Coventry University
Faculty of Engineering, Environment and Computing

7089CEM
Introduction to Statistical Methods for Data Science

Module Leader: Dr. Fei He
Student Name: Aryalakshmi Nellippillipathil Babu

Modelling EEG signals using polynomial regression

CONTENTS

Introduction

The objective of this assignment is to identify the best regression model (from a potential set of nonlinear regression models) that can adequately describe the brain activity elicited during guided meditation. The information is presumably gathered during a neuroscience experiment in which a person is encouraged to practise guided meditation while receiving instructions over speakers. The modulation of neural activity in two different brain regions during the meditation is of interest to the researchers.
Particularly, electroencephalography (EEG) is used to gauge the brain activity in the right auditory cortex and prefrontal cortex. Whereas area (2) is linked to executive function, planning, and consciousness, area (1) is responsible for processing auditory experiences. The prefrontal cortex is projected to have a nonlinear association, in contrast to the auditory cortex, which the researchers believe is linearly related to the audio signal (i.e., the voice of the mediation guide). Using nonlinear regression modelling, this report examines these correlations.
The two distinct Excel files contain the "simulated" EEG time-series data and the sound signal. The sound signal y is contained in the y.csv file, while the X.csv file comprises the EEG signals x1 and x2 that were measured from the prefrontal and auditory cortices. The sampling times for all three signals are listed in seconds in the file time.csv. A total of 2 minutes' worth of signal data were gathered at a sampling rate of 20 Hz. Due to distortions during recording, all signals are subject to additive noise with unknown variance (assumed to be independent and identically distributed ("i.i.d") Gaussian with zero-mean).

For coding and resolving this problem, R programming language is utilised.

1. Preliminary data analysis

```
Sum of null values in EEG signal data: 0
[1] "First five rows in EEG signals file:"
> print(data_eeg[1:5,])
      prefrontal   auditory
[1,] -0.5623198 -2.2744913
[2,]  0.5055505 -1.4515862
[3,] -0.8578116 -0.2712149
[4,] -0.7352617 -3.4025005
[5,] -1.2443687 -2.1991619
```

Figure 1: EEG Signal data

The first five rows of the EEG signal data are shown in Figure 1. The data does not contain any null values.

```
Sum of null values in Sound signal data: 0
[1] "First five rows in Sound signal file:"
> print(data_sound[1:5,])
[1]  -2.3865260  -0.3715598 -12.5864524  -0.2481209 -10.6814373
```

Figure 2: Sound signal data

Sound signal data's first five rows are depicted in Figure 2. The data has no null values.

```
Sum of null values in Time data: 0
[1] "First five rows in time data file:"
> print(data_time[1:5,])
[1] 0.05 0.10 0.15 0.20 0.25
```

Figure 3: Sampling Time data

Sampling time data's first five rows are displayed in Figure 3. The data has no null values.

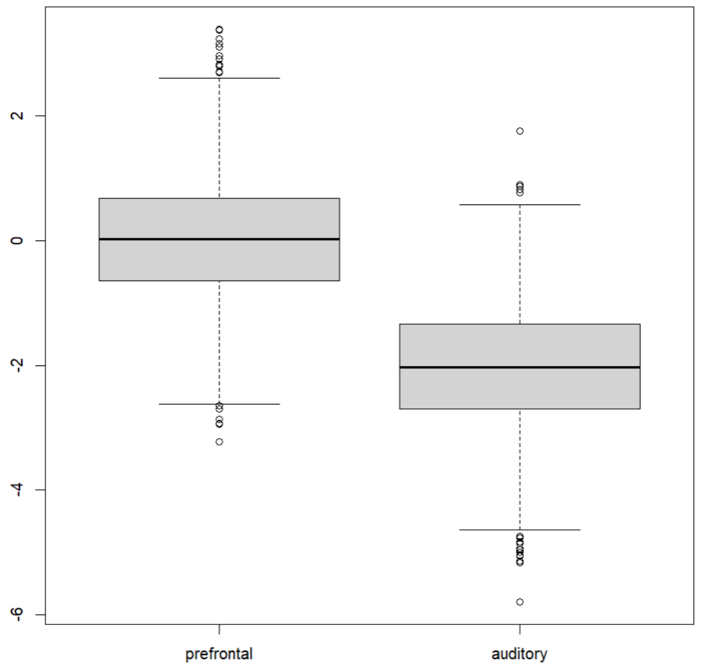

Figure 4: Boxplots for EEG signals

Figure 4's prefrontal boxplot demonstrates that this data has some outliers and a median that is close to zero. According to the auditory boxplot, this data exhibits some outliers and a median that is close to -2.

```
> boxplot.stats(data_eeg[,"prefrontal"])$out
 [1]  3.234398 -2.933734 -2.646973  2.805090 -2.690752  3.387346  2.699626  2.821873  2.917114  2.968601 -3.221049
[12] -2.858120  3.107550 -2.647741 -2.689952  2.713080  3.375305 -2.936310  3.156426
>
```

Figure 5: Outliers for EEG Prefrontal

```
> boxplot.stats(data_eeg[,"auditory"])$out
 [1]  0.8814358  0.7713153 -4.7415648  1.7647835  0.8970219 -5.0651898  0.8199376 -5.1379894 -4.8302484 -4.7598484
[11] -5.7864497 -4.9283581 -5.0497870 -4.9562220 -5.1652829 -5.0460736 -4.9905672 -4.7340947 -4.8504107
>
```

Figure 6: Outliers for EEG Auditory

The outliers observed in the prefrontal and auditory data are shown in Figures 5 and 6, respectively.

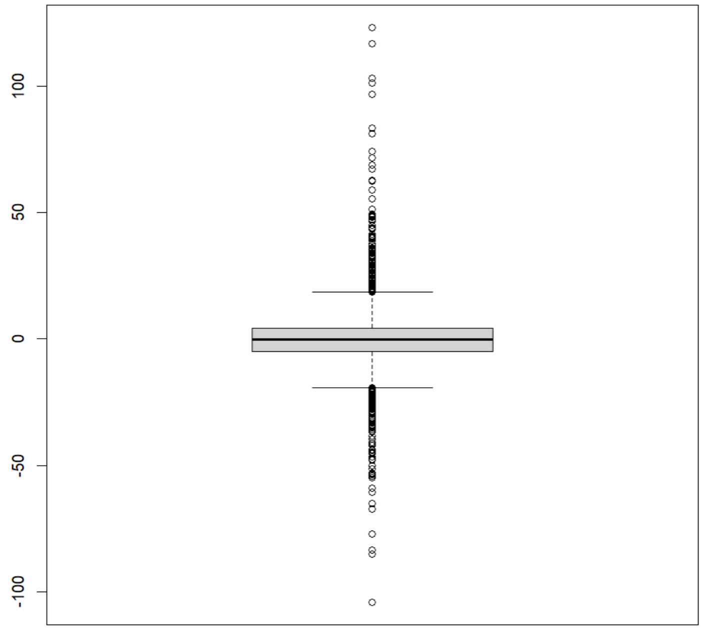

Figure 7: Boxplot for Sound signal data

Figure 7's sound signal boxplot demonstrates that this data has so many outliers and a median that is close to zero.

```
> boxplot.stats(data_sound)$out
  [1]  -54.81316  -20.08199  -24.05934   18.70339   23.57427   39.83029   18.51300   27.62835  -20.43185   23.97331
 [11]  -27.51334   48.39492  -19.81949   33.00226  -24.08229   36.16457   34.83100  -43.89455   25.05898   43.60682
 [21]   19.73322   38.93289  -27.32971  103.19974  -34.59836  -24.92184  -83.39562  -51.32239  -21.21256  -60.44009
 [31]   30.20714   23.63303   41.56379   22.25119   71.73693  -67.39525   67.04408  -21.91524  122.97410  -24.17197
 [41]   19.76737   20.68590   19.65068   31.14687   19.51008   33.65419  -46.62305   62.73246   33.74161  -32.32227
 [51]   40.44908   25.66549   29.26125  -44.58717  -24.58729   27.55516   27.31073   35.32149  -25.49952  -34.78329
 [61]   48.89890  -31.80178   21.26118  -22.28119  -44.18989  -28.39413  -40.94623  -23.22446  -41.41684  -25.67589
 [71]   74.22827   83.51235   23.01413  -26.68525   19.59926  -39.63925  -25.98711   81.05540   28.99586  -20.74497
 [81]  -20.82640  -44.56221   29.48178  -36.29393  -27.62345  -19.42082   20.64037   34.02868  -23.36156   25.62215
 [91]   29.85049   32.55481   24.20885   49.16082  -26.37183   37.04111  -45.41183   21.82793  -50.02622   62.34357
[101]  -25.32576   20.09597   18.66912   30.72835   18.94273  -19.66469  -32.90512   26.22701   19.84833  -36.62468
[111]  -52.85687   34.25187  -54.11745 -103.98880   27.33916  -27.67264  -51.36413   55.52980   35.10406  -19.97174
[121]  -30.88165   49.25778   51.13870   40.40019  -19.50898   40.10841  -24.77038   43.33850  -20.56602  -27.04730
[131]  -20.17854  -21.11775  -38.39270   27.06720  -76.96214  -30.24300  -31.02700  -36.36435  -22.41562   25.94116
[141]  -53.68596  -22.40946   25.95597  -42.16744   26.24000   26.17844   31.77054   25.26237  -26.31242   44.78982
[151]   20.99781   35.41564  -27.10958  -22.57729   47.20786   58.80171   96.64313   51.30171   19.51763  -58.85591
[161]  -33.98554   25.99611   18.88266  -26.50747  -20.06062  -20.89451   43.99320  -26.44985   21.42720   18.98053
[171]   44.90294   47.96904   30.14470   19.09473  -47.82002  -22.95058   30.37665   27.32213   21.28048  -20.82503
[181]  -33.83381   22.10131   35.56539  -64.98884  -19.74075   21.42281  -23.72842  -21.14053  -47.64618   28.48533
[191]  -45.04602  -29.50158  -24.32170  -47.68350   68.63041   32.09418   30.44865  -40.85758  -34.18093  -26.42839
[201]  116.87622  -27.86539  -35.09574   19.41264  -20.83404   21.46554  -22.72767  -22.15871  -35.87566   21.39410
[211]   24.28066  -31.48805   38.82674   22.51006  -24.63762   39.83762  -19.97351   23.61458  -26.91618   37.56393
[221]   27.46909   29.21365  -84.93285   19.53304   28.37116   44.11599  -53.21926   37.54716   34.93574  101.27140
[231]   21.83436  -26.62685  -21.15319   23.21790  -23.88758  -47.40176   22.47292  -22.32888   19.57662   25.58856
[241]   22.86036  -19.28362   46.54271   40.95413   19.56476  -28.79192
```

Figure 8: Sound signal outliers

The outliers observed in the sound signal data are shown in Figure 8.

