Coventry University

Course Work

Module Code: 7089CEM

Modelling Brain Signals using Nonlinear Regression

MODULE LEADER: Dr Fei He

Name: Jusel Justin

Student ID: 12646627

Introduction:

Magnetoencephalography (MEG) is that the newest, most advanced technique of recording and evaluating the brain whereas it's actively performing. This recording provides an instantaneous measuring of the continuing function of traditional neurons and may pinpoint the situation of awry neurons. Million are often used either to guage the brain's spontaneous activity (e.g., for epilepsy) or to ascertain its response to specific external stimuli (e.g., for mapping motor and sensory areas, language, vision and different functions). Brain cells (neurons) move with one another by generating electrical voltages. The flow of electrical current produces a field, which may then be recorded victimization sensitive magnetic sensors. as a result of the strength of the magnetic field created by the brain is therefore small, terribly specialized instrumentation is needed to choose up the signal. The non-invasive technology of magnetoencephalography (MEG) is used to study human brain function. It indicates where in the brain activity is created and permits microsecond evaluation on continuous brain activity.

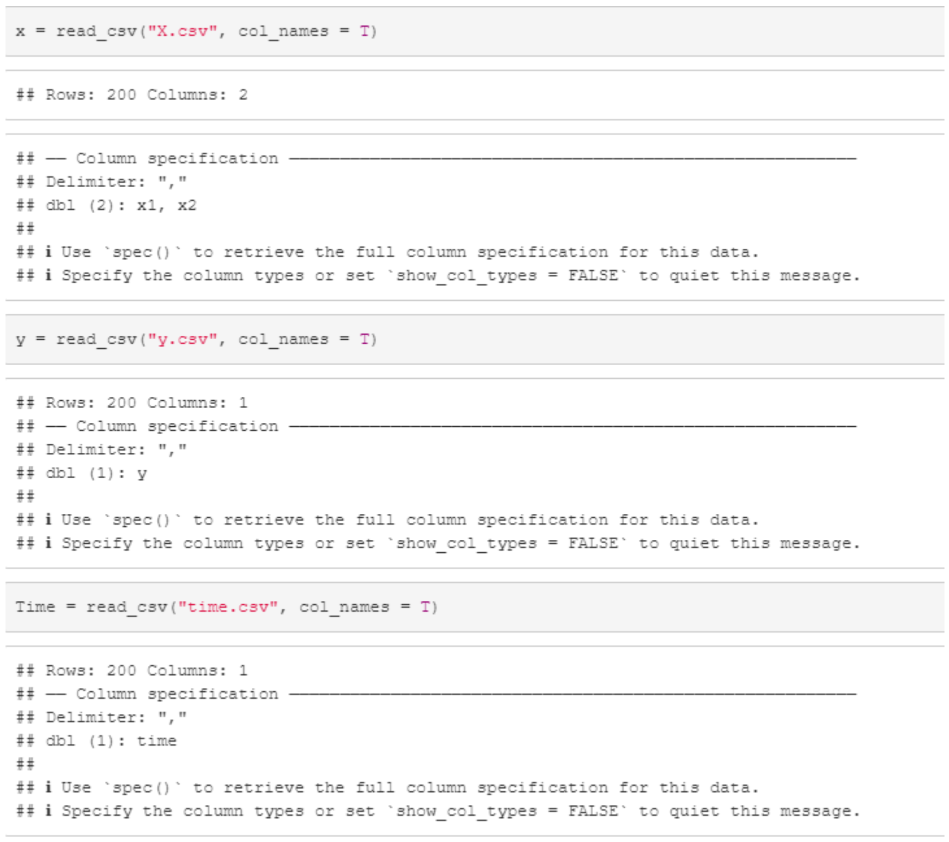
Nerve cells include electrochemical traits that effects the motion at the molecular level. The overall effect of this gradual ionic present day go with the drift produces electromagnetic fields. These brain-generated neuromagnetic impulses that are relatively vulnerable. The most important cause of this studies is to pick the top of the line regression version for decide the brain's response to sound from a fixed of nonlinear regression models.

Task 1:

This is the first task where I have investigated the time series plots, distribution plots, correlation plots, and box plots to undertake exploratory data analysis.

Time series charts represents a series of data points collected over a specified reporting period. The x-axis plots time and the y-axis plots data points. Generally, the x axis of the chart or graph is used to plot increments of time and the y axis indicates values of the variable that is being measured. As a result a series of peaks and valleys are generated.

