**Module Title - Introduction to Statistical Methods for Data Science**

**Coursework Title - Modelling brain signals using nonlinear regression**

**Submitted By**

**Arjun Jayan**

**ID – 12577022**

**Lecture**

**Dr.Fei He**

**TABLE OF CONTENTS**

1. Introduction………………………………………………………............4

**2**. **Task 1: Preliminary Data Analysis……………………………………...4**

Installing Common packages…………………………………………………………....**4**

Installing Common Packages……………………………………………………………**4**

Installing Date and Time-Series Packages……………………………………………….**4**

Setting Working Environment Directory………………………………………………..**4**

The setwd()……………………………………………………………………………...**4**

Importing Data set……………………………………………………………………….**5**

Combining Output and Input Signal and converting Data to Time Series………………**5**

Converting the Time-series Data into a Data Frame Structure………………………….**6**

Distribution of Output and Input Signal by Use of a Histogram………………………...**7**

Check for Correlation……………………………………………………………..……**10**

The Scatter Plot…………………………………………………………………………**11**

Boxplot Representation…………………………………………………………………**12**

3. Task 2:Regression – Modelling The Relationship Between MEG

Signals…………………………………………………………….13

Estimating Model Parameters Using Least Squares…………………………………….**13**

Residual Sum of Squared Error…………………………………………………………**13**

The log-likelihood Function…………………………………………………………….**15**

The AIC & BIC………………………………………………………………………….**16**

Distribution of Model Prediction Errors………………………………………………**...18**

The Best Model………………………………………………………………………….**23**

Splitting the Data………………………………………………………………………..**.23**

Estimating Model Parameter……………………………………………………………..**24**

4.Task3:Approximation Bayesian Computation…………………………..27

Absolute Values…………………………………………………………………………..**27**

Range of Prior Distribution………………………………………………………………**28**

Rejection of ABC………………………………………………………………………...**28**

Plot Joint Distribution of parameters input 3 & input 5………………………………….**29**

Plot for Marginal Distribution for input 3 parameters……………………………………**29**

Plot for Marginal Distribution for input 5 Parameters……………………………………**30**

Summary of the Tasks……………………………………………………………………**31**

5. References……………………………………………………………..32

6. Appendix………………………………………………………………33

1. **Introduction**

This report aims to get a clear vision of what is entailed in all of my codes. It is a description of regression models that will be used to identify and measure the auditory-brain interaction and effect of the emotional narration. A hypothesis will help to explain how emotional narration will evoke the brain response. There is a visual representation of the data followed by clearly explained codes.

1. **Task 1: Preliminary Data Analysis**

**Installing Common Packages**

Installing the common packages that will help in the data interpretation and visualization. Package **ggplot2** for assistance to create complex plots from data in a data frame. The package **tidyverse** is suited to make the data operations more friendly. The **dplyr** function comes with a consistent set of verbs that helps in solving data manipulations.

**Installing Data and Time-series Packages**

For a time-series data set, it was necessary to import the **time series** packages. The **lubridate** package makes it easier to work with dates and times in the time series set of data. The **tseries** package is a crucial package used in the creation of time series objects. Another useful times series package is the **car** package (Companion to Applied Regression). It complements the techniques by providing functions that perform the test, creates visualizations, and transform data.

**Setting Working Environment Directory**

**The setwd()**

Setting work directory environment in R that will be used to read and save files. To set a working directory a built-in function is used in r. The setwd ()

**Importing Data set**

Importing the output signal, input signal and time data in the R environment using the read.csv () inbuilt function. The input signal is read and stored in a variable input while the output signal is stored in the output variable proceeding to the time data stored in the times variable.

**Combining Output and Input signal and converting Data to Time Series**

For a good analysis, it was useful to combine the two data set output signals and input signals using a combining function **cbind ()** that will combine them into a data set in a column form and then store it in a variable meg. After combining the data, there is a need to convert the data to time series. For this purpose, the built-in function **ts()** is used for the data that is chosen for the time-series data to be of the interval of 10 seconds.

**meg<-cbind(output, input)**

**meg1<-ts(meg,end = 20,frequency = 10,start = 0.1)**

**head(meg1)**

**Graphical user interface, application

Description automatically generated**

**Converting the Time-Series Data into a Data Frame Structure**

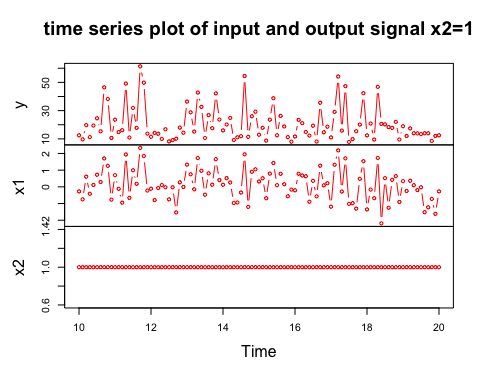
The visualization of the data needs to be improved. Here **as.data.frame()** is applied to get the data frame. the data frame quality is improved by making it visualizable into a plotted structure. There is a need to introduce r verbs like the filter verb to help the data manipulations.

**Time Series Time series plotting for x2=0 (neutral audio)**

Chart

Description automatically generated with medium confidence

**Time series plotting for x2=1 (Emotional audio)**



**Distribution of Output and Input Signal by Use of a Histogram**

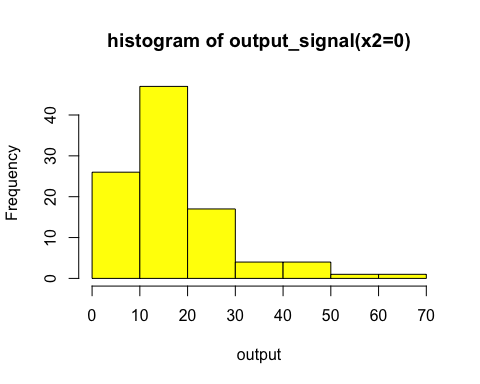
From the data frame created, it is needed to find out how the signals are distributed. The **select()** verb is used to only select the sections needed. the code needs to be stored in a variable for the case the **megdist** was used as the variable. After selecting the y,x1,x2, and filtering where x2 is 0. Here filtering is used to filter data where specific x2 is 0 and selecting only the y and x1 storing them in the variable megdist. After successfully creating the variable, the distributions of the data are seen now and for that, a data visual through histogram is created, the **hist()** is greatly useful. A similar method is used to filter where x2 is 1 and stored them in the **megdist1.**

**Histogram of an output signal**

Chart, histogram

Description automatically generated

**Histogram of the Output signal when x2=0**

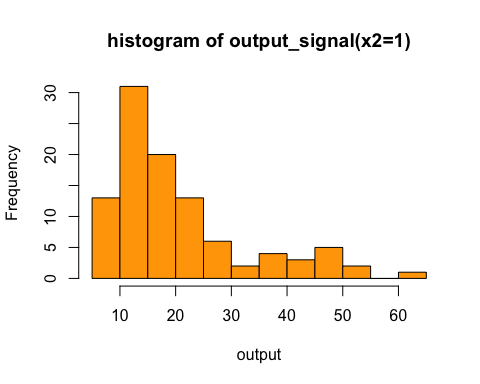


**Histogram of the Input signal when x2=0**

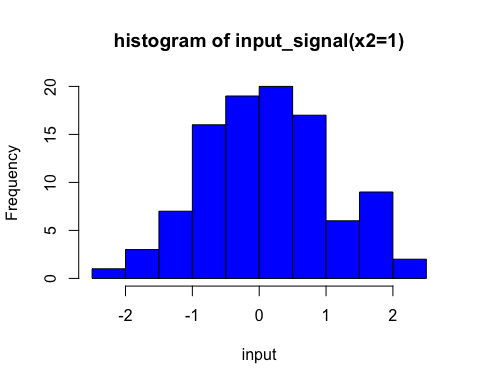
Chart, histogram

Description automatically generated

**Histogram of the Output signal when x2=1**



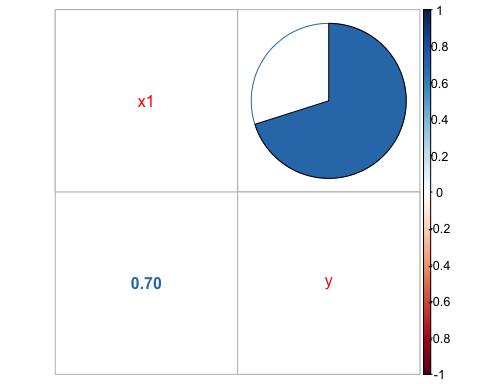
**Histogram of the Input signal when x2=1**



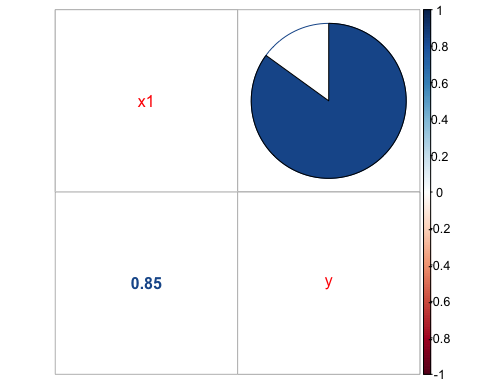
**Check for Correlation**

Given two or more data sets it is in the best interest to check for the correlation between them. For checking there is a need to run a correlation test the **cor()** function is used to test. For running the test, the cor () on the megdist variable is used. In this case, the correlation will be represented by a pie plot style.