Code for above task is given in Appendix 1 and Appendix 1.1.a.

1.1. Time series plots

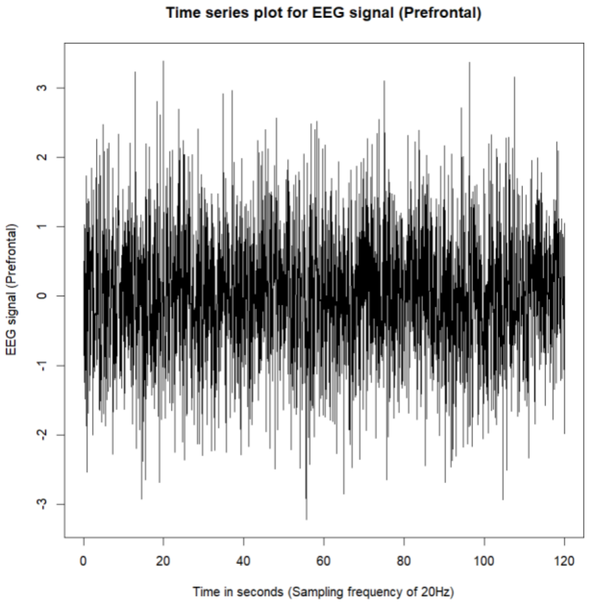

Figure 9: Time series plot for Prefrontal

Figure 9's plot of the prefrontal time series demonstrates that it has an erratic pattern and will include outliers. Although the data values range from about -2.5 to +2.5, the majority of the data is roughly centred between -1 and +1.

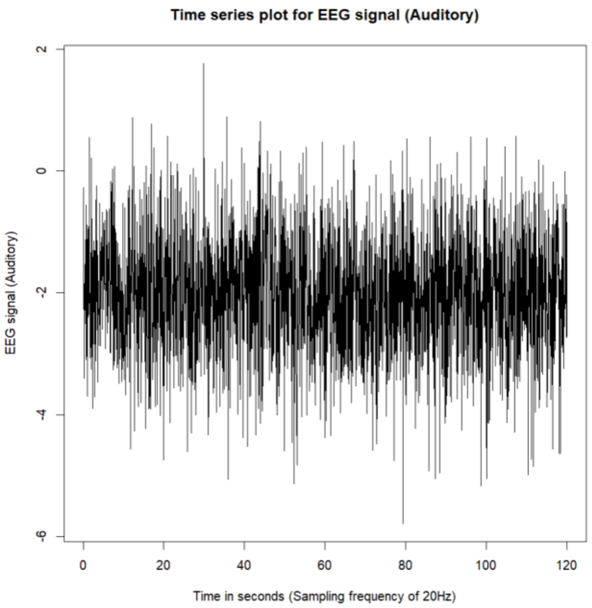

Figure 10: Time series plot for Auditory

The auditory time series plot in Figure 10 shows that it will contain outliers and has an irregular pattern. The majority of the data is broadly centred between -3 and -1, even though the data values range from about -4.5 to 0.

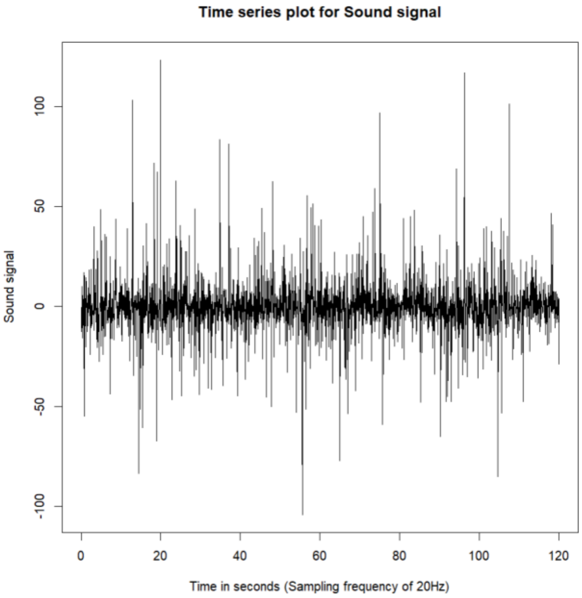

Figure 11: Time series plot for Sound signal

Figure 11's sound signal plot demonstrates the signal's irregular nature. Despite having so many outliers, the majority of the data is often centred around zero.

Code for above task is given in Appendix 1.1.

1.2. Distribution for each signal

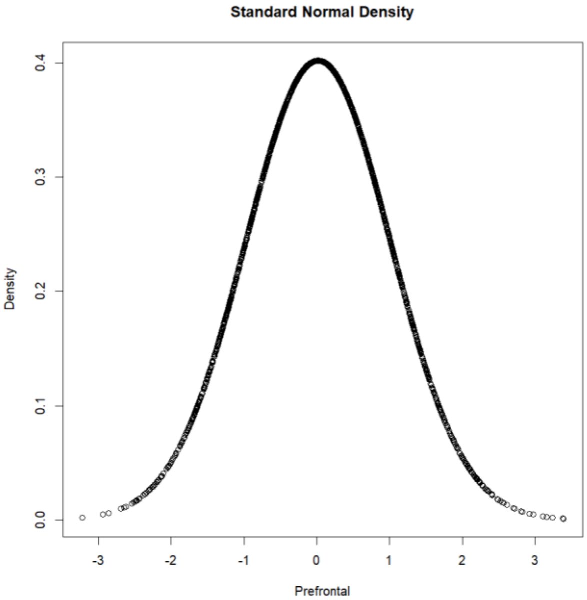

Figure 12: Prefrontal Density

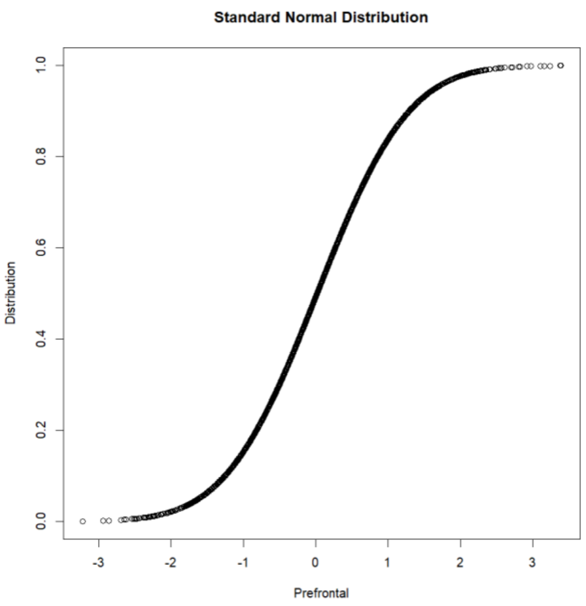

Figure 13: Prefrontal Distribution

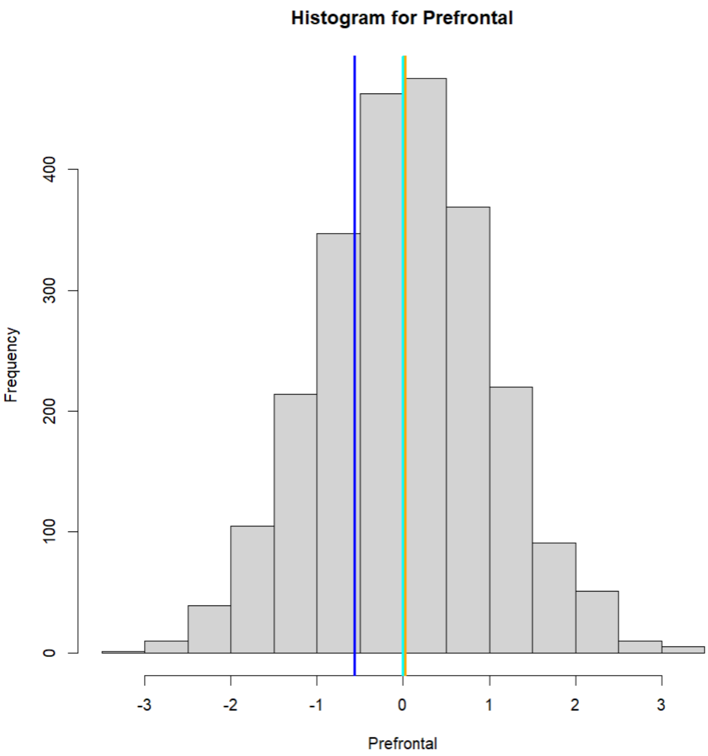

Figure 14: Prefrontal histogram

Figures 12 and 13 respectively display the density curve and cumulative density curve. There is no skewness in the density plot. The cyan line in the prefrontal histogram of figure 14 represents the mean value, the orange line the median value, and the blue line the mode value. Values of the mean and median are about equal. Since the histogram and density curve are approximately bell-shaped, the prefrontal data can be considered to have a normal distribution.