The implementation and preliminary data analysis are represented below



```
x = read_csv("X.csv", col_names = T)
```

```
## Rows: 200 Columns: 2
```

```
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl (2): x1, x2
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
y = read_csv("y.csv", col_names = T)
```

```
## Rows: 200 Columns: 1
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl (1): y
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
Time = read_csv("time.csv", col_names = T)
```

```
## Rows: 200 Columns: 1
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl (1): time
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

diagram1: Loaded data files

After succesful loading the data files, the input and output audio MEG signals time series plots are drawn.

Time series plot of MEG input signal X

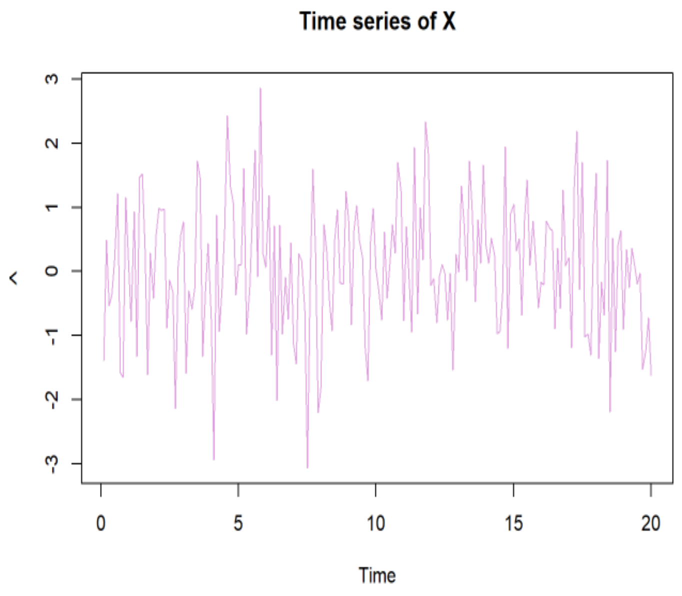


diagram 2: Time series plot against input X

Time series plot of MEG output signal Y

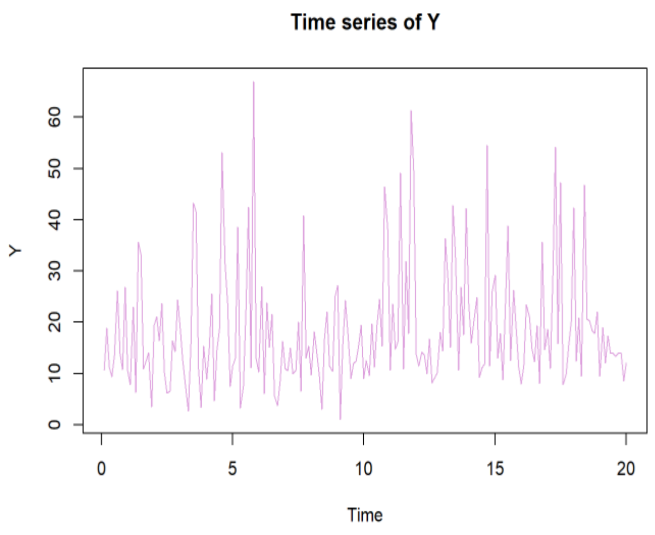


Diagram 3: Time series plot against output y

Distribution for each I/O Signal:

The distribution plot is useful for investigating the extent and distributions of statistical information in groups. Data is represented as a series of measurement points along each axis. It is used to check the distribution of data.Distributions for every input and output signal.

Distribution plot for histogram of X:

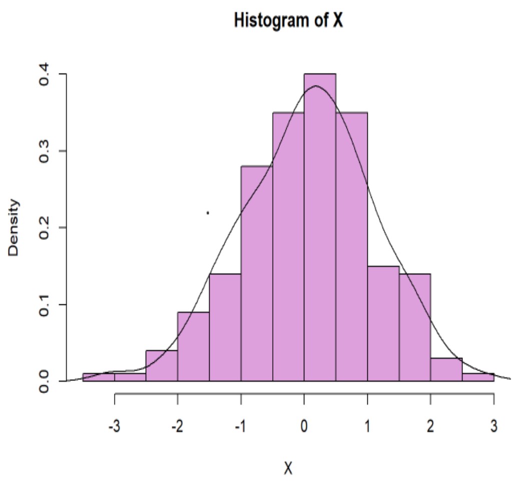


Diagram 4: Histogram plot of X

Distribution plot for histogram of Y:

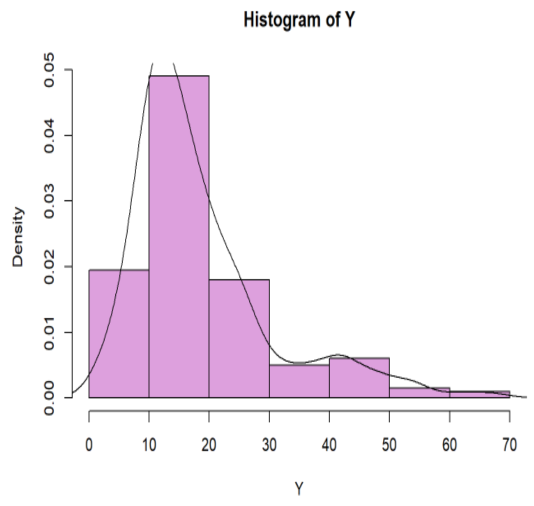


Diagram 5: Histogram plot of Y

Correlation and Scatter plots(against the input audio and output brain signals)to identify the dependencies:

Correlation coefficients are used to measure the strength of the linear relationship between two variables that is between sound input and output brain signals.

A correlation coefficient, abbreviated as r, is a measurement of the degree and directions of a connection between the two parameters.

In this section correlation has been plotted between sound input and output brain signals.

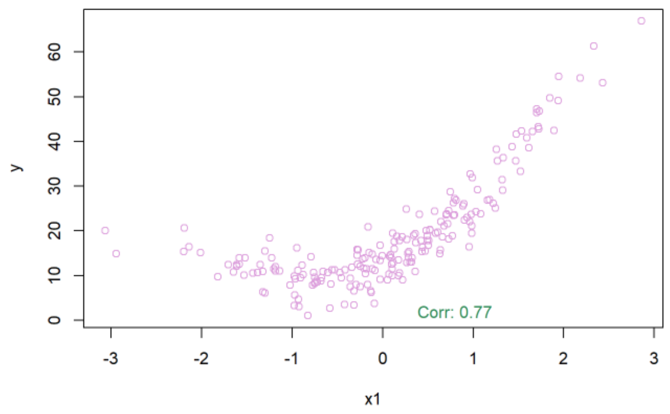


Diagram 6: Scatter plot between input and output

Boxplots

A box plot represents a box from the first quartile to the third quartile. A vertical line goes through the box at the median. The whiskers go from each quartile to the minimum or maximum. It represents a statistical data. As the name suggests "box plot" infers to a chart that resembles a rectangular box having dotted lines extending from the top and bottom.