**Correlation when x2=0**



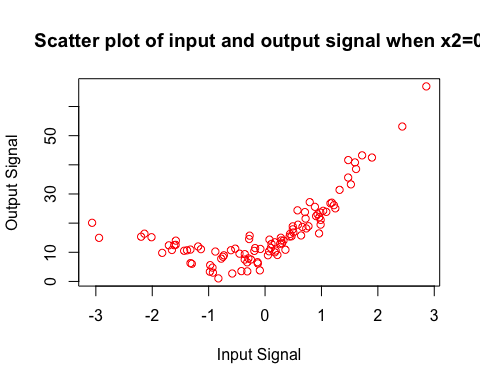
**Correlation when x2=1**



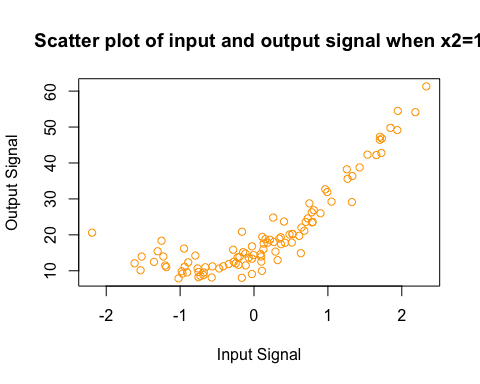
**The Scatter Plot**

A scatter plot is used to show the relationship between two variables. In the data, the relationship between the input and output signal is taken and used. It will look into their dependencies. The **plot()** function is applied.

**Scatter Plot when x2=0**



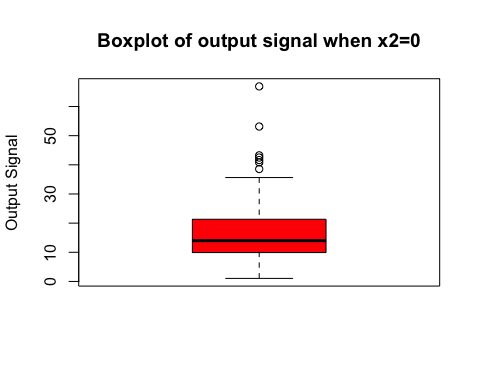
**Scatter Plot when x2=1**



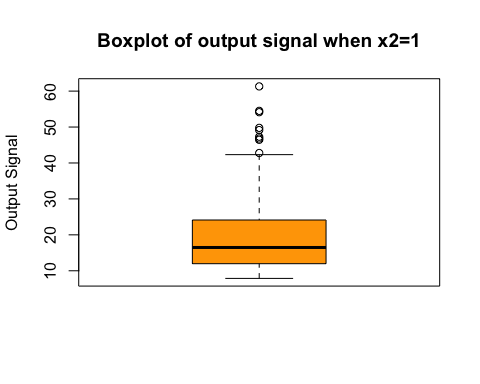
**Boxplot Representation**

Boxplot is a data visualization tool that is used to check for the distribution of the data. In this case, the boxplot is used to examine the effect of sound categories using the output brain signals.

**Box Plot Box plot when x2=0**



**Box plot when x2=1**



3. Task 2:Regression – Modelling The Relationship Between MEG

Signals

It is needed to come up with a mathematical model that can explain the relationship between the input signals and the output signals and how this relationship changes based on the content of the input audio signals. assuming such relation can be described using a regression model.

**Task 2.1 Estimating Model Parameters Using Least Squares**

Each model is analysed separately to aid in the relationship testing. Since there are different models, each value of each model is found separately. It is necessary to convert the data meg into a data frame, here the function **as.data.frame()** is used to make it a data frame and then stored to a variable **regrndata.** For the starting point, it is started with the evaluation of the models one by one. Then each model is made into a data frame. Since least squares are calculations concerning matrix, each data frame should be converted to a matrix form using the **as.matrix().** After successfully transforming the data frame into a matrix form, R programming is used to do the calculations. **Solve()** is used to do inverse calculations concerning matrices.

**t()** function used for the purpose of transposing a matrix.

**%\*%** does the matrix multiplications.

**head()** is a function that returns the first part of the matrix (6 rows) that contains many rows.

**Task 2.2 Residual Sum of Squared Error**

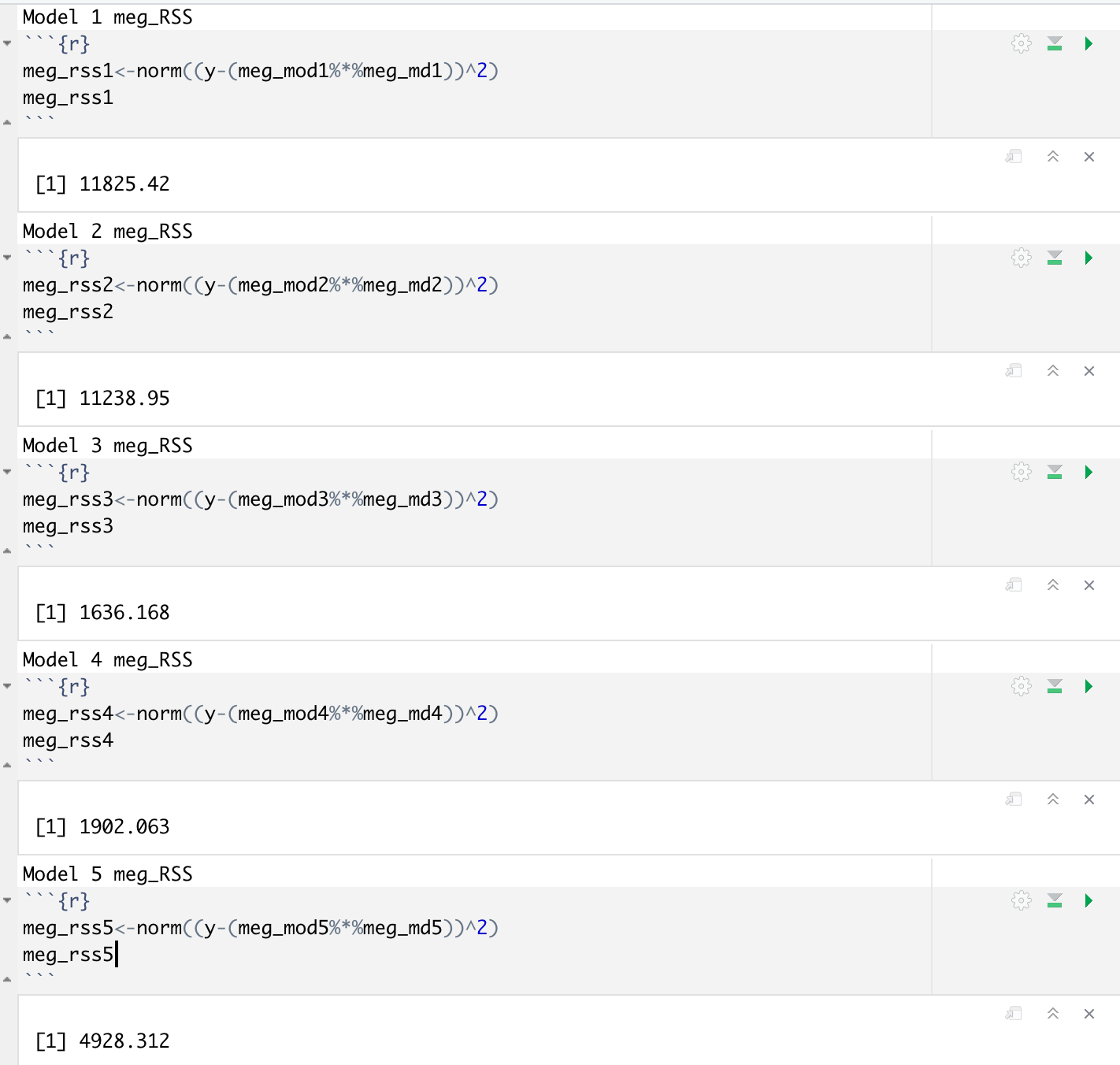
Considering the given model the RSS(Residual Sum of Squared Error) of every model can be figured out using the given formula. For RSS the sum of the error is obtained by taking all the deviations of output stored in a variable y. The variable y is converted into the matrix format.

The RSS for Model 1 is calculated using the R code.

**meg\_rss1<-norm((y-(meg\_mod1%\*%meg\_md1))^2)**

**meg\_rss1**

A similar method is used to find the RSS for the remaining models.



**Task 2.3 The log-likelihood Function**

The log-likelihood function is a function declared in r to take at least two or more arguments. Here every log function is performed by the provided formula on all the models i.e.

The log-likelihood for Model 1 is calculated using the R code.

**meg\_logfxn1<- -(nrow(y)/2\*log(2\*pi))-(nrow(y)/2\*(log(meg\_rss1/(nrow(y)-1))))-meg\_rss1\*(1/(2\*(meg\_rss1/(nrow(y)-1))))**

**meg\_logfxn1**

A similar method is used to find the log-likelihood for the remaining models.

****

**Task 2.4 The AIC & BIC**

In each model the number k (the number of parameters estimated) is assigned differently i.e., **model 1(k=4),model 2(k=3) , model 3(k=5), model 4(k=6) and model 5(k=5).** The Value of K can be calculated by using the code.

**length(meg\_md1)**

This will provide the no: of parameters passed to the First model.

The Akaike information criterion(AIC) and the Bayesian information criterion(BIC) provide a measure of model performance that accounts for model complexity. They also combine a reflection of how well the model fits the data. The lower the AIC & BIC Value the better the model.

The AIC & BIC for Model 1 is calculated using the R code.