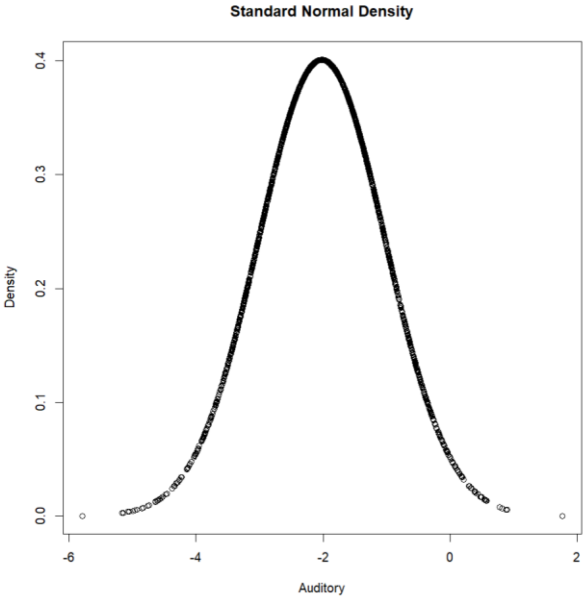

Figure 15: Auditory Density

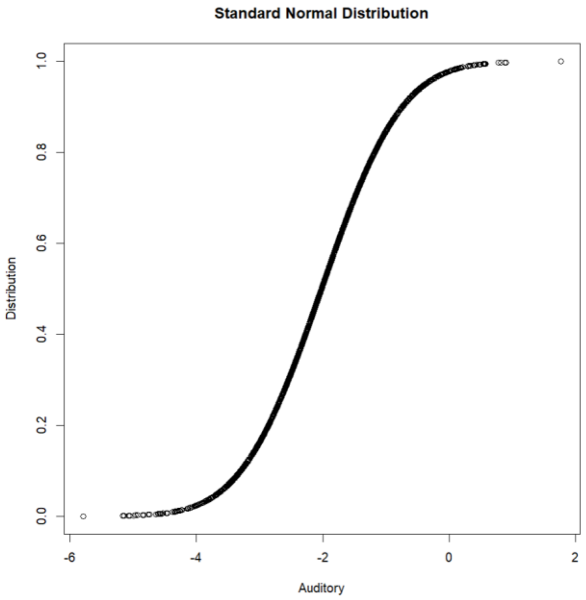

Figure 16: Auditory Distribution

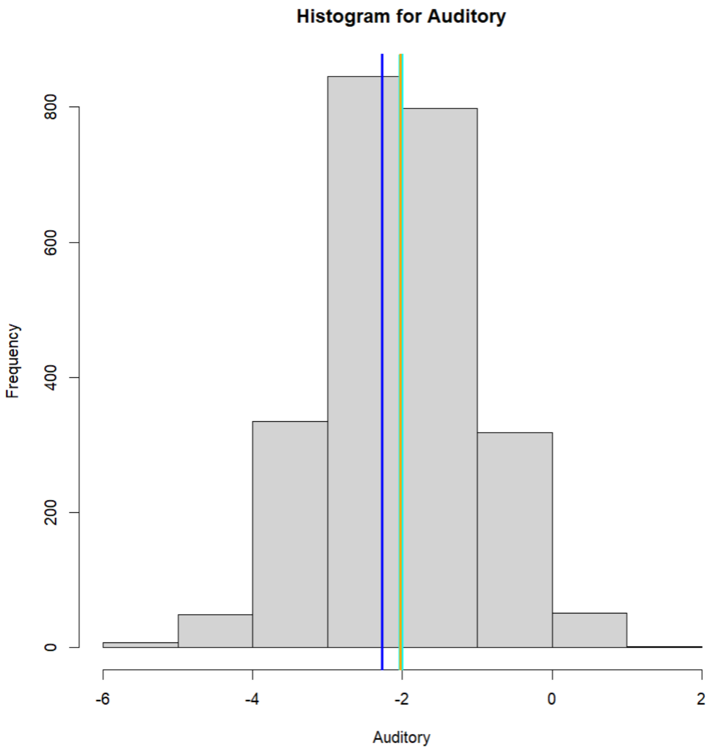

Figure 17: Auditory histogram

The density curve and cumulative density curve of auditory data are shown in Figures 15 and 16, respectively. The density plot is not skewed in any way. Figure 17's auditory histogram shows three lines: a cyan line for the mean value, an orange line for the median value, and a blue line for the mode value. The mean and median values are almost equal, hence the corresponding lines are nearly merged. The density curve and histogram, which are roughly bell-shaped, indicate that the auditory data has a normal distribution.

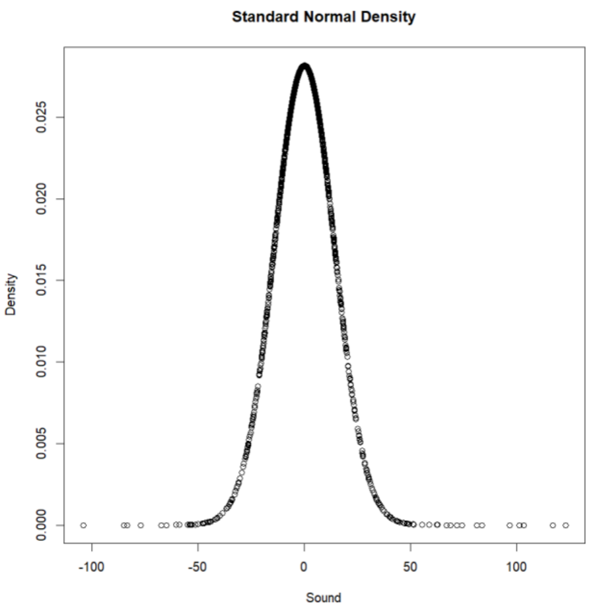

Figure 18: Sound signal Density

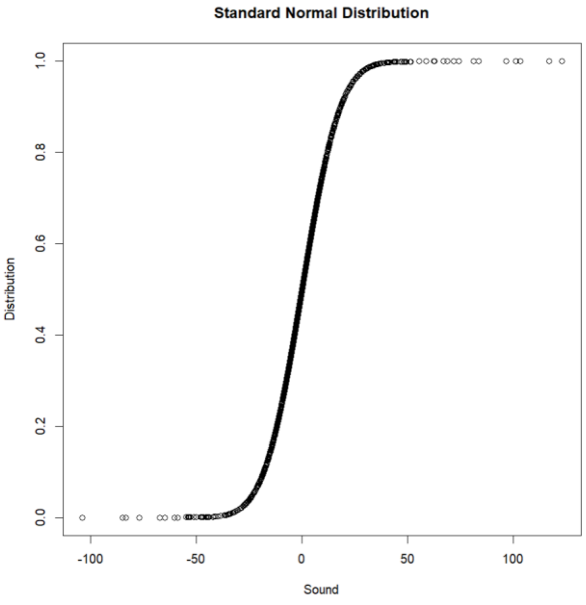

Figure 19: Sound signal distribution

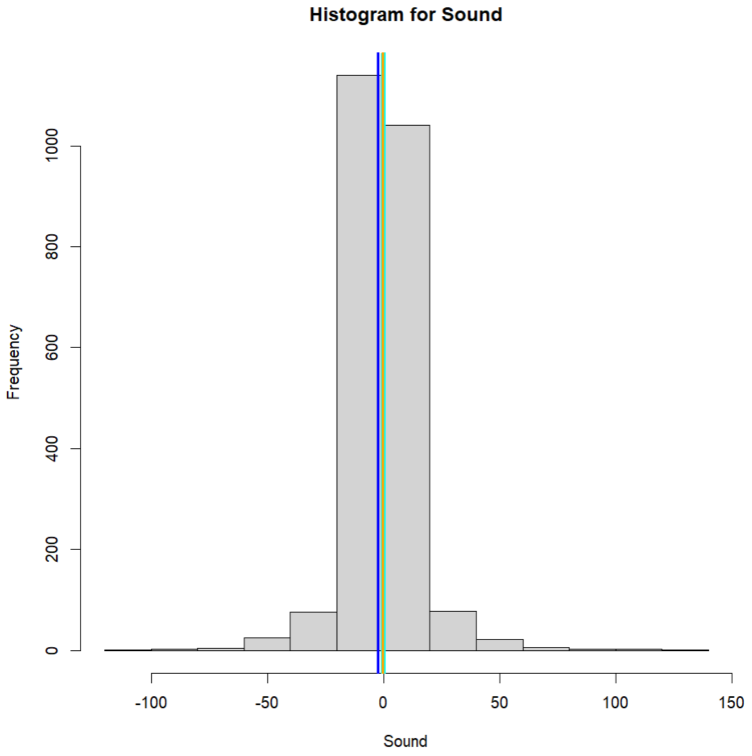

Figure 20: Sound signal histogram

Figures 18 and 19 respectively display the density curve and cumulative density curve of sound signal data. There is no skew in the density plot. The histogram of the sound signal in Figure 20 is represented by three lines: cyan for the mean value, orange for the median value, and blue for the mode value. The related lines are almost completely merged since the mean and median values are almost equal. The bell-shaped histogram and density curve show that the sound signal data has a normal distribution. And zero is the point where most of the values are located.

```
Mean:  0.01752658
Median:  0.02297666
Mode:  -0.5623198
Sample Maximum:  3.387346
Sample Minimum:  -3.221049
Sample Range:  6.608395Sample Variance:  0.9847206
Sample Standard Deviation:  0.9923309
Skewness: %s 0.03437
```

Figure 18. 1: Central tendencies of prefrontal

The summary statistics of prefrontal data are shown in figure 18.1.

```
Mean:  -2.019821
Median:  -2.02948
Mode:  -2.274491
Sample Maximum:  1.764783
Sample Minimum:  -5.78645
Sample Range:  7.551233Sample Variance:  0.9896497
Sample Standard Deviation:  0.9948114
Skewness: %s -0.01543933
```

Figure 19. 1: Central tendencies of auditory

The summary statistics of auditory data are displayed in figure 19.1.

```
Mean:  0.01876479
Median:  -0.3210923
Mode:  -2.386526
Sample Maximum:  122.9741
Sample Minimum:  -103.9888
Sample Range:  226.9629Sample Variance:  200.5665
Sample Standard Deviation:  14.16215
Skewness: %s 0.8473988
```

Figure 20. 1: Central tendencies for sound signal

The summary statistics of sound data are displayed in figure 20.1

Code for above task is given in Appendix 1.2.