This box plots represents output brain signals to examine the effect of sound signal categories

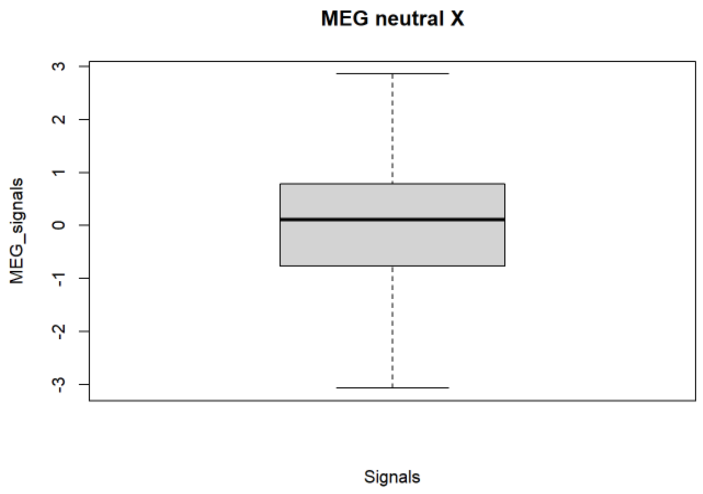


Diagram 7: Boxplot for neutral and emotional audio

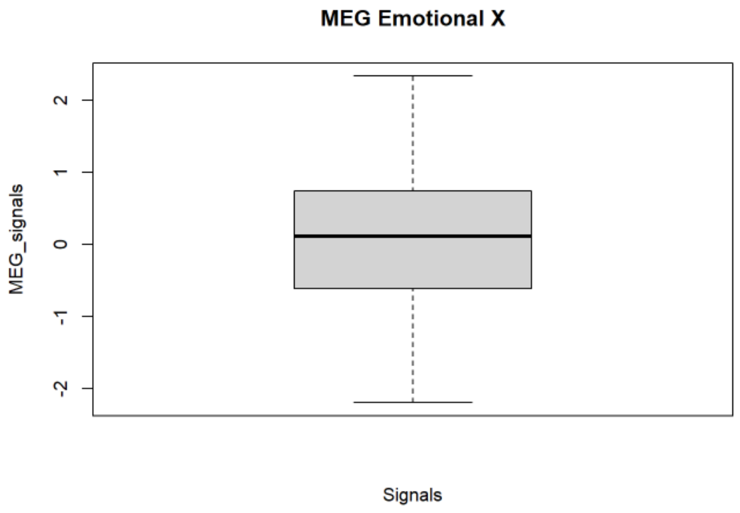


Diagram 8: Boxplot of MEG Emotional X

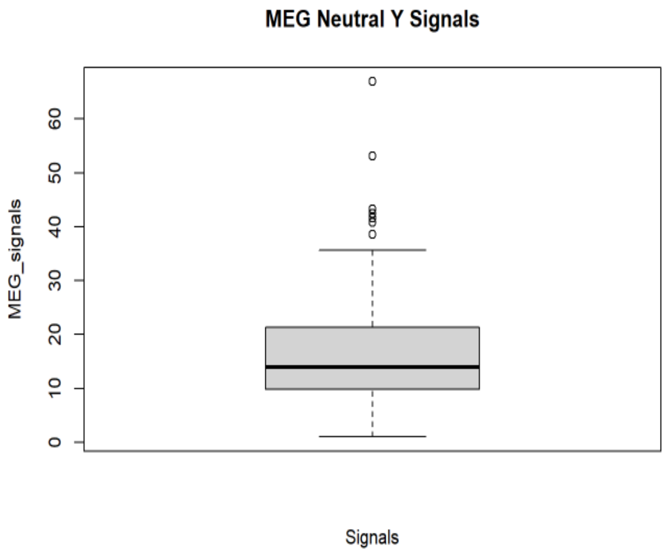


Diagram 9: Boxplot of MEG Neutral Y Signal

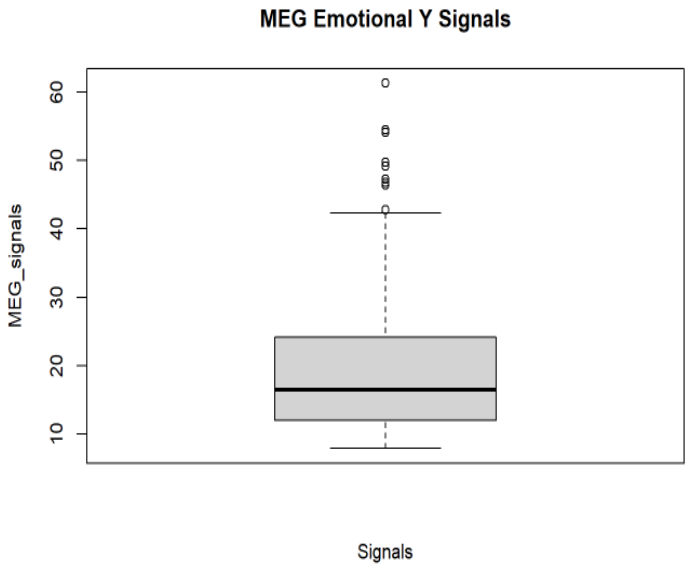
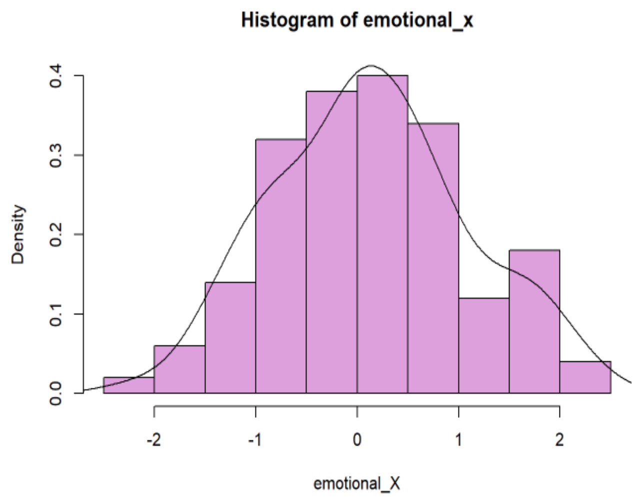


Diagram 10: Boxplot of MEG Emotional Y Signal

Performing the preliminary data analysis for each represented input sound signals separately using the distribution plots:

Distribution plots of Emotional audio and Neutral audio



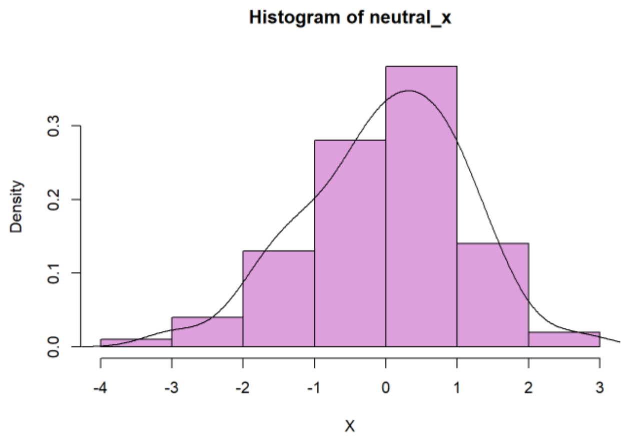
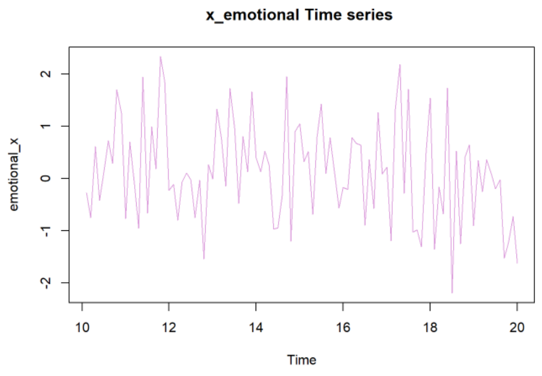


Diagram 11: Distribution plots of emotional Audio and Neutral audio

Time Series plots:



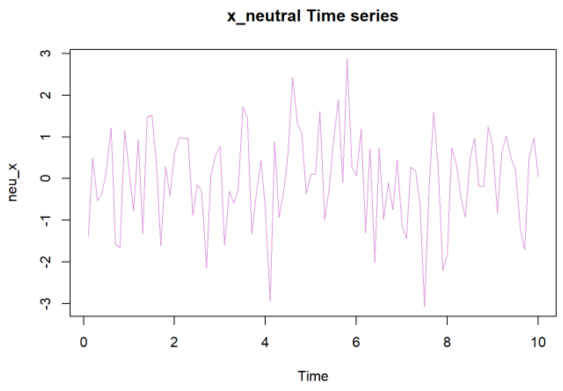


Diagram 12: Time series plots of emotional audio and Neutral audio

Task 2:

Regression modeling the brain response (MEG) to a sound signal

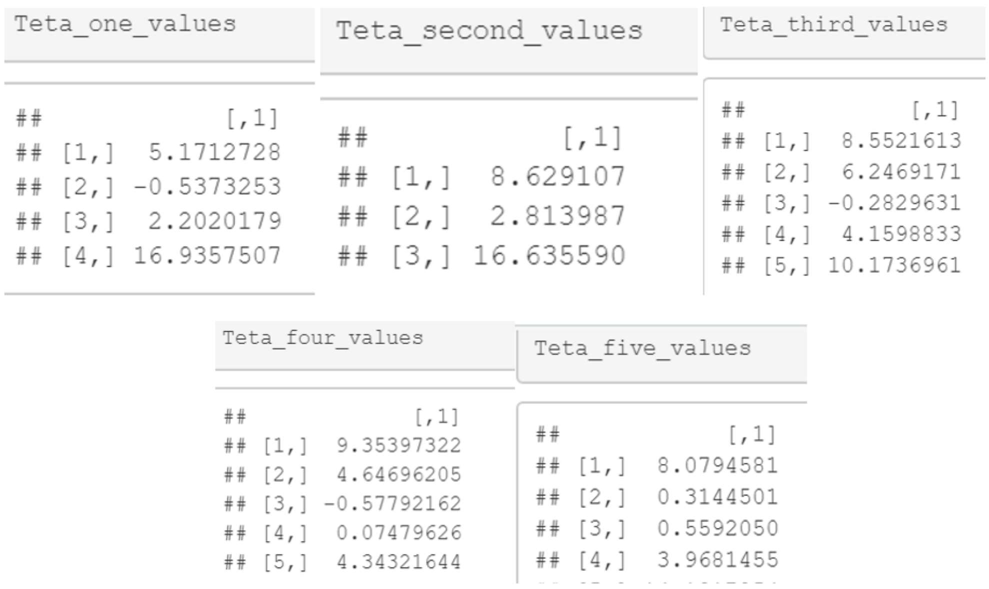
Candidate models are with the following structures: Model 1: y = $\theta 1x1$ 3 + $\theta 2x1$ 5 + $\theta 3x2$ + $\theta bias$ + ε Model 2: y = $\theta 1x1$ + $\theta 2x2$ + $\theta bias$ + ε Model 3: y = $\theta 1x1$ + $\theta 2x1$ 2 + $\theta 3x1$ 4 + $\theta 4$ $x2$ + $\theta bias$ + ε Model 4: y = $\theta 1x1$ + $\theta 2x1$ 2 + $\theta 3x1$ 3 + $\theta 4x1$ 5 + $\theta 5x2$ + $\theta bias$ + ε Model 5: y = $\theta 1x1$ + $\theta 2x1$ 3 + $\theta 3x1$ 4 + $\theta 4x2$ + $\theta bias$ + ε

We need to find the best suitable statistical model that can explain the correlation between the two audio signal and the output brain signal, as well as how that relationship changes depending on the content of the input audio signal. A polynomial regression model can be used to characterize this relationship.

Task 2.1

Estimate model parameters $\boldsymbol{\theta}$ = {$\theta 1$, $\theta 2$, ⋯ , $\theta bias$} T for every candidate model using Least Squares ($\boldsymbol{\theta}$ = ($\mathbf{X}$ $T\mathbf{X}$) −1$\mathbf{X}$ $T\mathbf{y}$), using the provided sound input and output MEG datasets (use all the data for training).

The following diagram 13 shows the least square results of all models.