**k1=4**

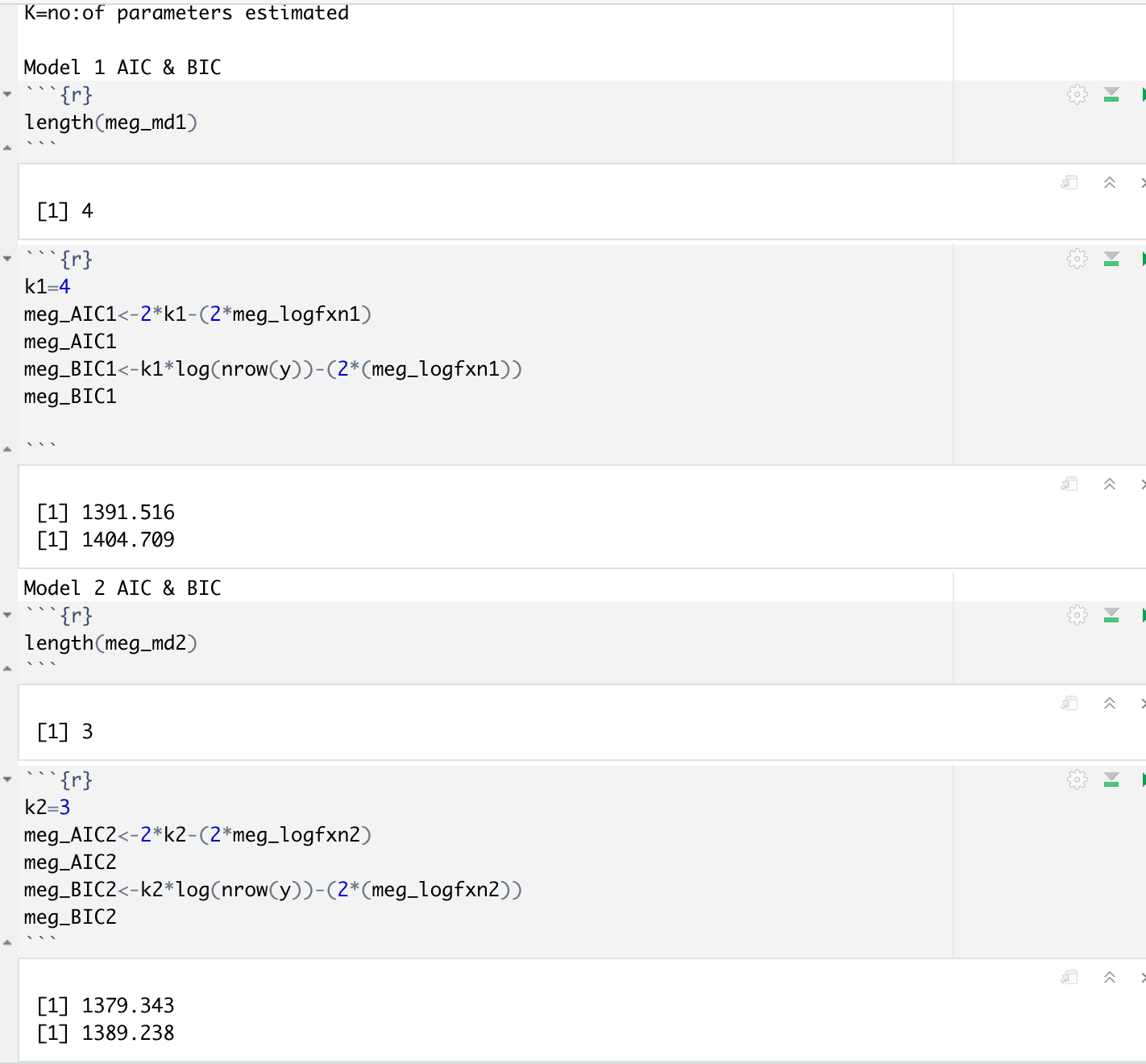
**meg\_AIC1<-2\*k1-(2\*meg\_logfxn1)**

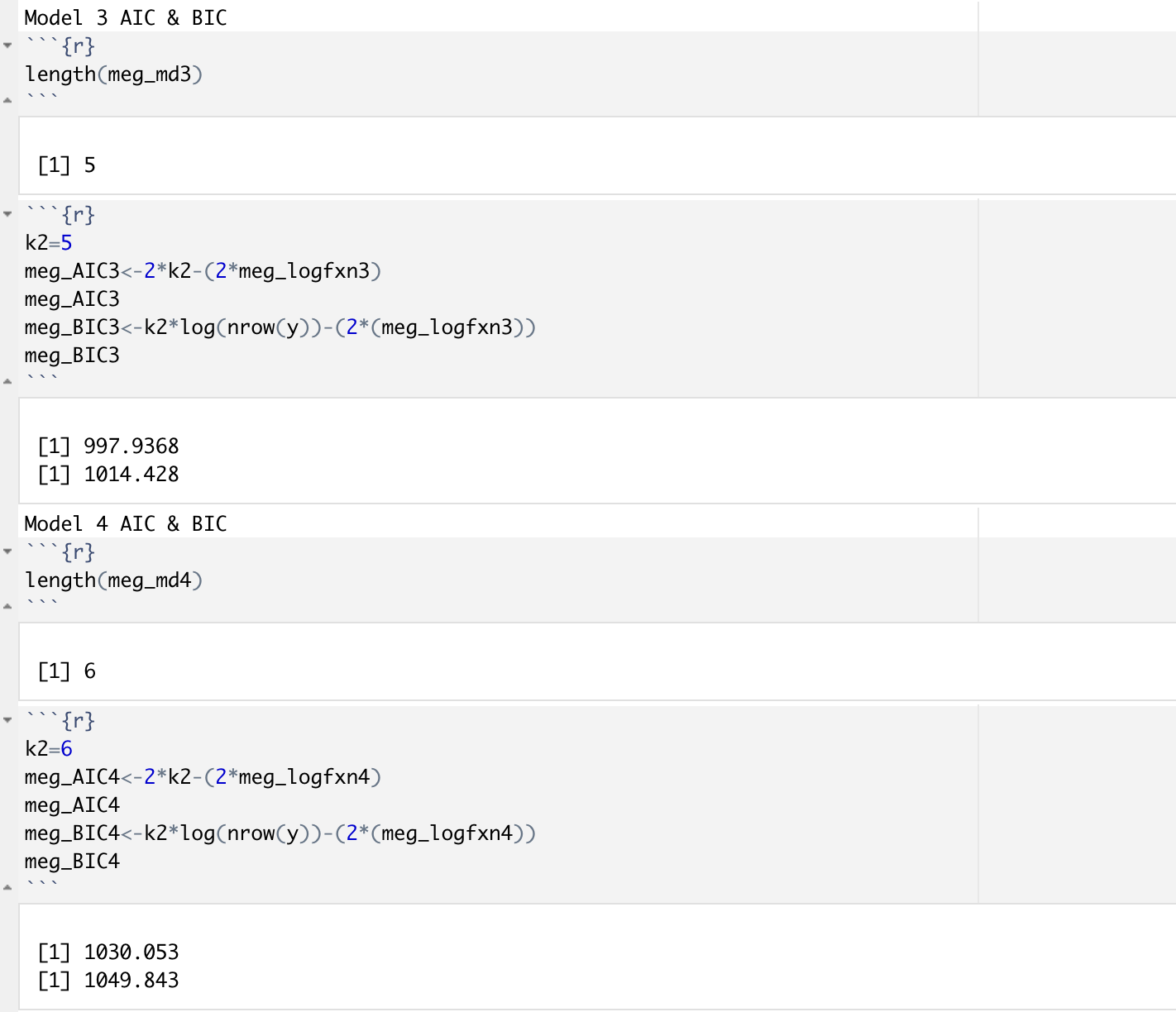
**meg\_AIC1**

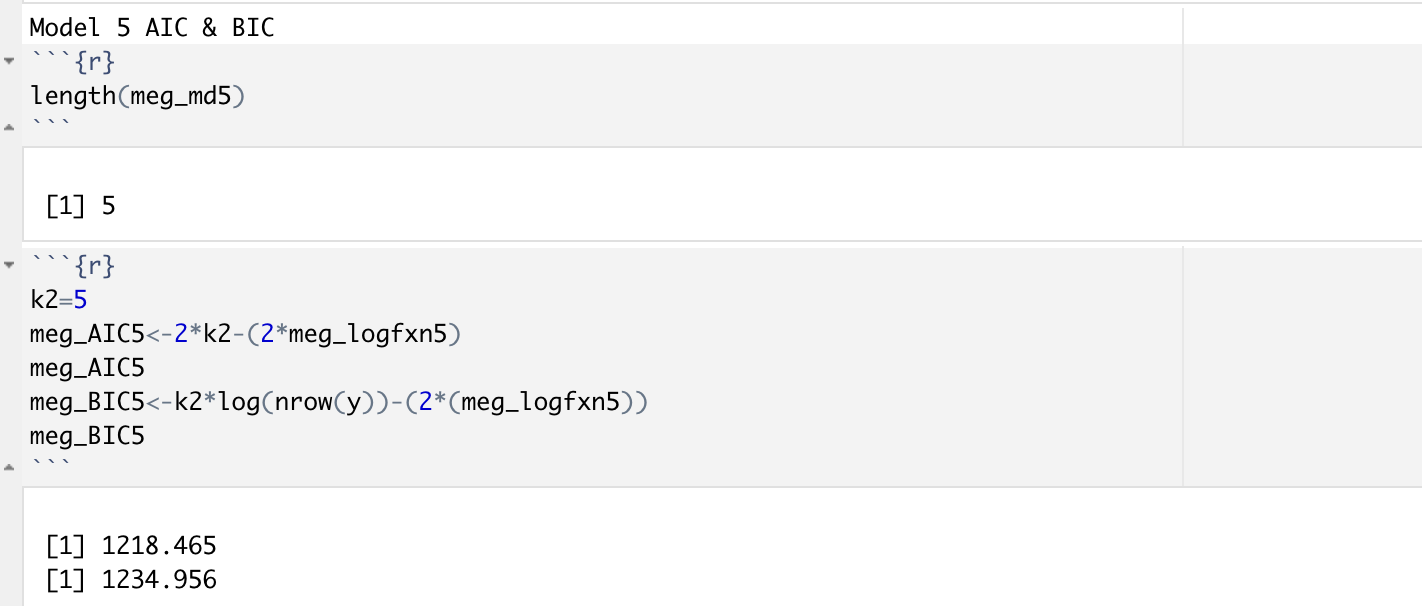
**meg\_BIC1<-k1\*log(nrow(y))-(2\*(meg\_logfxn1))**

**meg\_BIC1**

A similar method is used to find the AIC & BIC for the remaining models.