1.3. Correlation and scatter plots

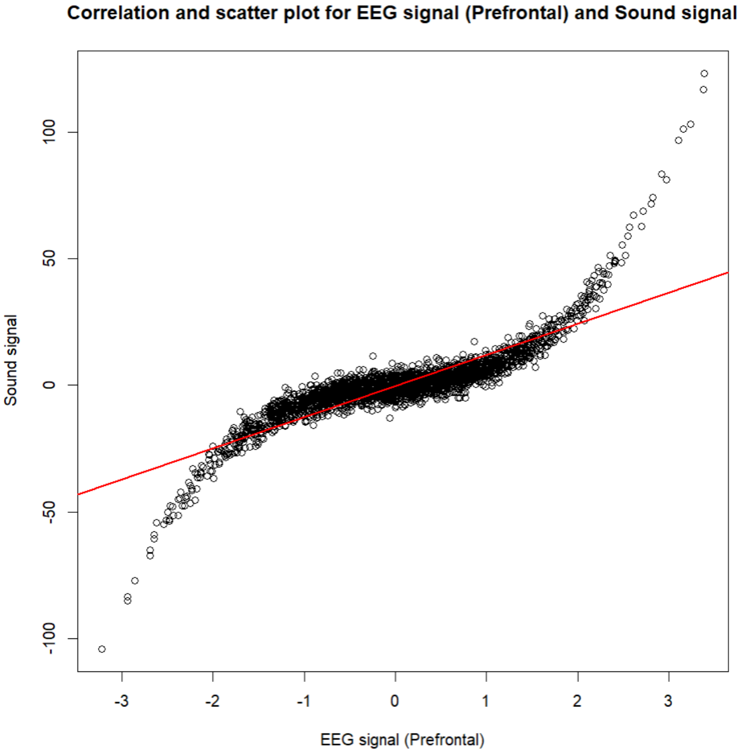

Figure 21: Correlation plot for prefrontal and sound signals

```
> cat("Correlation between EEG signal (Prefrontal) and Sound signal",cor(data_eeg[,"prefrontal"],data_sound))
Correlation between EEG signal (Prefrontal) and Sound signal 0.8625074
```

The relationship between prefrontal data and sound signal data is depicted in Figure 21. The correlation line is nearly diagonal and the data points almost closely overlap, giving the plot a shape. Code yields a correlation value of

0.8625074. Prefrontal data and sound signal have a positive correlation, it can be said.

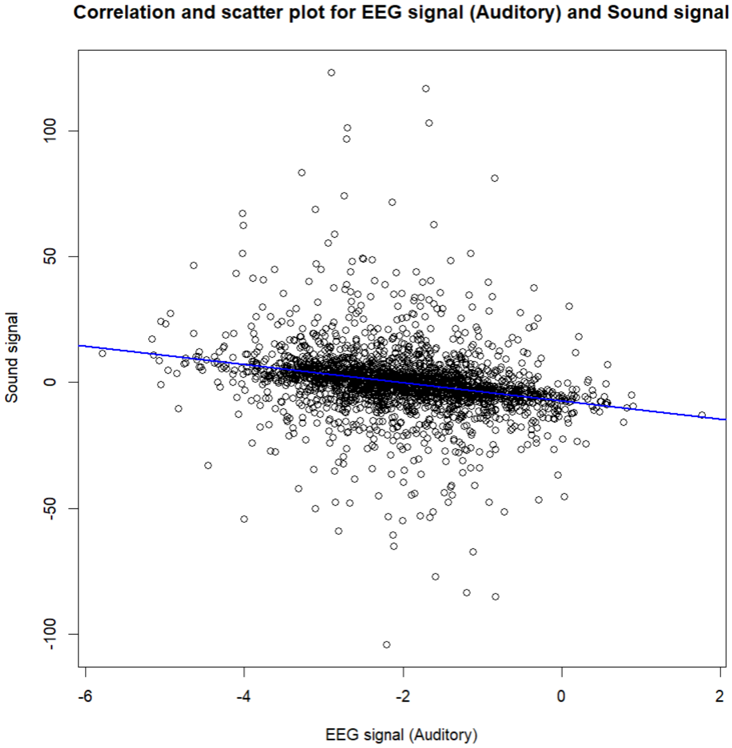

Figure 22: Correlation plot for auditory and sound signals

```
> cat("Correlation between EEG signal (Auditory) and Sound signal",cor(data_eeg[,"auditory"],data_sound))
Correlation between EEG signal (Auditory) and Sound signal -0.252638
>
```

Figure 22 shows the correlation between auditory data and sound signal data. The majority of the data points are not near to the regression line, and the correlation line is almost horizontal. A correlation value of -0.252638 is produced by code. It might be claimed that there is a weak negative correlation between auditory data and sound signal.

Code for above task is given in Appendix 1.3.

2. Regression (Modelling the relationship between audio and EEG signals)

```
No. of rows and columns in brain signal data: 2400 , 2
No. of rows and columns in sound signal data: 2400 , 1
```

Figure 23.1: Shape of EEG and sound signal data

Figure 23.1 shows that the shape of data for both sound and brain signals, both has 2400 rows. Sound signal data is the dependent variable, and prefrontal and auditory data are the independent variables in brain signal data.

Code for above task is given in Appendix 2.

2.1. Estimate model parameters

```
> cat("Model 1 parameter values for θ1, θ2 and θbias:", model1_theta_hat)
Model 1 parameter values for θ1, θ2 and θbias: 3.59671 -0.005881191 -1.149622
> cat("Model 2 parameter values for θ1, θ2 and θbias:", model2_theta_hat)
Model 2 parameter values for θ1, θ2 and θbias: 0.2743206 0.783168 -4.751963
> cat("Model 3 parameter values for θ1, θ2, θ3 and θbias:", model3_theta_hat)
Model 3 parameter values for θ1, θ2, θ3 and θbias: 2.715713 -3.15135 4.18139 -6.6514
>
> cat("Model 4 parameter values for θ1, θ2, θ3, θ4 and θbias:", model4_theta_hat)
Model 4 parameter values for θ1, θ2, θ3, θ4 and θbias: 4.171363 0.0594178 2.719001 -0.1494208 -2.474041
>
> cat("Model 5 parameter values for θ1, θ2, θ3 and θbias:", model5_theta_hat)
Model 5 parameter values for θ1, θ2, θ3 and θbias: 3.602873 -0.03954466 -3.154125 -6.54396
>
```

Figure 23: Model parameter results

Figure 23 shows the calculated model parameters (theta 1, theta 2, theta 3, theta4 and theta_bias) for models 1,2,3,4 and 5.

Code for above task is given in Appendix 2.1.1, Appendix 2.1.2, Appendix 2.1.3, Appendix 2.1.4, Appendix 2.1.5.

2.2. Compute RSS

```
> cat("Model 1 residual sum of squared errors:",model1_rss)
Model 1 residual sum of squared errors: 30738.22
> cat("Model 2 residual sum of squared errors:",model2_rss)
Model 2 residual sum of squared errors: 438230.4
>
> cat("Model 3 residual sum of squared errors:",model3_rss)
Model 3 residual sum of squared errors: 1525.621
> cat("Model 4 residual sum of squared errors:",model4_rss)
Model 4 residual sum of squared errors: 7949.273
>
> cat("Model 5 residual sum of squared errors:",model5_rss)
Model 5 residual sum of squared errors: 17138.1
```

Figure 24: RSS results for all models

The estimated residual sum of squared errors for models 1, 2, 3, 4, and 5 are displayed in Figure 24. The model with the lowest RSS value is the best model. Model 3 has the lowest rss value in this case.

Code for above task is given in Appendix 2.2.1, Appendix 2.2.2, Appendix 2.2.3, Appendix 2.2.4, Appendix 2.2.5.

2.3. Compute log-likelihood function

```
> cat("Model 1 log-likelihood: ",model1_log_liklhd)
Model 1 log-likelihood:  -6465.498
> cat("Model 2 log-likelihood: ",model2_log_liklhd)
Model 2 log-likelihood:  -9654.184
>
> cat("Model 3 log-likelihood: ",model3_log_liklhd)
Model 3 log-likelihood:  -2861.772
>
> cat("Model 4 log-likelihood: ",model4_log_liklhd)
Model 4 log-likelihood:  -4842.587
>
> cat("Model 5 log-likelihood: ",model5_log_liklhd)
Model 5 log-likelihood:  -5764.455
>
```

Figure 25: Log-likelihood results

Figure 25 shows the calculated log-likelihood results for models 1,2,3,4 and 5. A technique to gauge a regression model's goodness of fit is to look at its log-likelihood value. The better the model matches the dataset, the higher the log-likelihood value. Here, model 3 has the higher log-likelihood value.

Code for above task is given in Appendix 2.3.1, Appendix 2.3.2, Appendix 2.3.3, Appendix 2.3.4, Appendix 2.3.5.