```
Teta_one_values

##              [,1]
## [1,]  5.1712728
## [2,] -0.5373253
## [3,]  2.2020179
## [4,] 16.9357507
```

```
Teta_second_values

##            [,1]
## [1,]  8.629107
## [2,]  2.813987
## [3,] 16.635590
```

```
Teta_third_values

##              [,1]
## [1,]  8.5521613
## [2,]  6.2469171
## [3,] -0.2829631
## [4,]  4.1598833
## [5,] 10.1736961
```

```
Teta_four_values

##               [,1]
## [1,]  9.35397322
## [2,]  4.64696205
## [3,] -0.57792162
## [4,]  0.07479626
## [5,]  4.34321644
```

```
Teta_five_values

##             [,1]
## [1,]  8.0794581
## [2,]  0.3144501
## [3,]  0.5592050
## [4,]  3.9681455
```

Diagram 13: Least square results for all 5 models

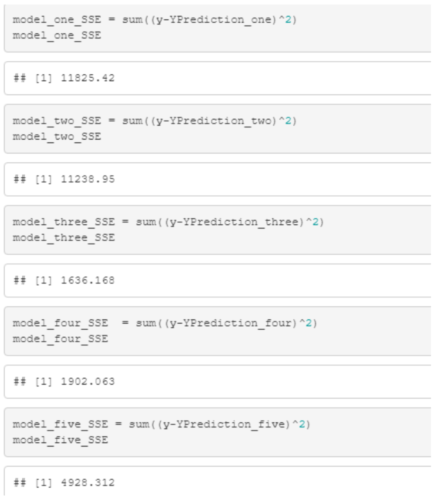
Task 2.2

Based on the estimated model parameters, compute the model residual (error) sum of squared errors (RSS), for every candidate model. RSS = ∑ (yi − $\mathbf{x}i\boldsymbol{\theta}$) 2 n i=1 Here $\mathbf{x}i$ denotes the i th row (i th data sample) in the input data matrix $\mathbf{X}$, $\boldsymbol{\theta}$ is a column vector.

In this part we are determining the residual sum of squares for all models.

The residual sum of squares (RSS) is **a statistical technique which is used to calculate the measurement of the the amount of variance in a data set that is not explained by a regression model itself**. Rather, it estimates the variance in the residuals, or error term.

This matric measures the disparity among data and an estimating model. A low RSS implies a good result between the model and the data.