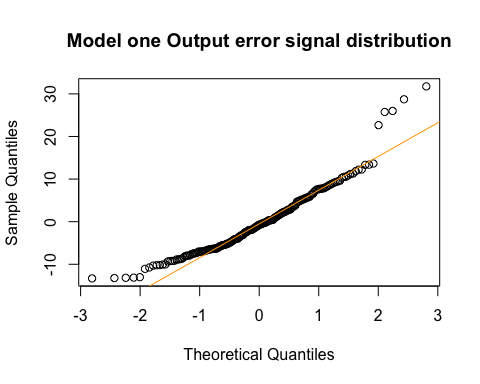
****

****

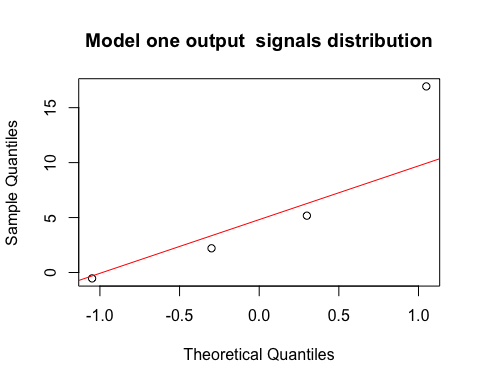
**Task 2.5 Distribution of Model Prediction Errors**

To check for the model prediction errors it is needed to plot the error distribution by use of the qqnorm() & qqline() of each model.

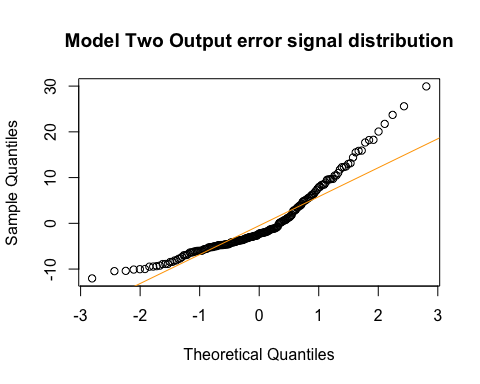
**Model one Output error signal distribution**



**Model one output signals distribution**



**Model Two Output error signal distribution**

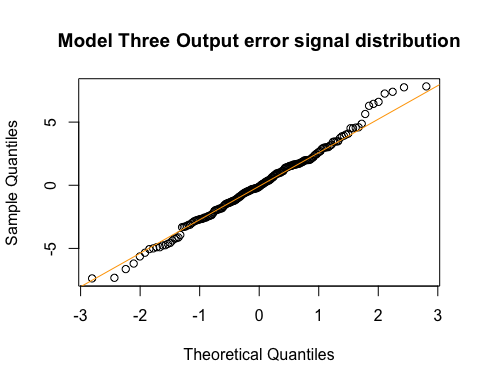


**Model Two output signals distribution**

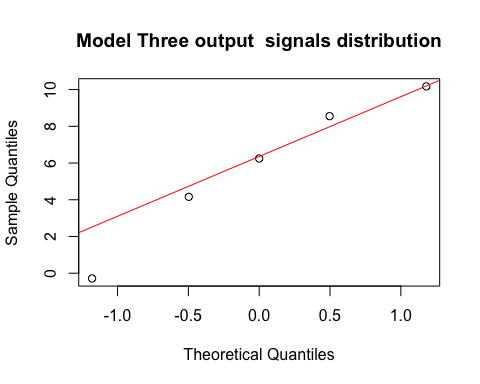
Chart, line chart

Description automatically generated

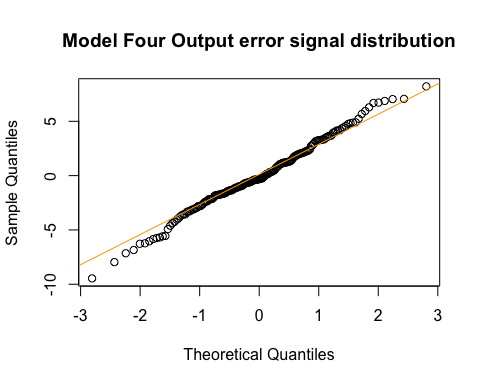
**Model Three Output error signal distribution**



**Model Three output signals distribution**



**Model Four Output error signal distribution**

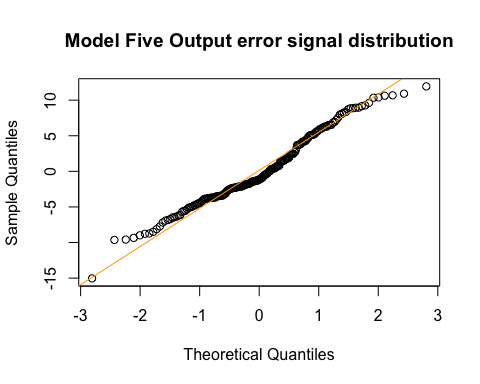


**Model Four output signals distribution**

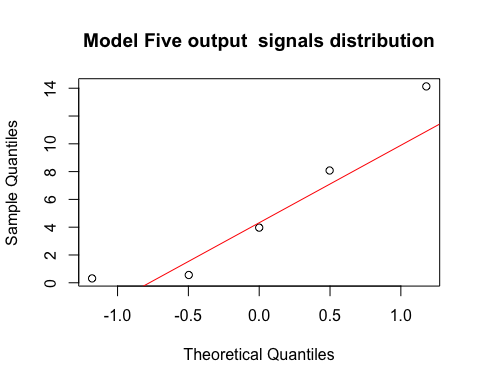
Chart, scatter chart

Description automatically generated

**Model Five Output error signal distribution**



**Model Five output signals distribution**



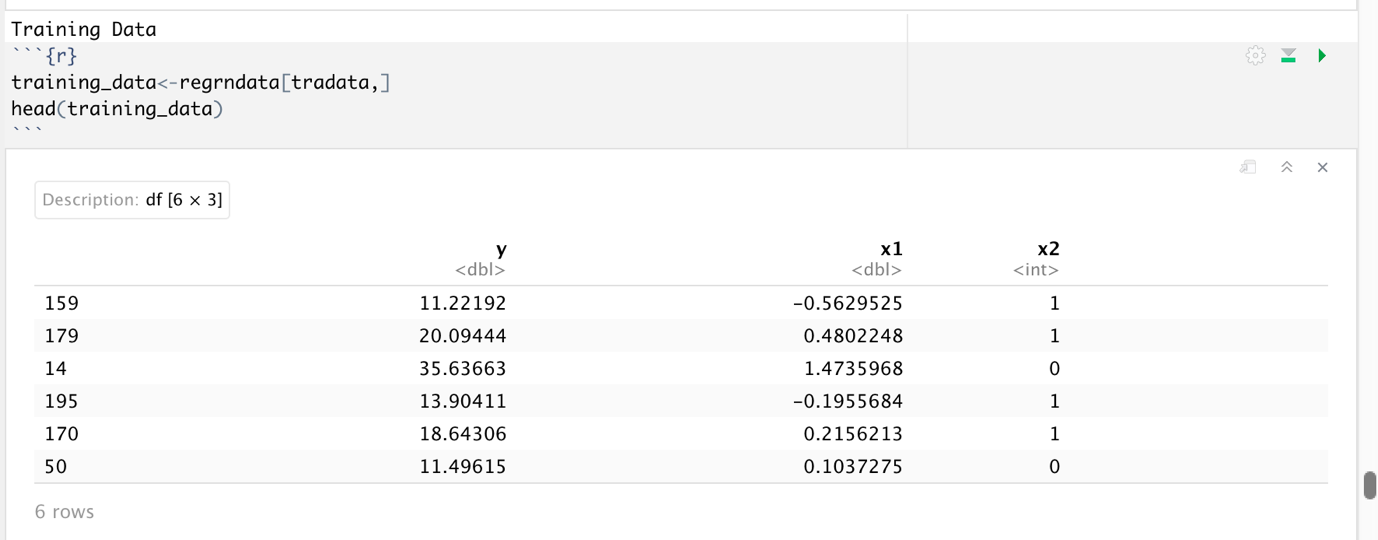
**Task 2.6 The Best Model**

For a model to be ranked the best it should have the least AIC and BIC values. Based on the calculations figured Model 3 is the best Regression model as it has the smallest values of AIC (997.368) & BIC (1014.428).