2.4. Compute Akaike information criterion (AIC) and Bayesian information criterion (BIC)

```
> cat("Model 1 Akaike information criterion: ",model1_aic)
Model 1 Akaike information criterion:  12937
> cat("Model 1 Bayesian information criterion: ",model1_bic)
Model 1 Bayesian information criterion:  12954.35
>
> cat("Model 2 Akaike information criterion: ",model2_aic)
Model 2 Akaike information criterion:  19314.37
>
> cat("Model 2 Bayesian information criterion: ",model2_bic)
Model 2 Bayesian information criterion:  19331.72
> cat("Model 3 Akaike information criterion: ",model3_aic)
Model 3 Akaike information criterion:  5731.544
>
> cat("Model 3 Bayesian information criterion: ",model3_bic)
Model 3 Bayesian information criterion:  5754.677
>
> cat("Model 4 Akaike information criterion: ",model4_aic)
Model 4 Akaike information criterion:  9695.173
>
> cat("Model 4 Bayesian information criterion: ",model4_bic)
Model 4 Bayesian information criterion:  9724.089
>
```

```
> cat("Model 5 Akaike information criterion: ",model5_aic)
Model 5 Akaike information criterion:  11536.91
> 
> cat("Model 5 Bayesian information criterion: ",model5_bic)
Model 5 Bayesian information criterion:  11560.04
> 
```

Figure 26: AIC and BIC results

Figure 26 shows the calculated AIC and BIC results for models 1,2,3,4 and 5. The number of fitted parameters is taken into consideration when calculating the goodness of fit using the AIC (Akaike information criterion). We often choose the model with the lowest BIC value because models with low test errors tend to have BIC values that are small. A small value for AIC and BIC denotes a model with a low test error. Here model 3 has the lowest AIC and BIC values.

Code for above task is given in Appendix 2.4.1, Appendix 2.4.2, Appendix 2.4.3, Appendix 2.4.4, Appendix 2.4.5.

2.5. Plot and evaluate the error distributions

The error distributions are plotted with QQ-plot and histograms. The Q-Q plot should be a straight line through the origin with slope 1 if the sample data is taken from the normal distribution. The data is thought to be regularly distributed if the q-q plot's points generally follow a straight diagonal line. The data is assumed to be normally distributed if the histogram resembles a bell shape**.**

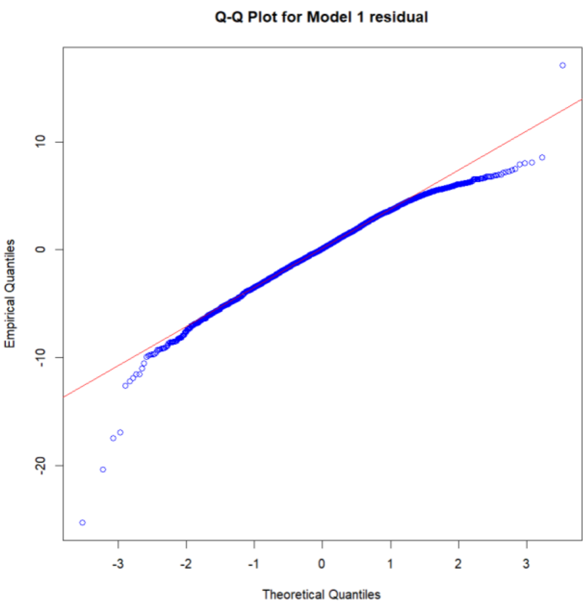

Figure 27: Q-Q plot for model 1

The points don't entirely follow the straight diagonal line, as can be seen from the Q-Q plot for model 1 in Figure 27.

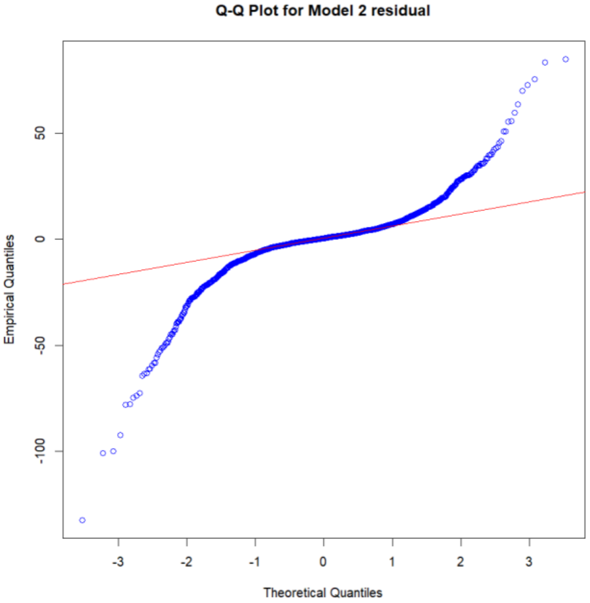

Figure 28: Q-Q plot for model 2

As seen by the Q-Q plot for model 2 in Figure 28, the points do not at all follow the straight diagonal line.

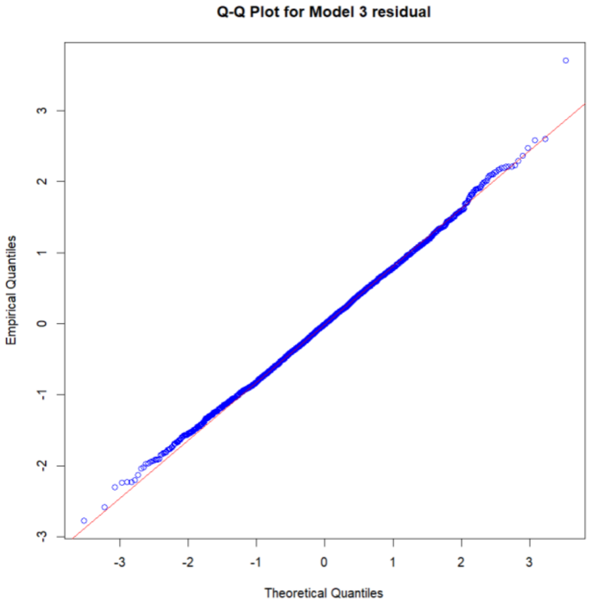

Figure 29: Q-Q plot for model 3

Nearly all of the points follow the straight diagonal line, as seen by the Q-Q plot for model 3 in Figure 29.

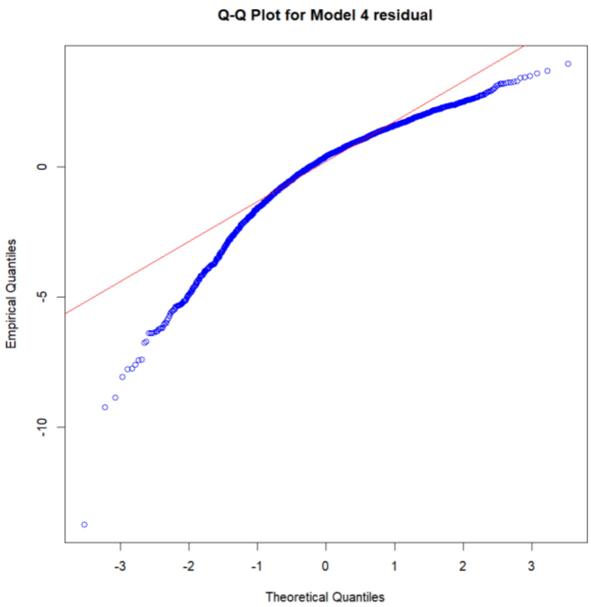

Figure 30: Q-Q plot for model 4

The points do not at all follow the straight diagonal line, as can be seen from the Q-Q plot for model 4 in Figure 30.

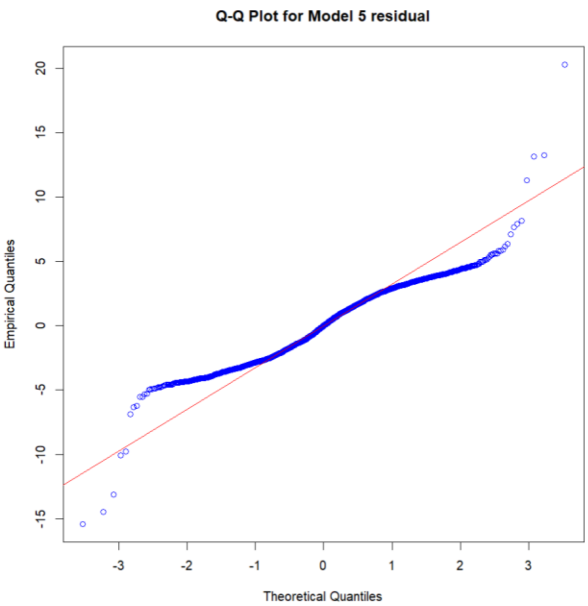

Figure 31: Q-Q plot for model 5

The Q-Q plot for model 5 in Figure 31 demonstrates that the points do not at all follow the straight diagonal line.

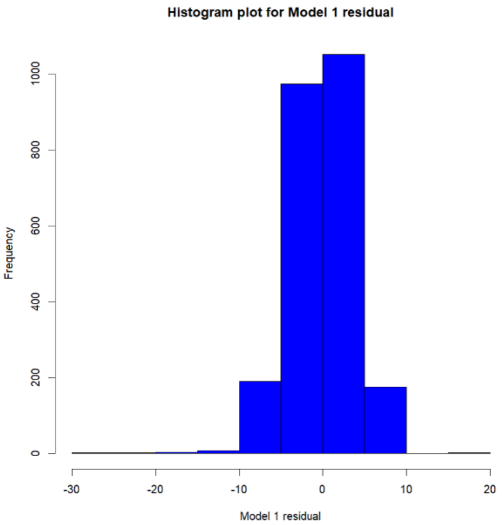

Figure 32: Histogram plot for model 1 residual

The model 1 residual histogram in figure 32 displays an uneven distribution of data and a squashed bell shape.