```
model_one_SSE = sum((y-YPrediction_one)^2)
model_one_SSE
```

```
## [1] 11825.42
```

```
model_two_SSE = sum((y-YPrediction_two)^2)
model_two_SSE
```

```
## [1] 11238.95
```

```
model_three_SSE = sum((y-YPrediction_three)^2)
model_three_SSE
```

```
## [1] 1636.168
```

```
model_four_SSE  = sum((y-YPrediction_four)^2)
model_four_SSE
```

```
## [1] 1902.063
```

```
model_five_SSE = sum((y-YPrediction_five)^2)
model_five_SSE
```

```
## [1] 4928.312
```

Diagram 13: RSS error results for 5 models

Task 2.3:

Compute the log-likelihood function for every candidate model:

ln $p(D|\boldsymbol{\theta})$ = − n 2 ln(2π) − n 2 ln($\hat{\sigma}$ 2) − 1 2$\hat{\sigma}$ 2 RSS Here, $\hat{\sigma}$ 2 is the variance of a model's residuals (prediction errors) distributions $\hat{\sigma}$ 2 = RSS/(n − 1) , with n the number of data samples.

In this section we are calculating the log likelihood function for all models.

Log likelihood Function:

We are going to understand how well a model can fit a data.

The following table represents the LLF results for all the five models.

Models	LLF
Model 1	**-691.7579**
Model 2	**-686.6713**
Model 3	**-493.9684**
Model 4	**-509.0267**
Model 5	**-604.2324**

Task 2.4:

Compute the Akaike information criterion (AIC) and Bayesian information criterion (BIC) for every candidate model: AIC = 2k − 2 ln $p(D|\boldsymbol{\theta})$ BIC = $k \cdot$ l n(n) − 2 ln $p(D|\boldsymbol{\theta})$ Here ln $p(D|\boldsymbol{\theta})$ is the log-likelihood function obtained from Task 2.3 for each model, k is the number of estimated parameters in each candidate model.

Determine the Akaike information criterion (AIC) and Bayesian information criterion (BIC) for every candidate model:

Akaike information criterion will help us understand how well a model fits the data without over-fitting it. The model with the lowest AIC score is predicted to achieve a better balance between its capacity to fit the large dataset and its ability to stop overfitting problem.

AIC and BIC of all candidate models.

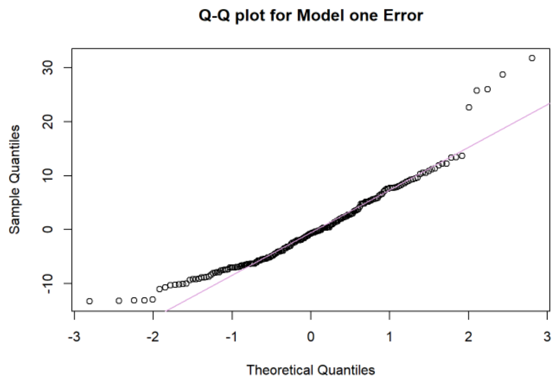
Models	AIC	BIC
Model 1	**1391.516**	**1404.709**
Model 2	**1379.343**	**1389.238**
Model 3	**997.9368**	**1014.428**
Model 4	**1030.053**	**1049.843**
Model 5	**1218.465**	**1234.956**

Task 2.5

Check the distribution of model prediction errors (residuals) for each candidate model. Plot the error distributions, and evaluate if those distributions are close to Normal/Gaussian (as the output MEG has additive Gaussian noise), e.g. by using Q-Q plot.

Q-Q Plot:

The quantile-quantile plot, also known as the Q-Q plot, is a useful visual tool used for estimating the fit of a standard normal distribution by examining the patterns of the plot.



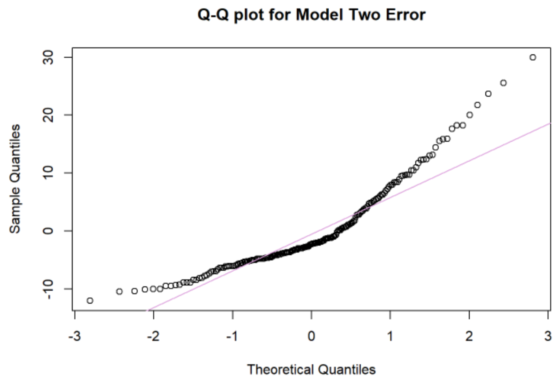
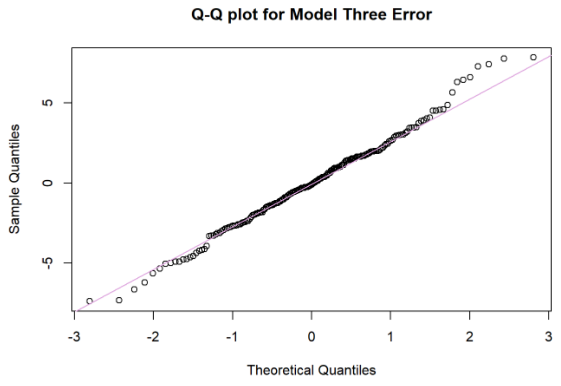


Diagram 14: Q-Q plot for model 1 and Mode 2



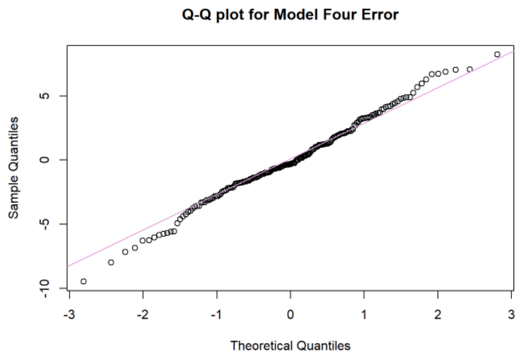
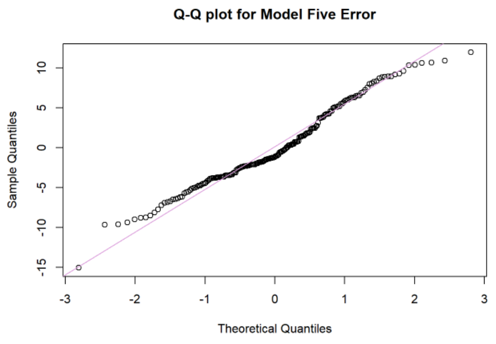


Diagram 15: Q-Q plot for model 3 and Mode 4



Digure 15: Q-Q plot for model-5

Task 2.6:

Select ‘best’ regression model according to the AIC, BIC and distribution of model residuals from the 5 candidate models, and explain why you would like to choose this specific model.

Considering the previous task results model 3 Log likelihood value is -493.9684 and BIC, AIC values are 1014.428 and 997.9368. The majority of data follow a straight normal line, When compared to other model results Model 3 got the best fit line. As, the model with lowest result value of AIC, BIC and highest Value of LLF is the best fit model. Model-3 has highest value when compared to other values. Thus proved Model 3 is the best fit model.

Task 2.7:

Split the input (sound) and output (MEG) dataset (**X** and **y**) into two parts: one part used to train the model, the other used for testing (e.g. 70% for training, 30% for testing). For the selected ‘best’ model, 1) estimate model parameters use the training dataset; 2) compute the model’s output/prediction on the testing data; and 3) also compute the 95% (model prediction) confidence intervals and plot them (with error bars) together with the model prediction, as well as the testing data samples.

Spliting the input and output (MEG) dataset. For every point,the 75% confidence intervals are computed and shown. The first part is used to train the data model, the other used for testing. Here model 3 data is selected and divided into 70:30 ratio. 70 percentage is used for training and 30 percent used for testing the data . We need to calculate the training data least squares, and error variance values. Then test data is used to compute the parameters accordingly. Which gives valuable insights as the outcomes.

Confidence Interval:

In R calculating R is very easy. We use the test since there is no function in R that will simply build a confidence interval. The 95 percent confidence interval was computed here.

Length of the test data is stored in ‘n1’ value. We use CI=1.96 for 95% confidence interval.

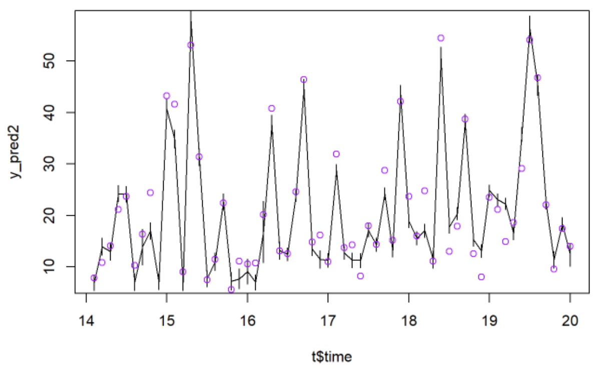


Diagram 16: Plot with error bars and CI

Task 3:

Approximate Bayesian Computation (ABC):

Using 'rejection ABC' method to compute the posterior distributions of the 'selected' regression model parameters in Task 2. A collection of computer techniques based on Bayesian statistics is known as approximate Bayesian computing (ABC). In Task 2, the posterior distributions of the 'selected' regression model parameters were estimated using the 'rejection ABC' approach.

Here we are determining the two parameter posterior distributions with largest absolute values in the least square results.

Posterior Distribution of Parameter One

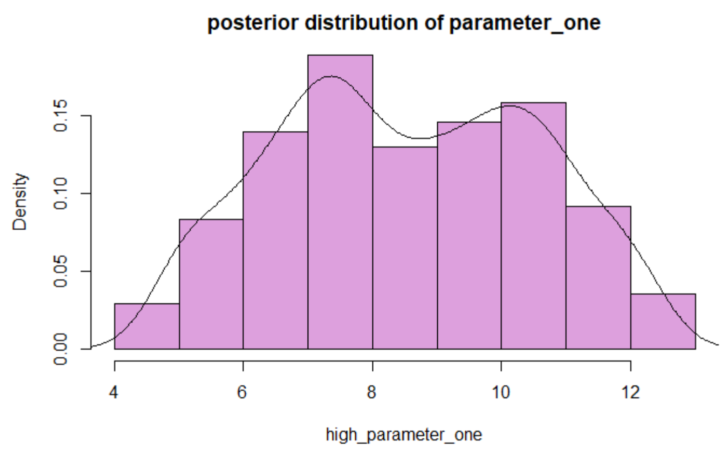


Figure 17: Posterior Distribution of Parameter One

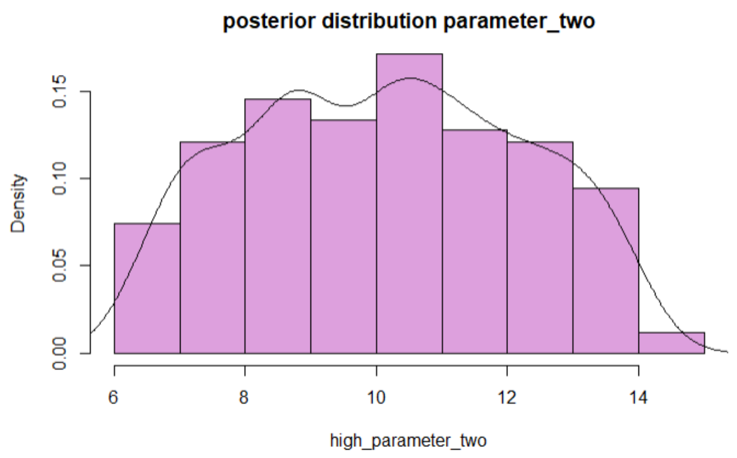


Figure 18: Posterior Distribution of Parameter Two

Rejection ABC for two parameters has been computed and plotted using the above uniform prior. The two rejected vectors uniform plot has been shown in figure 19.

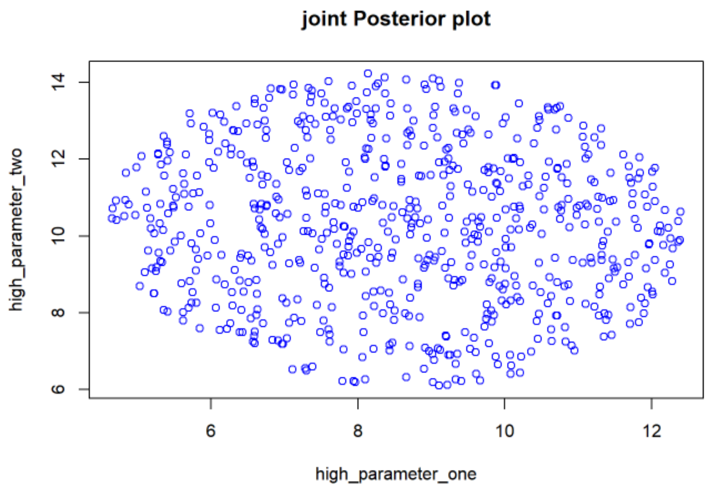


Figure 19: Joint and Marginal Posterior Plot

Conclusion:

The main motive of this coursework was to find the best regression model that could determine the brain response to a advertising sound signal which is emotional and neutral. The simulated MEG time series data has been taken and it is divided into two separated files. After applying models like the least square, Log likelihood functions, AIC, BIC and Distribution plots to all candidate models, Model 3 has been considered to be known as the best fit model. Because, when compared to the other model, results of model 3 has highest LLF and lowest AIC, BIC. Therefore, model 3 is the best regression model to evaluate the brain response to a sound signal.

References:

F. Varela, J.P. Lachaux, E. Rodriguez, J. Martinerie

The brainweb: phase synchronization and large-scale integration

Nat Rev Neurosci, 2 (2001), pp. 229-239

. “The Brief History of Brain Computer Interfaces” by Hans Berger’s

2. “An introduction to statistical modelling” by Wojtek j Krzanowski

Singh S. P. (2014). Magnetoencephalography: Basic principles. *Annals of Indian Academy of Neurology*, *17*(Suppl 1), S107–S112. https://doi.org/10.4103/0972-2327.128676

Mikael Sunnåker,Giovanni,A,Busetto, Elina Numminen ,Jukka,C.(2013).Approximate Bayesian Computation[Online], https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.100280
G. Buzsaki, A. Draguhn

Neuronal oscillation in cortical networks

Science, 304 (2004), pp. 1926-1929

Google Scholar

Appendix:

Source Code

# Load dataset