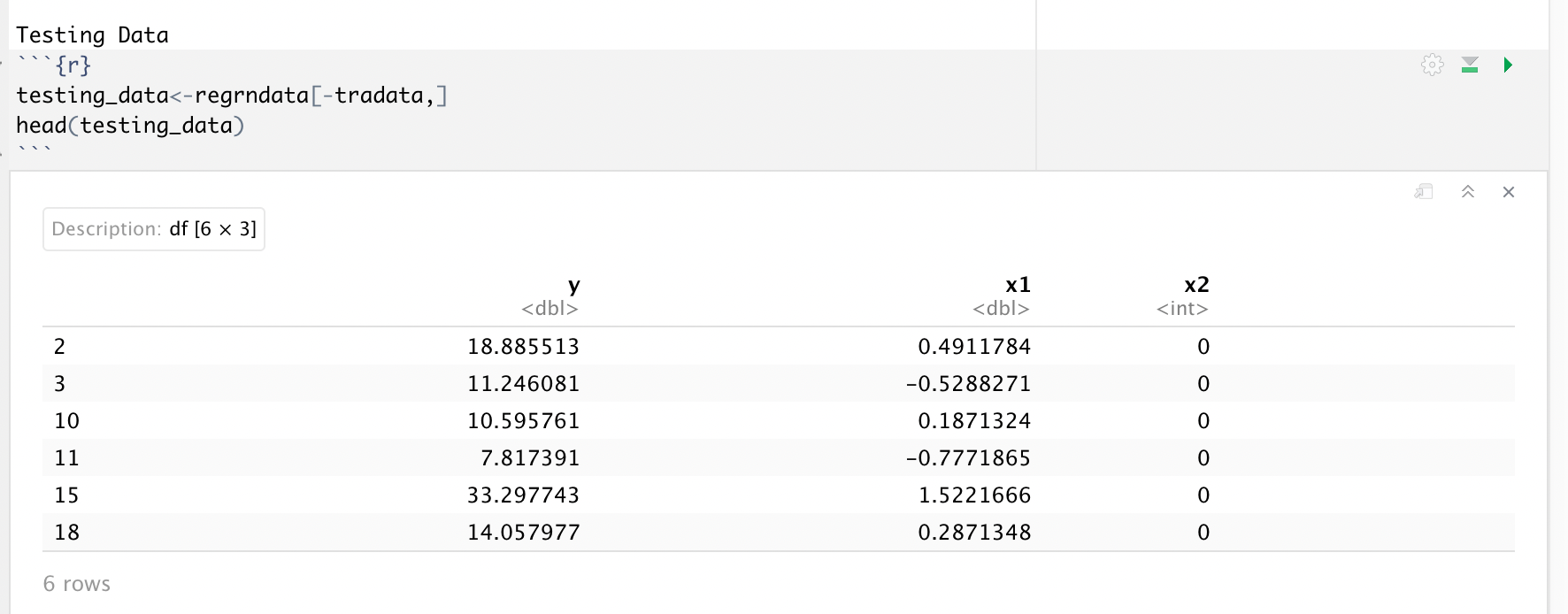
**Task 2.7 Splitting the Data**

It is necessary to collect the random numbers, then it is needed to get random variables that will be the same for all the chosen random variables. For that to be achieved it is needed to use a function of generating random variables set.seed(). A sample of 140 rows is selected. The input and output are split into two parts one part is used for training and the other part is used for testing.

**Training Data**

****

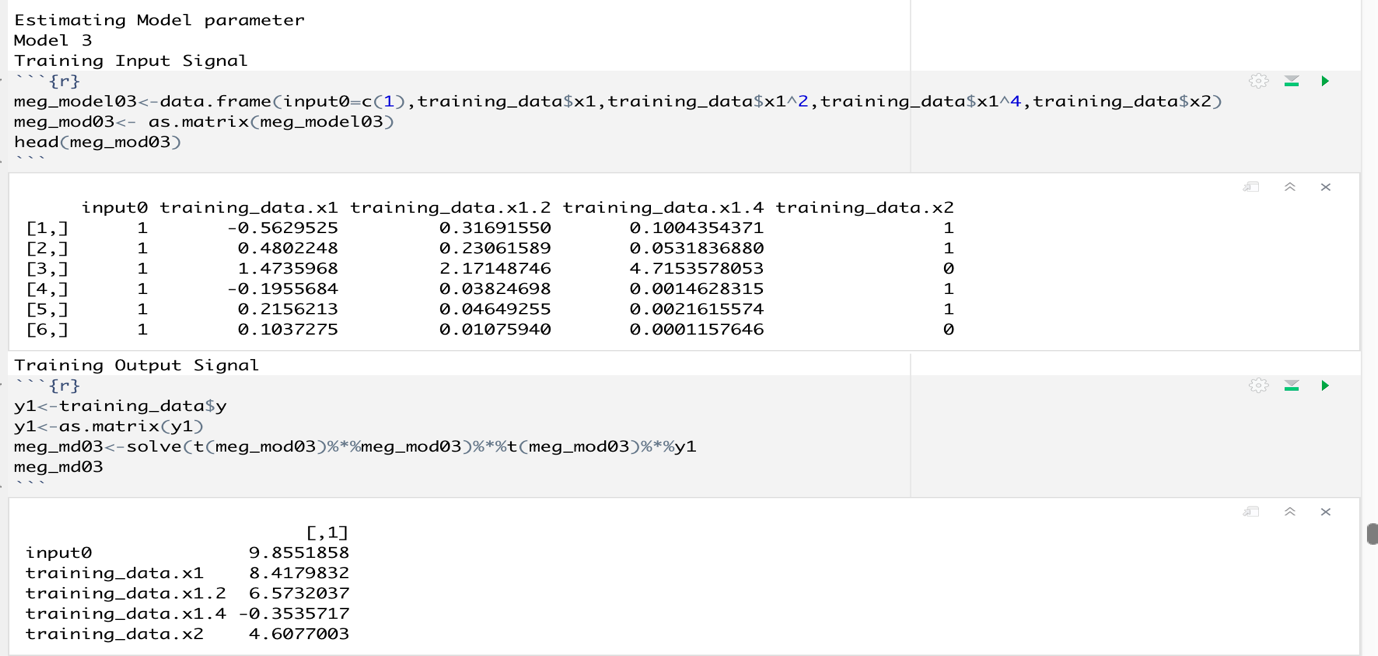
**Testing Data**



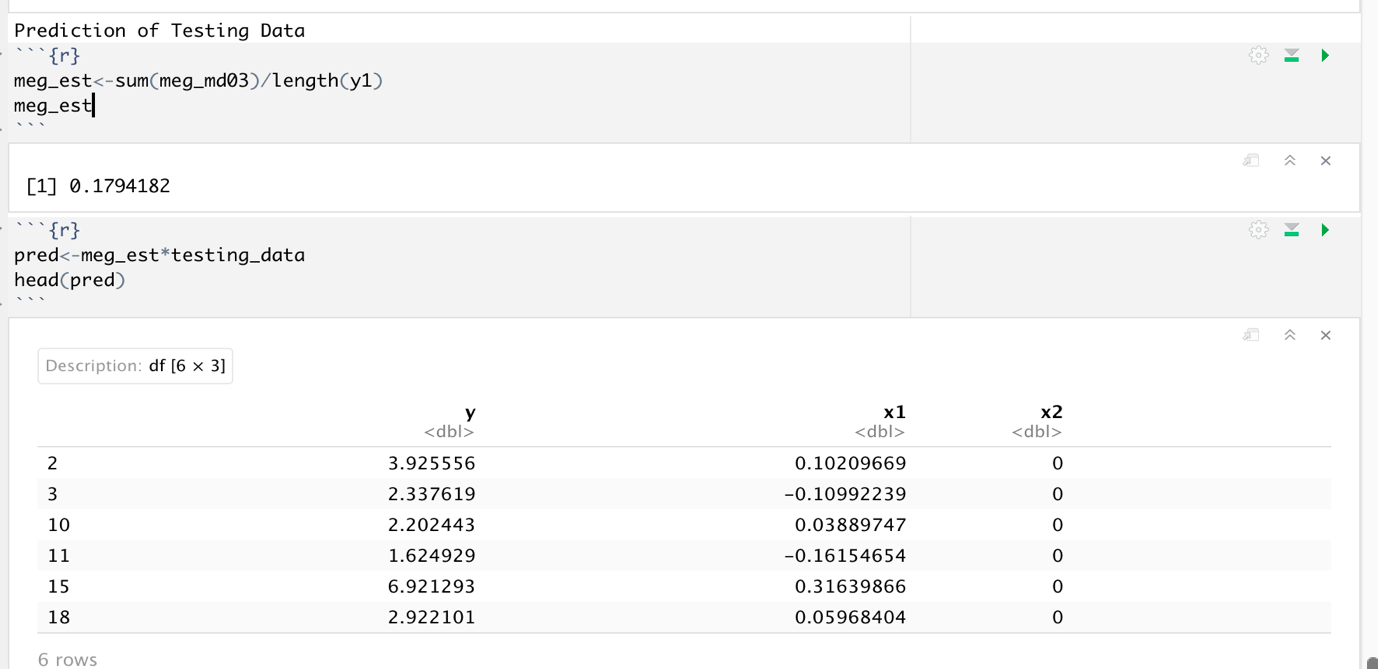
**Estimating Model Parameters**

According to the previous evaluation of which is the best model, the result was that Model 3 is the best fit due to its low AIC & BIC values compared to the rest of the models. The parameters are estimated using the least squares. The training and testing methods are implemented for Model 3. It’s required to create a data frame of the training input data. After getting the result the data frame is converted into a matrix format to enable an easy arithmetic calculation.

**Training Input and Output Signal**

****

The testing data prediction is carried out by the formula sum of the training output signal divided by the length of the output, the obtained value is multiplied by the testing variable.



**Confidence Interval**

Confidence intervals measure the degree of uncertainties or certainty. To get the confidence intervals in r some of the functions are needed and are so useful.

* Sqrt() to get the square root.
* Var() to id in getting the variance.
* Mean() to get the mean.
* Paste() gives the same functions as the concatenate functions.
* $y to extract specific data in r in the case data y is needed.

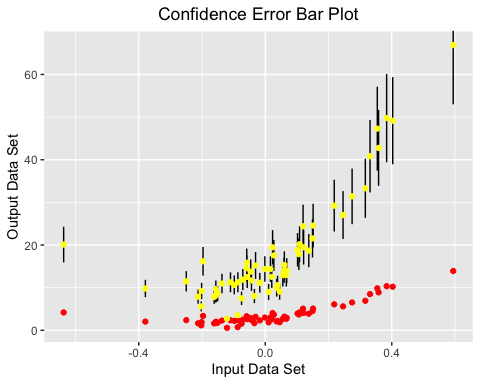
Testing will be done to find out about the two data sets the input and the output signal.