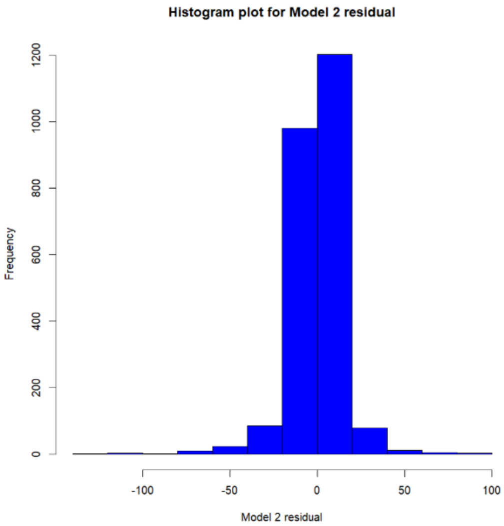

Figure 33: Histogram plot for model 2 residual

The model 2 residual histogram in figure 33 exhibits a squashed bell shape and an unbalanced data distribution.

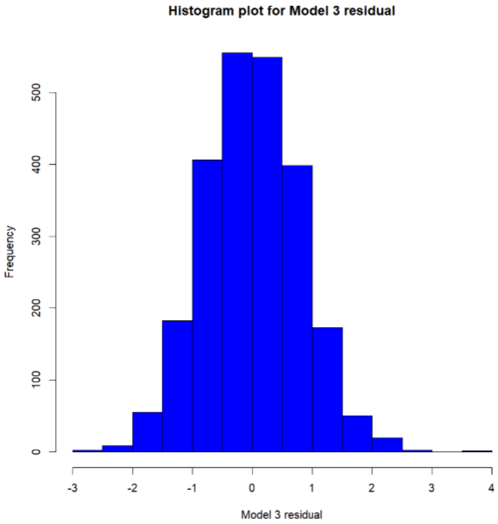

Figure 34: Histogram plot for model 3 residual

The model 3 residual histogram in figure 34 exhibits a perfect bell shape and an equal distribution of data.

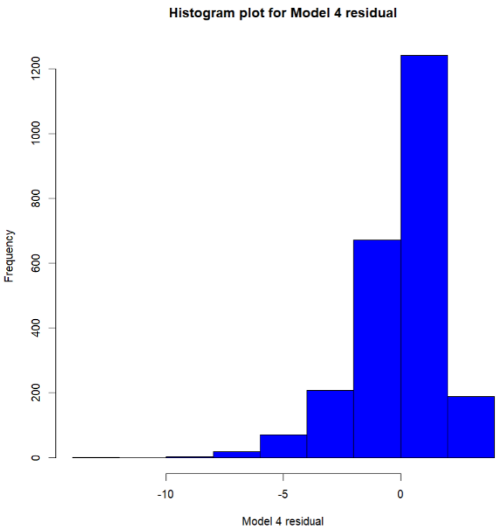

Figure 35: Histogram plot for model 4 residual

Figure 35's model 4 residual histogram reveals an asymmetric data distribution and a right skew.

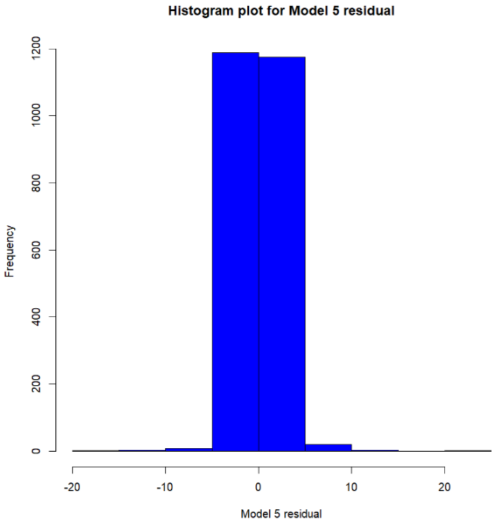

Figure 36: Histogram plot for model 5 residual

Figure 36's model 5 residual histogram shows a narrow bell shape and an asymmetric data distribution.

Analysing all models's Q-Q plot and histogram, model 3 seems to be the best model.

```
> shapiro.test(model1_error)

        Shapiro-Wilk normality test

data:  model1_error
W = 0.97979, p-value < 2.2e-16
```

```
> shapiro.test(model2_error)

        Shapiro-Wilk normality test

data:  model2_error
W = 0.797, p-value < 2.2e-16
```

```
> shapiro.test(model3_error)

        Shapiro-Wilk normality test

data:  model3_error
W = 0.99877, p-value = 0.08227
```

```
> shapiro.test(model4_error)

        Shapiro-Wilk normality test

data:  model4_error
W = 0.91342, p-value < 2.2e-16
```

```
> shapiro.test(model5_error)

        Shapiro-Wilk normality test

data:  model5_error
W = 0.96098, p-value < 2.2e-16
```

Figure 37: Shapiro-wilk test on all models

If the p-value is greater than alpha=0.05, then the data is presumed to be normally distributed in the formal statistical test, the Shapiro-Wilk test. Here model 3 error has p-value greater than 0.05.

```
> ks.test(model1_error,"pnorm")

        Asymptotic one-sample Kolmogorov-Smirnov test

data:  model1_error
D = 0.28929, p-value < 2.2e-16
alternative hypothesis: two-sided
```

```
> ks.test(model2_error,"pnorm")

        Asymptotic one-sample Kolmogorov-Smirnov test

data:  model2_error
D = 0.36561, p-value < 2.2e-16
alternative hypothesis: two-sided
```

```
> ks.test(model3_error,"pnorm")

        Asymptotic one-sample Kolmogorov-Smirnov test

data:  model3_error
D = 0.059427, p-value = 8.691e-08
alternative hypothesis: two-sided
```

```
> ks.test(model4_error,"pnorm")

        Asymptotic one-sample Kolmogorov-Smirnov test

data:  model4_error
D = 0.17241, p-value < 2.2e-16
alternative hypothesis: two-sided
```

```
> ks.test(model5_error,"pnorm")

        Asymptotic one-sample Kolmogorov-Smirnov test

data:  model5_error
D = 0.26879, p-value < 2.2e-16
alternative hypothesis: two-sided
```

Figure 38: Kolmogorov-Smirnov test for all models

If the p-value is greater than alpha=0.05, then the data is presumed to be regularly distributed in the formal statistical test, the Kolmogorov-Smirnov test. Here no model error has that such value.

Code for above task is given in Appendix 2.5.1, Appendix 2.5.2, Appendix 2.5.3, Appendix 2.5.4, Appendix 2.5.5.

2.6. Select best regression model

Model 3 is the best performing model among the five given models based on the results.

Model 3: $y = \mathbf{2.715713*}x_{1}^{3}+\mathbf{-3.15135*}x_{2}+\mathbf{4.18139*}x_{1}+\mathbf{-6.6514}+\varepsilon$

	Θ1	Θ2	Θ3	Θ4	Θbias
Model 1	3.59671	-0.005881191			-1.149622
Model 2	0.2743206	0.783168			-4.751963
Model 3	**2.715713**	**-3.15135**	**4.18139**		**-6.6514**
Model 4	4.171363	0.0594178	2.719001	-0.1494208	-2.474041
Model 5	3.602873	-0.03954466	-3.154125		-6.54396

Figure 2.6.1: Theta values for all models

Figure 2.6.1 displays the theta hat values for models.

	Number of estimated parameters (k)	Residual sum of squared errors	Log-likelihood	Akaike information criterion	Bayesian information criterion
Model 1	3	30738.22	-6465.498	12937	12954.35
Model 2	3	438230.4	-9654.184	19314.37	19331.72
Model 3	**4**	**1525.621**	**-2861.772**	**5731.544**	**5754.677**
Model 4	5	7949.273	-4842.587	9695.173	9724.089
Model 5	4	17138.1	-5764.455	11536.91	11560.04

Figure 2.6.2: Summary of all model's error test values

From figure 2.6.2, it can be concluded that model 3 is the best,

1. Model 3 has the lowest rss value.
2. Model 3 has the higher log-likelihood value.
3. Model 3 has the lowest AIC and BIC values.

From the above explained Q-Q plots and residual histogram plots, model 3 seems to be the best model.

From above explained Shapiro-Wilk test, model 3 error has p-value greater than 0.05, which seems to be good.

Thus, model 3 is selected as the best regression model.

2.7. Modelling with train-test data

```
> cat("No. of rows and columns in original dataset: ",dim(data_eeg_sound))
No. of rows and columns in original dataset:  2400 4
> cat("No. of rows and columns in train dataset: ",dim(data_train))
No. of rows and columns in train dataset:  1680 4
> cat("No. of rows and columns in test dataset: ",dim(data_test))
No. of rows and columns in test dataset:  720 4
```

Figure 39: Shape of original data, train data and test data

Figure 39 describes the shape of original dataset, training dataset and test dataset.

Code for above task is given in Appendix 2.7.

2.7.1. Estimate model parameters using training and test dataset

```
> cat("Best Model (train data) parameter values for θ1, θ2, θ3 and θbias:", modelBest_theta_hat)
Best Model (train data) parameter values for θ1, θ2, θ3 and θbias: 2.714109 -3.153057 4.195578 -6.661907
```

Figure 40: Model Parameter values for train data

The values of the theta hat parameters for the data model utilising the training dataset are explained in Figure 40.

```
> cat("Best Model (Test data) residual sum of squared errors:",modelBest_rss)
Best Model (Test data) residual sum of squared errors: 412.3641
>
```

Figure 41: RSS for test data

Figure 41, describes the RSS for the model tested with test data, which has the value 412.3641.