```
#setwd("C:\\Users\\user\\Downloads")

require(readr)

x = read_csv("X.csv", col_names = T)

y = read_csv("y.csv", col_names = T)

Time = read_csv("time.csv", col_names = T)


#Histogram of X

First_Dense <- density(x$x1)

hist(x$x1,prob = TRUE,main = "Histogram of X",col='plum',xlab = "X")

lines(First_Dense)


#Histogram of y

Second_Dense <- density(y$y)

hist(y$y,prob = TRUE,main = "Histogram of Y",col='plum',xlab = "Y")

lines(Second_Dense)


#Time series plots

plot(Time$time,x$x1,type='l',xlab = "Time",ylab = "X",main = 'Time series of X',col='plum')

plot(Time$time,y$y,type='l',xlab = "Time",ylab = "Y",main = 'Time series of Y',col='plum',)
```

```
#Correlation
correlation <- cor(x$x1, y$y)
plot(x$x1,y$y,col='plum',xlab = 'x1', ylab = 'y')
text(paste("Corr:", round(correlation, 2)), x=0.8 ,y=2.0,col='seagreen')
neu_x = x$x1[1:100]
neu_y = y$y[1:100]
emotional_x = x$x1[101:200]
emotional_y = y$y[101:200]
neu_t = Time$time[1:100]
emotional_t= Time$time[101:200]

#boxplots
boxplot(neu_x, main="MEG neutral X",
   xlab="Signals", ylab="MEG_signals",
   names=c("neutral_input"))

#Boxplot emotional-X
boxplot(emotional_x, main="MEG Emotional X",
   xlab="Signals", ylab="MEG_signals",
   names=c("emotional_input"))

#Boxplot of Neutral Y
boxplot(neu_y, main="MEG Neutral Y Signals",
   xlab="Signals", ylab="MEG_signals",
   names=c("neutral_output"))

boxplot(emotional_y, main="MEG Emotional Y Signals",
   xlab="Signals", ylab="MEG_signals",
   names=c("emotional_output"))

#Distribution plots
```

```
Dens1_emot <- density(emotional_x)
hist(emotional_x,prob = TRUE,main = "Histogram of emotional_x",xlab = "emotional_X",col=
'plum')
lines(Dens1_emot)
dens1_neutral <- density(neu_x)
hist(neu_x,prob = TRUE,main = "Histogram of neutral_x",xlab = "X",col= 'plum')
lines(dens1_neutral
plot(emotional_t,emotional_x,type='l',xlab = "Time",ylab = "emotional_x",col ='plum',main =
'x_emotional Time series ')
plot(neu_t,neu_x,type='l',xlab = "Time",ylab = "neu_x",col ='plum',main = 'x_neutral Time series')
```

Task 2

```
One = matrix(1 , length(x$x1),1)
One_model = cbind(x$x1^3,x$x1^5,x$x2,One)
Teta_one_values = solve(t(One_model) %*% One_model) %*% t(One_model) %*% y$y
Teta_one_values
YPrediction_one = One_model %*% Teta_one_values
Two_model = cbind(x$x1,x$x2,One)
Teta_second_values  = solve(t(Two_model) %*% Two_model) %*% t(Two_model) %*% y$y
Teta_second_values
YPrediction_two = Two_model%*% Teta_second_values

Three_model = cbind(x$x1,x$x1^2,x$x1^4,x$x2,One)
Teta_third_values = solve(t(Three_model) %*% Three_model) %*% t(Three_model) %*% y$y
Teta_third_values
YPrediction_three = Three_model%*% Teta_third_values

Four_model = cbind(x$x1,x$x1^2,x$x1^3,x$x1^5,x$x2,One)
Teta_four_values = solve(t(Four_model) %*% Four_model) %*% t(Four_model) %*% y$y
Teta_four_values
YPrediction_four = Four_model%*% Teta_four_values
```

```
Five_model= cbind(x$x1,x$x1^3,x$x1^4,x$x2,One)
Teta_five_values = solve(t(Five_model) %*% Five_model) %*% t(Five_model) %*% y$y
Teta_five_values
YPrediction_five = Five_model%*% Teta_five_values
```

Task 2.2

