2 * 2 ## fixing value of epsilon
num <- 100 #number of iteration

##Calculating Y-hat for performing rejection ABC
counter <- 0
for (i in 1:num) {
  range1 <- runif(1,-0.483065688,0.483065688) # calculating the range using uniform stochastic model
  range2 <- runif(1,-0.143578928,0.143578928)
  New_thetahat <- matrix(c(range1,range2,thetathree,thetabias))
  New_Y_Hat <- Xmodel2 %*% New_thetahat ## New Y hat
  new_RSS <- sum((Y-New_Y_Hat)^2)
  new_RSS
  if (new_RSS > Epsilon){
    arr_1[i] <- range1
    arr_2[i] <- range2
    counter = counter+1
    f_value <- matrix(arr_1)
    s_value <- matrix(arr_2)
  }
}

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

```
hist(f_value)
hist(s_value)
###plotting the graph
par(mfrow = c(1,1))
plot(f_value,s_value, col = c("red", "blue"), main = "Joint and Marginal Posterior Distribution")
```

References

Wikipedia (2023) *Scatter Plot* [online] available from < https://en.wikipedia.org/wiki/Scatter_plot> [4 Feb 2023]

JMP Statitistical Discovery (n.d) *Correlation* [online] available from <https://www.jmp.com/en_ca/statistics-knowledge-portal/what-is-correlation.html#:~:text=Correlation%20is%20a%20statistical%20measure,statement%20about%20cause%20and%20effect.> [4 Feb 2023]

CFA Institute (n.d.) *Residual Sum of Squares* [online] available from < https://www.wallstreetmojo.com/residual-sum-of-squares/> [5 Feb 2023]

G. Stephanie (n.d.) *Log Likelihood Function* [online] available from < https://www.statisticshowto.com/log-likelihood-function/> [5 Feb 2023]

K. Nathan (2020) *Model selection: Information criteria.* Youtube [online video] available from <https://www.youtube.com/watch?v=Gc9EzmfcSas&ab_channel=NathanKutz> [5 Feb 2023]

ModelAssist (n.d.) *Approximate Bayesian Computation (ABC) rejection algorithm* [online] available from < https://modelassist.epixanalytics.com/display/EA/Approximate+Bayesian+Computation+%28ABC%29+rejection+algorithm> [5 Feb 2023]

Selvanathan, Mahiswaran & Jayabalan, Neeta & Saini, Gurmeet & Supramaniam, Mahadevan & Hussain, Norasmahani. (2020*). Employee Productivity in Malaysian Private Higher Educational Institutions.. PalArch's Journal of Archaeology of Egypt/ Egyptology*. 17. 66-79. 10.48080/jae.v17i3.50. [5 Feb 2023]