**95% Confidence Interval of Input, the Output signal**

****

Error is calculated by subtracting the y values of testing data and prediction. Confidence limits are added to predict the data. With the help of ggplot Confidence Intervals and error bars are plotted.

**Plotting confidence intervals and error bars**



**4. Task 3: Approximation Bayesian Computation**

To get the absolute values the sum of columns needs to be obtained and to achieve that a function **colSums()** is introduced and used.

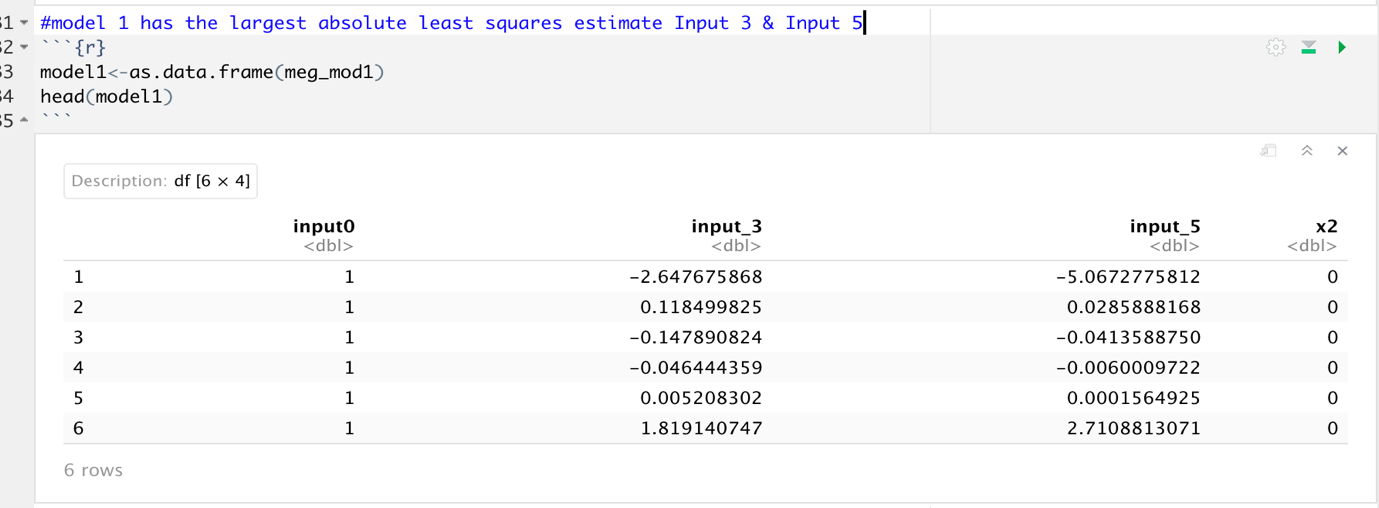
**2 parameters with the largest absolute least squares estimation**

**Graphical user interface, text, application

Description automatically generated**

**Absolute Values**

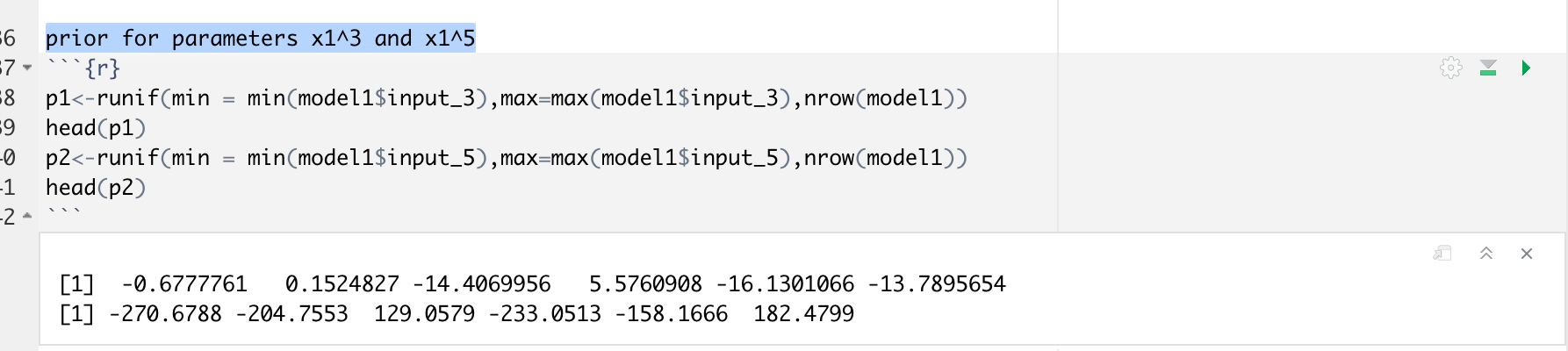
The largest absolute least square estimation is carried out for all the five models. Based on the results obtained Model 1 has the largest absolute value. It possesses the largest input values Input 3 and Input 5

****

**Range of Prior Distribution**

To get the range of the prior distribution, will have to call the model 1 data and get to a data frame. Using the **runif()** to use it as a uniform distribution prior. **runif()** will aid the generation of random deviates. It also helps in generating n numbers of randomness defined in the **runif()** argument.

**Prior for parameters input 3 and input 5**

****

A sample value of 50 is stored in the variable n

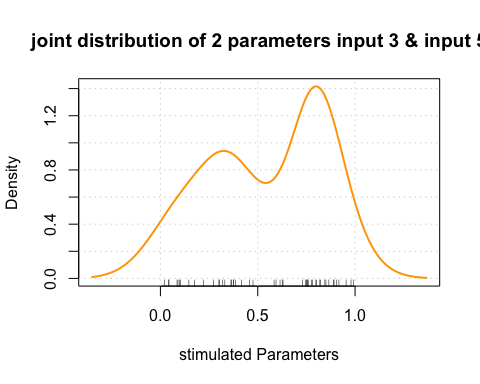
**Rejection of ABC**

For performing the below tasks importing the EasyABC installed package in r by the library function.

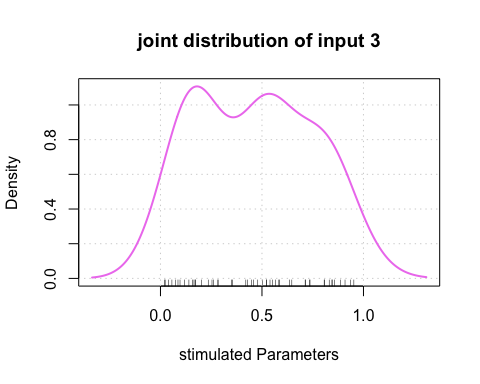
**Plotting of Marginal Distribution**

A plot of a marginal distribution will show the distribution of data along only one dimension. List() used to create a list of elements of different types. By using the Density plot we have plotted the joint distribution.

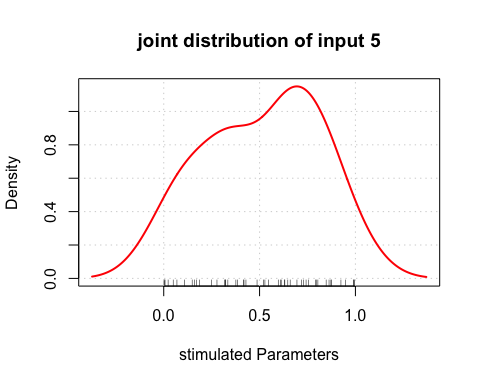
**Plot Joint Distribution of parameters input 3 & Input 5**



**plot the marginal distribution of Input 3 parameter**



**plot the marginal distribution of Input 5 parameter**



**From the plot, it is clearly understandable that the density curves seem to be normally distributed for both the marginal and joint distribution of the 2 posterior parameters.**

**Summary of the Task**

* Time series analysis of the non-linear regression model is plotted using the ts() function. The graph is plotted for the input(x2=0 & x2=1) and also for the output.
* The distribution for each input and output signal is plotted using the Histogram method. The hist() function is used for the same.
* The correlation method is used to find the relation between dependent and independent variables. The input is the independent variable and the output is the dependent variable. The pie method is used for representing the correlation.
* The scatter plot is used to plot the dependencies between the input and output variables.
* The Boxplot is used to find the skewness of the output signal corresponding to the time interval. Boxplot() function is used to perform the task.
* A suitable model is figured out from the given 5 sets of models.
* Using the least-squares method theta parameter is calculated for all 5 models.
* Based on the model parameters estimated, the model residual sum of squared errors is calculated for every candidate's model using the provided formula.
* The log-likelihood function is calculated for every candidate model using the provided formula to find out how well a model fits the data.
* The AIC and BIC values are calculated for all 5 models to figure out the best regression model
* The errors are calculated for the given models and the estimated errors are plotted using qqplot to evaluate if the distribution is close to the normal one.
* According to the findings Model 3 is the best regression method due to its low AIC & BIC values.
* The input and output signals are split into two one for training and another one for testing. From the selected best model parameters, prediction and 95% confidence intervals are figured.
* ggplot is used for plotting the confidence intervals and error bars.
* Based on the calculation Model 1 has the largest Absolute value.
* ABC rejection method is used to compute the distribution of the selected regression model.
* The density curves seem to be normally distributed for both the marginal and joint distribution of the 2 posterior parameters.