Code for above task is given in Appendix 2.7.1.

2.7.2. Compute model's prediction on the testing data

```
> cat("Predicted Values:", modelBest_y_hat)
Predicted Values: 0.3867918 -0.09730488 9.009996 2.691973 3.550689 14.32924 -1.210821 1.270523 1.168911 -14.75616 3.099311 -9.116692 6.058896 0.2988404 -0.114327
-5.097131 18.40509 3.252658 2.004174 -12.79986 -4.317284 0.5584268 -1.192349 -16.22804 -7.707421 23.82333 1.608321 2.087324 -0.6560144 -3.13082 -12.67729 -1.9925
89 -2.419447 -14.85293 -1.177215 -5.56838 0.7411255 24.71526 -5.688283 -16.74591 2.817524 3.028729 -1.96925 -6.690494 -3.073322 -0.6015128 5.375087 3.079646 -2.4
83149 3.50349 -6.214424 0.2500893 -2.566382 -3.271895 2.832852 6.322641 4.500817 4.255293 6.886277 0.3238207 -1.422711 5.197983 2.780894 6.14575 -4.452442 1.0512
39 12.19131 -5.174166 10.24366 -10.30504 -26.76444 -3.588148 -2.703367 1.743152 -0.6852071 -34.80421 -15.87159 5.581833 -1.445373 -5.164139 -2.69855 -1.280126 6.
181895 -5.309771 1.792464 8.845679 -51.09294 -15.57751 2.036889 -4.899569 -8.577388 -0.9875141 22.12587 -2.554511 3.51792 -9.254337 11.97325 4.36714 -11.06008 -1
1.0326 -5.016232 5.859094 -5.297122 -0.6502126 -19.29264 -1.497615 -67.29894 -2.824682 -2.785164 -4.297146 7.621005 -3.623488 -12.56655 -9.475116 18.56311 4.8105
29 -9.533887 -10.45675 -18.6285 19.88079 6.334503 4.479978 0.06767234 -14.5796 -12.47222 7.826404 -11.80588 0.1431779 1.605321 2.971408 7.704587 0.3114635 5.5240
79 0.1302674 5.63139 -2.474508 40.8479 -4.464011 1.719789 2.447703 2.021459 -7.29266 9.400794 3.543996 11.42109 7.897057 -5.509047 1.120725 -14.62911 36.0179 1.0
9897 -1.531764 -4.284375 2.343992 -5.978013 -13.14321 4.257858 0.1228156 -4.127446 -9.835001 15.28448 -1.09466 9.19695 3.212339 0.9625955 -3.502462 3.474325 -5.8
59137 -4.325453 -4.194117 17.45965 4.1869 -9.1588 13.37754 0.8777702 0.6657088 -4.166025 -0.1085866 6.971077 -41.98806 -0.5877026 5.555093 5.846369 -1.279345 -6.
061091 -1.47069 -8.65001 -3.133906 3.060561 -4.032775 0.781423 -6.241624 1.435034 2.959876 1.479272 4.397193 -1.040876 83.29365 0.7738864 23.26618 8.401628 -6.96
4513 16.04681 19.79704 -40.7285 -10.31277 -0.4172475 -4.27851 -4.657102 -7.765788 -7.553224 -5.010477 4.100037 -1.972891 -1.793547 1.785696 -17.94567 6.457158 -
0.4612687 -12.96441 5.571688 -7.109263 -19.65685 -8.621101 -2.503487 7.23146 8.052575 -10.43347 7.918003 -1.704194 -1.780051 7.04394 0.5529044 -17.5853 -0.333698
8 13.5902 -7.159607 -0.3111028 2.968297 -2.693738 3.818551 6.87341 -0.5480253 -2.087805 6.289282 6.750962 -9.427754 -22.83115 2.035569 0.9245789 -11.87168 9.1719
16 -0.07510302 25.21811 1.169288 -1.028925 0.10497 11.78031 -8.541677 48.99061 15.48748 -10.51069 -1.682197 -4.667372 3.557466 0.2588686 -3.382755 -7.342042 18.3
989 -1.404793 -4.45104 -8.398899 -9.666153 2.987856 -0.876763 3.24064 -2.581213 62.75935 -5.287573 -7.271643 1.383072 -6.370175 17.73695 6.034245 4.741959 -0.618
8403 -6.969383 2.961949 3.022748 -7.647923 0.9537588 5.53435 -1.303114 -1.574392 -2.158038 3.8705 -3.482947 -1.904209 3.348407 6.924999 1.063345 18.27281 3.49536
10.54175 -2.029189 0.348793 -2.199629 -1.150807 11.08119 5.380494 -6.356044 -16.61664 2.613577 7.787817 -3.971725 -5.204657 -2.276291 -1.150681 -0.9177577 -3.654
408 -19.72777 -9.321445 -1.517244 -3.391175 -2.466533 2.212156 -53.62019 -1.908063 -2.59577 -6.730012 -3.729509 1.126203 -26.77298 5.102354 34.33468 0.207709 -1
5.10505 -2.249683 -8.114322 8.086792 51.26483 4.32236 -1.019891 -7.404883 -3.162281 -0.6102109 -4.162127 1.146055 2.117043 -3.554881 -0.8353431 -3.543105 -19.547
92 -11.05713 -3.335127 -4.577203 3.705873 -20.82563 -6.404254 -4.956068 0.376937 7.980649 2.335333 -15.15429 1.616544 8.573482 9.943456 1.915431 17.78936 -10.735
6 1.290879 2.09923 -5.72754 -2.34282 1.853549 0.178443 15.60698 2.681698 -8.568415 -16.84796 4.678861 18.58699 27.03838 -8.535674 -1.433839 -4.44092 -10.33685 1
2.15978 -3.742734 -1.442108 -3.434159 -2.025251 2.876068 -2.61781 -0.03932749 3.228757 -2.981839 3.397143 -21.67596 6.842844 16.945 -7.811871 1.577409 -22.15829
-0.08453036 -3.59247 -3.961596 1.114518 -3.759394 -11.92669 17.03564 -8.173255 -4.073634 2.662088 -0.8719026 -2.359589 -18.57093 25.0634 -1.410353 6.521739 4.677
257 14.12555 2.050653 0.4003367 0.5750491 -2.998131 1.786845 3.188991 10.07712 -4.547823 -1.757388 -3.484108 7.624017 1.908267 11.58072 5.202341 -4.351024 6.4648
07 58.13912 -7.130803 -4.097976 -7.814237 3.765697 -2.075562 -7.913125 8.496281 0.6120283 96.35933 -3.283635 51.12767 2.538671 -4.512315 -0.8870319 6.831586 6.27
5423 25.97205 10.00646 0.5097718 -4.179674 1.112923 -0.3789681 -2.714336 -11.17298 2.004677 2.903149 -3.338257 2.496253 -6.52031 0.3011174 -7.619721 -20.49022 -
6.725452 -1.255147 2.7718 -15.0274 3.845071 3.119883 -8.514703 0.3205205 -6.146347 2.624707 0.7978685 -1.546956 4.722526 13.27899 3.352039 16.8154 -1.673784 -2.0
30561 -0.6133394 5.465561 -1.845603 6.84866 -7.746682 -2.809514 -17.4812 -0.8945346 30.99084 1.438177 -2.633476 16.96337 -4.934469 2.756065 0.2826814 8.982326 1
2.63971 -22.46949 30.22443 -2.61385 -7.160139 0.790546 4.947395 -4.356313 -5.841755 -0.5472113 -3.831248 3.297665 9.381338 6.520716 -4.49841 1.847678 -5.21497 -
0.7133341 9.089539 1.47531 -2.282131 1.071932 9.73114 8.56435 2.830194 -21.62897 -3.272544 4.268603 -4.827828 -11.27471 28.35535 -6.758586 -16.72288 -1.493177 -
0.2300464 2.683744 -1.296102 0.974328 -0.5437062 0.3811942 -3.805631 -16.37363 -6.17366 3.459906 68.71485 8.548901 -2.802819 4.467974 4.72703 -2.810517 18.86916
-3.721105 1.696196 -6.808084 -5.580163 0.5972259 3.085502 -1.745657 -7.994237 8.863954 -4.168713 -2.325718 -1.400677 9.783578 13.01882 -34.47719 -2.986499 12.441
21 5.012435 1.210273 2.560048 21.59866 2.732234 -0.1250063 3.662648 4.787118 -21.73269 -0.7651664 -8.553436 9.780905 1.497655 -11.04108 -4.888034 22.03599 -5.377
358 1.129222 0.101071 -16.3619 -4.689509 -17.1801 -4.077756 0.7950726 -2.639453 -1.455865 40.14039 1.189268 -3.639406 -5.462235 -14.82345 3.753108 0.2630377 -0.0
5982525 -20.14884 -3.695931 22.55403 37.32492 28.0546 -10.66903 3.054154 2.311315 30.91231 -9.463548 -0.8984634 -11.16131 19.29442 -2.411261 6.647955 -0.04490983
5.684741 -6.437356 -0.1667475 -2.317971 4.072194 -0.08824841 0.4481961 -4.202062 -0.2052206 -5.138227 35.30695 -8.277348 -4.953228 -7.194724 1.758481 -0.5571106
-5.341811 1.166122 5.200306 -2.798824 -26.60612 17.9364 1.000072 2.765637 -0.6742499 0.2649312 13.61441 3.998889 1.875016 -12.89008 5.435921 4.198553 -2.74735 -1
2.48557 9.525777 0.4385914 23.68345 -9.647253 -22.00542 -8.460014 3.751085 7.637489 8.516351 -4.365304 6.45964 17.86665 23.14928 -5.204883 -5.607955 -19.06345 -2
3.89651 12.0654 -7.341271 6.322486 2.407373 14.88081 8.32654 -5.045511 11.37127 21.9418 9.658205 1.411654 -0.991283 3.502793 0.2013013 5.479528 6.118119 5.541011
-1.063169 -2.570373 -0.4293743 -6.206931 1.501675 8.155151 -7.773128 3.970381 -0.1516209 -3.932805 2.496383 -1.791367 11.68726 -13.34824 -7.834999 2.267389 7.516
45 -3.15759 2.957051 -10.56473 9.061243 -0.07696135 -3.989373 6.065039 -1.175329 -0.3186741 -1.68514 -5.696187 -7.033413
>
```

Figure 42: Predicted values on test data

Figure 42 displays the predicted values for test data.