```
model_one_SSE = sum((y-YPrediction_one)^2)
model_one_SSE


model_two_SSE = sum((y-YPrediction_two)^2)
model_two_SSE

model_three_SSE = sum((y-YPrediction_three)^2)
model_three_SSE

model_four_SSE  = sum((y-YPrediction_four)^2)
model_four_SSE

model_five_SSE = sum((y-YPrediction_five)^2)
model_five_SSE
```

Task 2.3

```
loglikelihood_one=-((n/2)*log(2*pi))-((n/2)*log(model_one_SSE/(n-1)))-((1/(2*model_one_SSE/(n-
1)))*model_one_SSE)
loglikelihood_one

loglikelihood_two=-((n/2)*log(2*pi))-((n/2)*log(model_two_SSE/(n-1)))-((1/(2*model_two_SSE/(n-
1)))*model_two_SSE)
```

```
loglikelihood_two

loglikelihood_three=-((n/2)*log(2*pi))-((n/2)*log(model_three_SSE/(n-1)))-((1/(2*model_three_SSE/(n-1)))*model_three_SSE)
loglikelihood_three

loglikelihood_four=-((n/2)*log(2*pi))-((n/2)*log(model_four_SSE/(n-1)))-((1/(2*model_four_SSE/(n-1)))*model_four_SSE)
loglikelihood_four

loglikelihood_five=-((n/2)*log(2*pi))-((n/2)*log(model_five_SSE/(n-1)))-((1/(2*model_five_SSE/(n-1)))*model_five_SSE)
loglikelihood_five
```

Task 2.4:

```
Model_one_AIC=(2*4)-2*(loglikelihood_one)
Model_one_BIC=(4*log(n))-2*(loglikelihood_one)
Model_one_AIC
Model_one_BIC

Model_two_AIC=(2*3)-2*(loglikelihood_two)
Model_two_BIC=(3*log(n))-2*(loglikelihood_two)
Model_two_AIC
Model_two_BIC

Model_three_AIC=(2*5)-2*(loglikelihood_three)
Model_three_BIC=(5*log(n))-2*(loglikelihood_three)
Model_three_AIC
Model_three_BIC

Model_four_AIC=(2*6)-2*(loglikelihood_four)
Model_four_BIC=(6*log(n))-2*(loglikelihood_four)
```

```
Model_four_AIC
Model_four_BIC

Model_five_AIC=(2*5)-2*(loglikelihood_five)
Model_five_BIC=(5*log(n))-2*(loglikelihood_five)
Model_five_AIC
Model_five_BIC
```

Task 2.5:

```
Model_SSE1=y-YPrediction_one
qqnorm(Model_SSE1$y,main = "Q-Q plot for Model one Error")
qqline(Model_SSE1$y,col="plum")

Model_SSE2=y-YPrediction_two
qqnorm(Model_SSE2$y,main = "Q-Q plot for Model Two Error")
qqline(Model_SSE2$y,col="plum")
Model_SSE3=y-YPrediction_three
qqnorm(Model_SSE3$y,main = "Q-Q plot for Model Three Error")
qqline(Model_SSE3$y,col="plum")

Model_SSE4=y-YPrediction_four
qqnorm(Model_SSE4$y,main = "Q-Q plot for Model Four Error")
qqline(Model_SSE4$y,col="plum")

Model_SSE5=y-YPrediction_five
qqnorm(Model_SSE5$y,main = "Q-Q plot for Model Five Error")
qqline(Model_SSE5$y,col="plum")
```

Task 2.7:

```
set.seed(101)
```

```
data_one=data.frame(x,y)
sample <- sample.int(n = nrow(data_one), size = floor(.70*nrow(data_one)), replace = F)
train <- data_one[sample, ]
x_train <- data.frame(train$x1,train$x2)
names(x_train) <- c('x1', 'x2')
y_train <- data.frame(train$y)
names(y_train) <-c('y')
test  <- data_one[-sample, ]
x_test <- data.frame(test$x1,test$x2)
names(x_test) <- c('x1', 'x2')
y_test <- data.frame(test$y)
names(y_test) <- c('y')

Ones_1 = matrix(1 , length(x_train$x1),1)
X_train_one=cbind(x_train$x1,x_train$x1^2,x_train$x1^4,x_train$x2,Ones_1)
Tetaparameter_one = solve(t(X_train_one) %*% X_train_one) %*% t(X_train_one) %*% y_train$y
Tetaparameter_one
Ones_2 = matrix(1 ,length(x_test$x1),1)
X_test_two=cbind(x_test$x1,x_test$x1^2,x_test$x1^4,x_test$x2,Ones_2)
y_pred2 = X_test_two %*% Tetaparameter_one

error_one = y_test-y_pred2
RSS = sum((error_one)^2)
RSS

n1=nrow(X_test_two)
sigma_two = RSS/( n1 - 1 )
covarience =  sigma_two * (solve(t(X_test_two) %*% X_test_two))
number_of_parameters=5
Y_Hat = matrix(0 , n1 , 1)
for( i in 1:n1){
  i_X = matrix( X_test_two[i,] , 1 , number_of_parameters )
  Y_Hat[i,1] = i_X %*% covarience %*% t(i_X)
}
CI = 1.96 * sqrt(Y_Hat)
t=Time[141:200,1]
plot(t$time,y_pred2,type='l')
segments(t$time,y_pred2-CI,t$time,y_pred2+CI)
points(t$time,y_test$y,col="purple")
`Task 3:

high_parameter_one=10.1736961
high_parameter_two=8.5521613
param_low1=high_parameter_one-1.5*(high_parameter_one)
param_upper1=high_parameter_one+1.5*(high_parameter_one)
param_low2=high_parameter_two-1.5*(high_parameter_two)
param_upper2=high_parameter_two+1.5*(high_parameter_two)

AR=NULL
SSE=NULL
for(i in 1:10000){
  sample_one=runif(1,param_low1,param_upper1)
  sample_two=runif(1,param_low2,param_upper2)
  thetavalues=c(sample_one,6.2469171,-0.2829631,4.1598833,sample_two)
  yhat = Three_model %*% thetavalues
  SSE = sum((y$y - yhat)^2)
  if (SSE < 5070){
    AR= rbind(AR,thetavalues)
  }
}
```
# posterior distribution of parameter_one

```
hist(AR[,1],probability = TRUE,main = "posterior distribution of parameter_one",col='plum',
xlab="high_parameter_one")
```

```
lines(density((AR[,1])))
```

#posterior distribution of parameter_two

```
hist(AR[,5],probability = TRUE,main = "posterior distribution parameter_two",
col='plum',xlab="high_parameter_two")
lines(density(AR[,5]))


plot(AR[,1],AR[,5],main = "joint Posterior
plot",xlab="high_parameter_one",col="blue",ylab="high_parameter_two")
```