**5. References**

* Data visualization with R and ggplot2 retrieved from geeks for geeks - [**https://www.geeksforgeeks.org/data-visualization-with-r-and-ggplot2/**](https://www.geeksforgeeks.org/data-visualization-with-r-and-ggplot2/)
* Linear regression from scratch in r retrieved from Data science plus [**https://datascienceplus.com/linear-regression-from-scratch-in-r/**](https://datascienceplus.com/linear-regression-from-scratch-in-r/)
* A grammar for Data manipulation retrieved from Cran.R project [**https://cran.r-project.org/web/packages/dplyr/dplyr.pdf**](https://cran.r-project.org/web/packages/dplyr/dplyr.pdf)
* For R Basics referred from R tutorial W3 Schools

[**https://www.w3schools.com/r/**](https://www.w3schools.com/r/)

**6. Appendix**

R code Drive Link - <https://drive.google.com/drive/u/1/folders/1qSqGjZvmTseMYbrHhk1-TYwG8O0dNsGS>

---

importing common packages

```{r}

library(ggplot2)

library(tidyverse)

library(dplyr)

```

importing data **and** time series packages

```{r}

library(lubridate)

library(tseries)

library(car)

```

Setting working environment directory

```{r}

setwd("/Users/arjun/Downloads/Arjun\_Course")

```

Task 1: Preliminary data analysis

Importing input signal, output signal, **and** time data

```{r}

input<-read.csv("X.csv")

output<-read.csv("y.csv")

times<-read.csv("time.csv")

```

combining output signal **and** input signal

```{r}

meg<-cbind(output,input)

```

Converting data to time series **with** interval of 10 sec

```{r}

meg1<-ts(meg,end = 20,frequency = 10,start = 0.1)

head(meg1)

```

Converting time series data to data frame structure Selecting column

y,x2,x1

*##Time Series Time series plotting for x2=0 (neutral audio)*

```{r}

tsmeg<-**as**.data.frame(meg1)%>%filter(x2=='0')%>%ts(start = 0.1,end=10,frequency = 10)%>%

plot(type='b',col='blue',main='time series plot of input and output signal x2=0')

```

Time series plotting **for** x2=1 (Emotional audio)

```{r}

tsmeg1<-**as**.data.frame(meg1)%>%filter(x2=='1')%>%ts(start = 10,end = 20,frequency = 10)%>%

plot(type='b',col='Red',main='time series plot of input and output signal x2=1')

```

Distribution of input signal **and** output signal

Filtering the values **for** x2==0,neutral audio

```{r}

megdist<-**as**.data.frame(meg1)%>%

filter(x2=='0')%>%

select(x1,y) *#selecting column x1,y*

```

Filtering the values **for** x2==1,emotional audio

```{r}

megdist1<-**as**.data.frame(meg1)%>%

filter(x2=='1')%>%

select(x1,y) *#selecting column x1,y*

```

*##Histogram*

Histogram of output signal

```{r}

hist(meg$y,xlab = 'output',main ='histogram of output\_signal',col = 'purple' )

```

Histogram of Output signal when x2=0

```{r}

hist(megdist$y,xlab = 'output',main ='histogram of output\_signal(x2=0)',col = 'Yellow' )

```

Histogram of Input signal when x2=0

```{r}

hist(megdist$x1,xlab = 'input',main='histogram of input\_signal(x2=0)',col ='Red')

```

Histogram of Output signal when x2=1

```{r}

hist(megdist1$y,xlab = 'output',main ='histogram of output\_signal(x2=1)',col = 'orange' )

```

Histogram of Input signal when x2=1

```{r}

hist(megdist1$x1,xlab = 'input',main='histogram of input\_signal(x2=1)',col ='blue')

```

Correlation when x2=0

```{r}

megcor<-cor(megdist)

corrplot::corrplot.mixed(megcor,upper = "pie",lower = "number")

```

Correlation when x2=1

```{r}

megcor1<-cor(megdist1)

corrplot::corrplot.mixed(megcor1,upper = "pie",lower = "number")

```

*#Scatter Plot*

Scatter Plot when x2=0

```{r}

megdist%>%plot(type='p',main="Scatter plot of input and output signal when x2=0",col='Red',xlab='Input Signal',ylab='Output Signal')

```

Scatter Plot when x2=1

```{r}

megdist1%>%plot(type='p',main="Scatter plot of input and output signal when x2=1",col='Orange',xlab='Input Signal',ylab='Output Signal')

```

*##Box Plot Box plot when x2=0*

```{r}

boxplot(megdist$y,main='Boxplot of output signal when x2=0',col='Red',ylab='Output Signal')

```

Box plot when x2=1

```{r}

boxplot(megdist1$y,main='Boxplot of output signal when x2=1',col='Orange',ylab='Output Signal')

```

*##Task 2 Regression*

Task 2.1

```{r}

regrndata<-**as**.data.frame(meg)

```

Input values of x1

```{r}

input\_1=regrndata$x1

input\_2=regrndata$x1^2

input\_3=regrndata$x1^3

input\_4=regrndata$x1^4

input\_5=regrndata$x1^5

```

Input values of x2

```{r}

x2=regrndata$x2

```

Output value

```{r}

y<-output

y<-**as**.matrix(y)

```

Model 1

```{r}

meg\_model1<-data.frame(input0=c(1),input\_3,input\_5,x2) *## Creating Data frame for Model 1*

meg\_mod1<-**as**.matrix(meg\_model1)

head(meg\_mod1)

meg\_md1<-solve(t(meg\_mod1)%\*%meg\_mod1)%\*%t(meg\_mod1)%\*%y *##Estimating theta Parameter using least squares method*

meg\_md1

```

Model 2

```{r}

meg\_model2<-data.frame(input0=c(1),input\_1,x2) *##Creating Data frame for Model 2*

meg\_mod2<-**as**.matrix(meg\_model2)

head(meg\_mod2)

meg\_md2<-solve(t(meg\_mod2)%\*%meg\_mod2)%\*%t(meg\_mod2)%\*%y *##Estimating theta Parameter using least squares method*

meg\_md2

```

Model 3

```{r}

meg\_model3<-data.frame(input0=c(1),input\_1,input\_2,input\_4,x2) *##Creating Data frame for Model 3*

meg\_mod3<-**as**.matrix(meg\_model3)

head(meg\_mod3)

meg\_md3<-solve(t(meg\_mod3)%\*%meg\_mod3)%\*%t(meg\_mod3)%\*%y *##Estimating theta Parameter using least squares method*

meg\_md3

```

Model 4

```{r}

meg\_model4<-data.frame(input0=c(1),input\_1,input\_2,input\_3,input\_5,x2) *##Creating Data frame for Model 4*

meg\_mod4<-**as**.matrix(meg\_model4)

head(meg\_mod4)

meg\_md4<-solve(t(meg\_mod4)%\*%meg\_mod4)%\*%t(meg\_mod4)%\*%y *##Estimating theta Parameter using least squares method*

meg\_md4

```

Model 5

```{r}

meg\_model5<-data.frame(input0=c(1),input\_1,input\_3,input\_4,x2) *##Creating Data frame for Model 5*

meg\_mod5<-**as**.matrix(meg\_model5)

head(meg\_mod5)

meg\_md5<-solve(t(meg\_mod5)%\*%meg\_mod5)%\*%t(meg\_mod5)%\*%y *##Estimating theta Parameter using least squares method*

meg\_md5

```

*##Task 2.2 RSS*

Model 1 meg\_RSS

```{r}

meg\_rss1<-norm((y-(meg\_mod1%\*%meg\_md1))^2)

meg\_rss1

```

Model 2 meg\_RSS

```{r}

meg\_rss2<-norm((y-(meg\_mod2%\*%meg\_md2))^2)

meg\_rss2

```

Model 3 meg\_RSS

```{r}

meg\_rss3<-norm((y-(meg\_mod3%\*%meg\_md3))^2)

meg\_rss3

```

Model 4 meg\_RSS

```{r}

meg\_rss4<-norm((y-(meg\_mod4%\*%meg\_md4))^2)

meg\_rss4

```

Model 5 meg\_RSS

```{r}

meg\_rss5<-norm((y-(meg\_mod5%\*%meg\_md5))^2)

meg\_rss5

```

*##Task2.3 Likelihood function*

log likelihood 1

```{r}

meg\_logfxn1<- -(nrow(y)/2\*log(2\*pi))-(nrow(y)/2\*(log(meg\_rss1/(nrow(y)-1))))-meg\_rss1\*(1/(2\*(meg\_rss1/(nrow(y)-1))))

meg\_logfxn1

```

log likelihood 2

```{r}

meg\_logfxn2<- -(nrow(y)/2\*log(2\*pi))-(nrow(y)/2\*(log(meg\_rss2/(nrow(y)-1))))-meg\_rss2\*(1/(2\*(meg\_rss2/(nrow(y)-1))))

meg\_logfxn2

```

log likelihood 3

```{r}

meg\_logfxn3<- -(nrow(y)/2\*log(2\*pi))-(nrow(y)/2\*(log(meg\_rss3/(nrow(y)-1))))-meg\_rss3\*(1/(2\*(meg\_rss3/(nrow(y)-1))))

meg\_logfxn3

```

log likelihood 4

```{r}

meg\_logfxn4<- -(nrow(y)/2\*log(2\*pi))-(nrow(y)/2\*(log(meg\_rss4/(nrow(y)-1))))-meg\_rss4\*(1/(2\*(meg\_rss4/(nrow(y)-1))))

meg\_logfxn4

```

log likelihood 5

```{r}

meg\_logfxn5<- -(nrow(y)/2\*log(2\*pi))-(nrow(y)/2\*(log(meg\_rss5/(nrow(y)-1))))-meg\_rss5\*(1/(2\*(meg\_rss5/(nrow(y)-1))))

meg\_logfxn5

```

*#TASK 2.4 AIC AND BIC*

K=no:of parameters estimated

Model 1 AIC & BIC

```{r}

length(meg\_md1)

```

```{r}

k1=4

meg\_AIC1<-2\*k1-(2\*meg\_logfxn1)

meg\_AIC1

meg\_BIC1<-k1\*log(nrow(y))-(2\*(meg\_logfxn1))

meg\_BIC1

```

Model 2 AIC & BIC

```{r}

length(meg\_md2)

```

```{r}

k2=3

meg\_AIC2<-2\*k2-(2\*meg\_logfxn2)

meg\_AIC2

meg\_BIC2<-k2\*log(nrow(y))-(2\*(meg\_logfxn2))

meg\_BIC2

```

Model 3 AIC & BIC

```{r}

length(meg\_md3)

```

```{r}

k2=5

meg\_AIC3<-2\*k2-(2\*meg\_logfxn3)

meg\_AIC3

meg\_BIC3<-k2\*log(nrow(y))-(2\*(meg\_logfxn3))

meg\_BIC3

```

Model 4 AIC & BIC

```{r}

length(meg\_md4)

```

```{r}

k2=6

meg\_AIC4<-2\*k2-(2\*meg\_logfxn4)

meg\_AIC4

meg\_BIC4<-k2\*log(nrow(y))-(2\*(meg\_logfxn4))

meg\_BIC4

```

Model 5 AIC & BIC

```{r}

length(meg\_md5)

```

```{r}

k2=5

meg\_AIC5<-2\*k2-(2\*meg\_logfxn5)

meg\_AIC5

meg\_BIC5<-k2\*log(nrow(y))-(2\*(meg\_logfxn5))

meg\_BIC5

```

*##TASK 2.5 Check the distribution of each model prediction errors*

Plot the errors **and** check **if** they are normally distributed

Model 1 residual distribution

```{r}

meg\_err1=(y-(meg\_mod1%\*%meg\_md1))*#error*

head(meg\_err1)

```

Model one Output error signal distribution

```{r}

qqnorm(meg\_err1,main ="Model one Output error signal distribution")*#error plot*

qqline(meg\_err1,col="Orange")

```

Model one output signals distribution

```{r}

qqnorm(meg\_md1,main = "Model one output signals distribution")

qqline(meg\_md1,col="Red")

```

Model 2 residual distribution

```{r}

meg\_err2=(y-(meg\_mod2%\*%meg\_md2))*#error*

head(meg\_err2)

```

Model Two Output error signal distribution

```{r}

qqnorm(meg\_err2,main ="Model Two Output error signal distribution")*#error plot*

qqline(meg\_err2,col="Orange")

```

Model Two output signals distribution

```{r}

qqnorm(meg\_md2,main = "Model Two output signals distribution")

qqline(meg\_md2,col="Red")

```

Model 3 residual distribution

```{r}

meg\_err3=(y-(meg\_mod3%\*%meg\_md3))*#error*

head(meg\_err3)

```

Model Three Output error signal distribution

```{r}

qqnorm(meg\_err3,main ="Model Three Output error signal distribution")*#error plot*

qqline(meg\_err3,col="Orange")

```

Model Three output signals distribution

```{r}

qqnorm(meg\_md3,main = "Model Three output signals distribution")

qqline(meg\_md3,col="Red")

```

Model 4 residual distribution

```{r}

meg\_err4=(y-(meg\_mod4%\*%meg\_md4))*#error*

head(meg\_err4)

```

Model Four Output error signal distribution

```{r}

qqnorm(meg\_err4,main ="Model Four Output error signal distribution")*#error plot*

qqline(meg\_err4,col="Orange")

```

Model Four output signals distribution

```{r}

qqnorm(meg\_md4,main = "Model Four output signals distribution")

qqline(meg\_md4,col="Red")

```

Model 5 residual distribution

```{r}

meg\_err5=(y-(meg\_mod5%\*%meg\_md5))*#error*

head(meg\_err5)

```

Model Five Output error signal distribution

```{r}

qqnorm(meg\_err5,main ="Model Five Output error signal distribution")*#error plot*

qqline(meg\_err5,col="Orange")

```

Model Five output signals distribution

```{r}

qqnorm(meg\_md5,main = "Model Five output signals distribution")

qqline(meg\_md5,col="Red")

```

*#TASK 2.6 Select the best regression model according to AIC and BIC*

The best regression model **is** model 3, reason: it has the smallest values of AIC(997.9368)

**and** BIC(1014.428) comparing of all the 5 regression models.

*#Task 2.7*

Splitting the data **for** training **and** testing

```{r}

set.seed(123)

tradata<-sample(1:200,140) *#Taking sample of 140 rows*

head(tradata)

```

Training Data

```{r}

training\_data<-regrndata[tradata,]

head(training\_data)

```

Testing Data

```{r}

testing\_data<-regrndata[-tradata,]

head(testing\_data)

```

Estimating Model parameter

Model 3

Training Input Signal

```{r}

meg\_model03<-data.frame(input0=c(1),training\_data$x1,training\_data$x1^2,training\_data$x1^4,training\_data$x2)

meg\_mod03<- **as**.matrix(meg\_model03)

head(meg\_mod03)

```

Training Output Signal

```{r}

y1<-training\_data$y

y1<-**as**.matrix(y1)

meg\_md03<-solve(t(meg\_mod03)%\*%meg\_mod03)%\*%t(meg\_mod03)%\*%y1

meg\_md03

```

Prediction of Testing Data

```{r}

meg\_est<-sum(meg\_md03)/length(y1)

meg\_est

```

```{r}

pred<-meg\_est\*testing\_data

head(pred)

```

95% Confidence Interval of Output signal

```{r}

sd<-sqrt(var(pred$y))

sd

means<-mean(pred$y)

means

paste('Confidence Interval',means-1.96\*sd/sqrt(nrow(testing\_data)),means+1.96\*sd/sqrt(nrow(testing\_data)))

```

95% Confidence Interval of Input signal

```{r}

sd1<-sqrt(var(pred$x1))

sd1

means1<-mean(pred$x1)

means1

paste('Confidence Interval',means1-1.96\*sd1/sqrt(nrow(testing\_data)),means1+1.96\*sd1/sqrt(nrow(testing\_data)))

```

Error

```{r}

meg\_error<-testing\_data$y-pred$y

head(meg\_error)

meg\_error1<-testing\_data$y+pred$y

head(meg\_error1)

```

Adding Confidence limits to predict the data

```{r}

pred\_Lower<-meg\_error

pred\_Upper<-meg\_error1

pred\_testy<-testing\_data$y

```

Plotting Confidence intervals **and** error bars

```{r}

pred %>% ggplot(aes(x1,y))+geom\_errorbar(ymin=pred\_Lower,ymax=pred\_Upper)+ geom\_point(aes(x1),col='Red') +geom\_point(aes(x1,pred\_testy),col='Yellow')+ggtitle('Confidence Error Bar Plot')+theme(plot.title = element\_text(hjust = 0.5))+xlab('Input Data Set') + ylab('Output Data Set')

```

*#Task 3 APPROXIMATE BAYESIAN COMPUTATION (ABC)*

*#1 2 parameters with largest absolute least squares estimation*

```{r}

meg\_mean1<-colSums(abs(meg\_mod1))/nrow(meg\_mod1)

head(meg\_mean1)

meg\_mean2<-colSums(abs(meg\_mod2))/nrow(meg\_mod2)

head(meg\_mean2)

meg\_mean3<-colSums(abs(meg\_mod3))/nrow(meg\_mod3)

head(meg\_mean3)

meg\_mean4<-colSums(abs(meg\_mod4))/nrow(meg\_mod4)

head(meg\_mean4)

meg\_mean5<-colSums(abs(meg\_mod5))/nrow(meg\_mod5)

head(meg\_mean5)

```

*#model 1 has the largest absolute least squares estimate Input 3 & Input 5*

```{r}

model1<-**as**.data.frame(meg\_mod1)

head(model1)

```

prior **for** parameters Input 3 **and** input 5

```{r}

p1<-runif(min = min(model1$input\_3),max=max(model1$input\_3),nrow(model1))

head(p1)

p2<-runif(min = min(model1$input\_5),max=max(model1$input\_5),nrow(model1))

head(p2)

```

Sample Value

```{r}

n=50

```

Importing **for** ABC Rejection

```{r}

library(EasyABC)

```

Plot Joint Distribution of parameters

```{r}

sample\_model <-function(x)

{

c(x[3]^3+p1,x[5]^5+p2)

}

prior <- list(c('unif',0,1))

```

ABC rejection **with** tolerance of 0.2

```{r}

simulation <- ABC\_rejection(model = sample\_model,prior = prior,nb\_simul = n,tol = 0.2)

densityPlot(simulation$param,xlab = 'stimulated Parameters',main = 'joint distribution of 2 parameters input 3 & input 5',col = 'Orange')

```

plot marginal distribution of Input 3 parameter

```{r}

sample\_model1<-function(x)

{

x[3]^3+p1

}

prior1 <- list(c('unif',0,1))

```

ABC rejection **with** tolerance of 0.2

```{r}

simulation1<-ABC\_rejection(model = sample\_model1,prior = prior1,nb\_simul = n,tol = 0.2)

densityPlot(simulation1$param,xlab = 'stimulated Parameters',main = 'joint distribution of input 3',col = 'Violet')

```

plot marginal distribution of Input 5 parameter

```{r}

sample\_model2<-function(x)

{

x[5]^5+p2

}

prior2 <- list(c('unif',0,1))

```

ABC rejection **with** tolerance of 0.2

```{r}

simulation2<-ABC\_rejection(model = sample\_model2,prior = prior2,nb\_simul = n,tol = 0.2)

densityPlot(simulation2$param,xlab = 'stimulated Parameters',main = 'joint distribution of input 5',col = 'Red')

```

*#Result*

The density curves seems to normal distributed **for** both the marginal **and** joint distribution of the 2 posterior parameters.