Code for above task is given in Appendix 2.7.2.

2.7.3. Compute 95% confidence intervals and plot them.

With a certain degree of certainty, confidence interval formula generates an interval with a lower bound and an upper bound that most likely contains a population parameter.

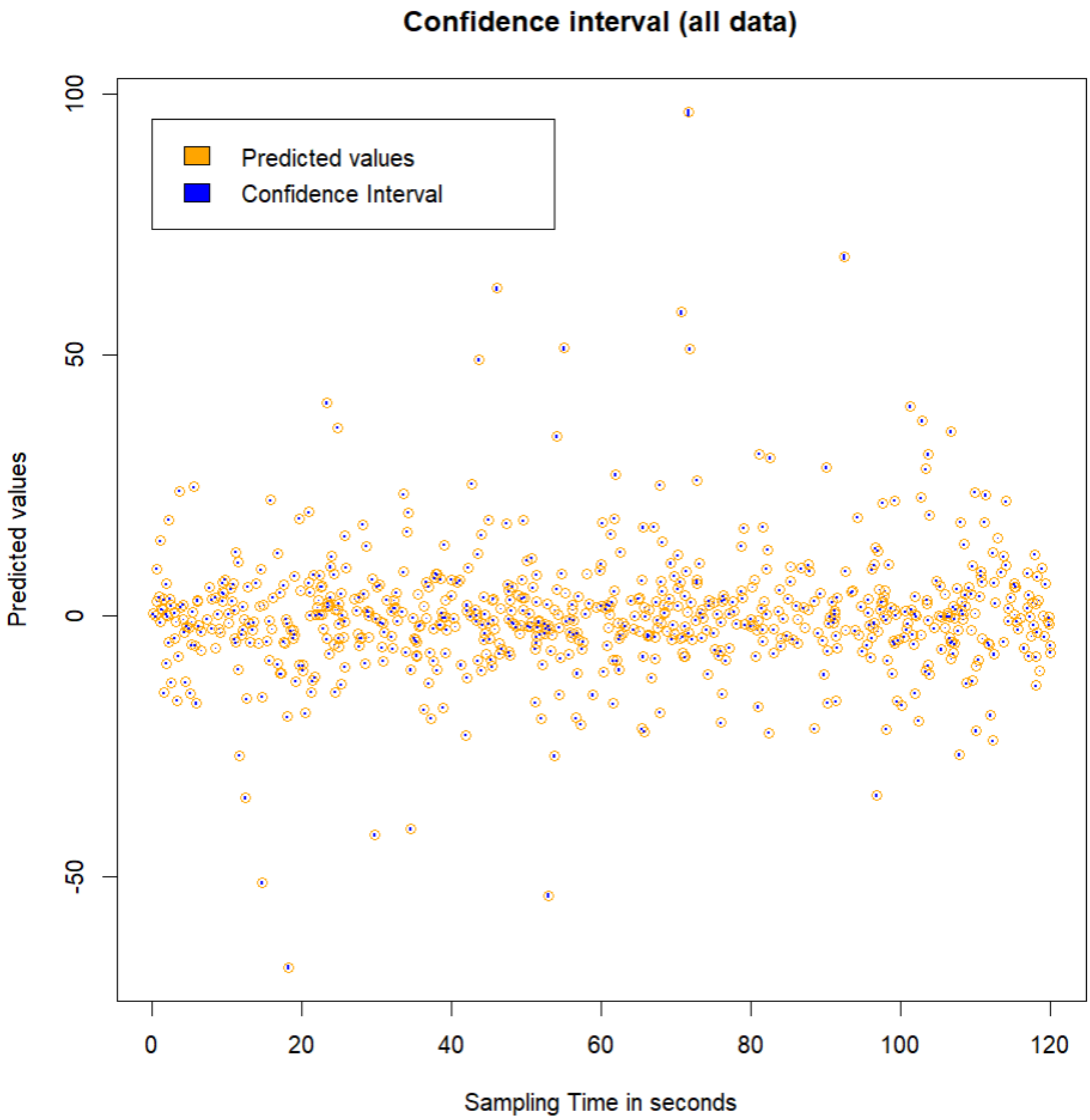

Figure 43: Confidence interval for all predicted values

Figure 43 describes the 95% confidence interval for all predicted values. Almost all points are near to confidence interval range.

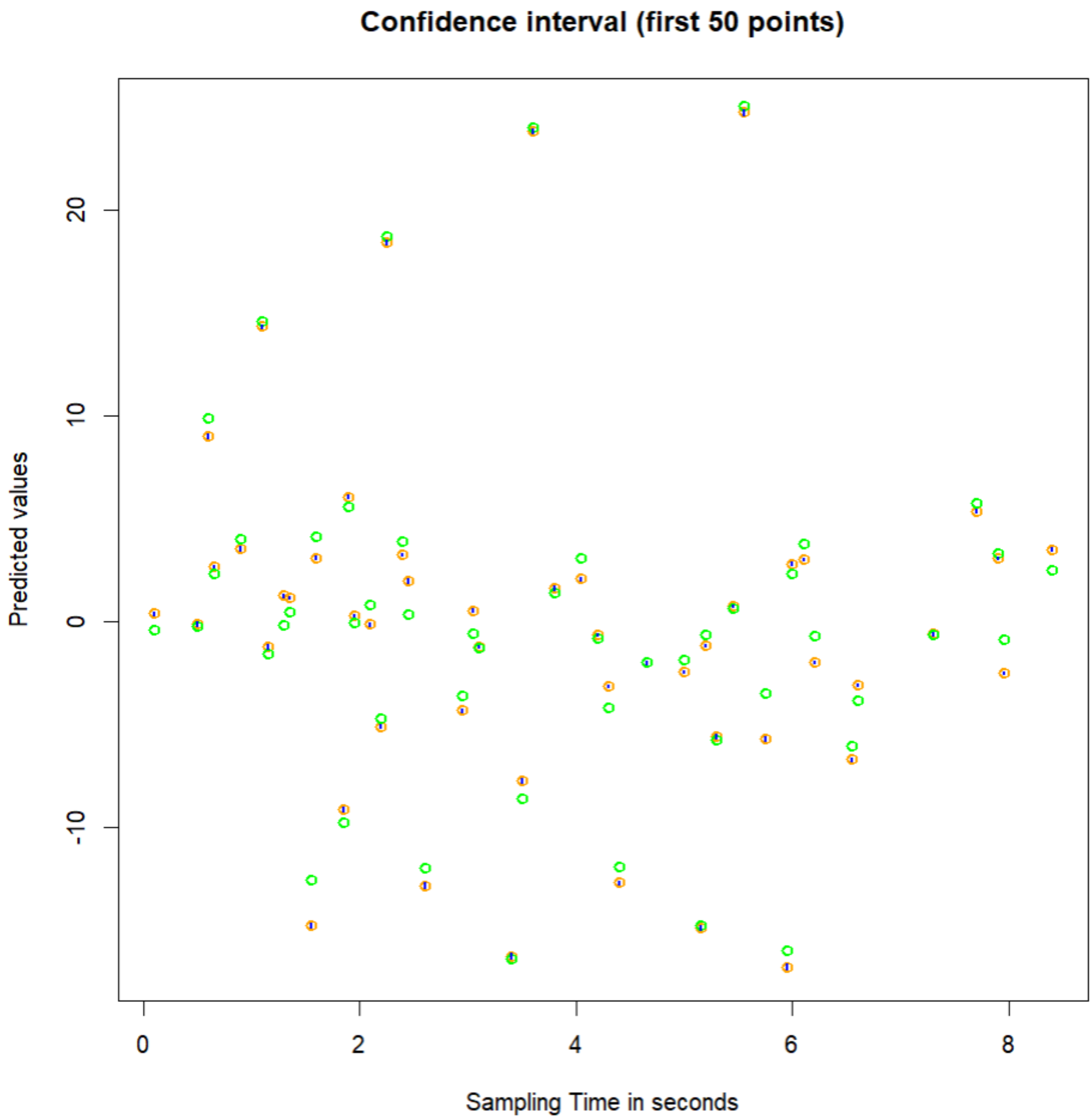

Figure 44: Confidence interval for first 50 predicted values

Figure 44 describes the 95% confidence interval for first 50 predicted values. Almost all points are near to confidence interval range.

Code for above task is given in Appendix 2.7.3.

3. Approximate Bayesian Computation (ABC) using rejection ABC
Estimating the posterior distributions of model parameters is possible using a class of computing techniques called approximate Bayesian computation (ABC), which has its roots in Bayesian statistics.
Model 3: y = 2.715713*x1^3+-3.15135*x2+4.18139*x1+-6.6514+ε

The two parameters with largest absolute values are 2.715713 and 4.18139

3.1. Compute two parameter posterior distribution

Code for above task is given in Appendix 3.

3.2. Create uniform distribution as prior

Code for above task is given in Appendix 3.

3.3. Using uniform prior, perform rejection ABC for the two parameters

Code for above task is given in Appendix 3.

3.4. Plot the joint and marginal posterior distribution

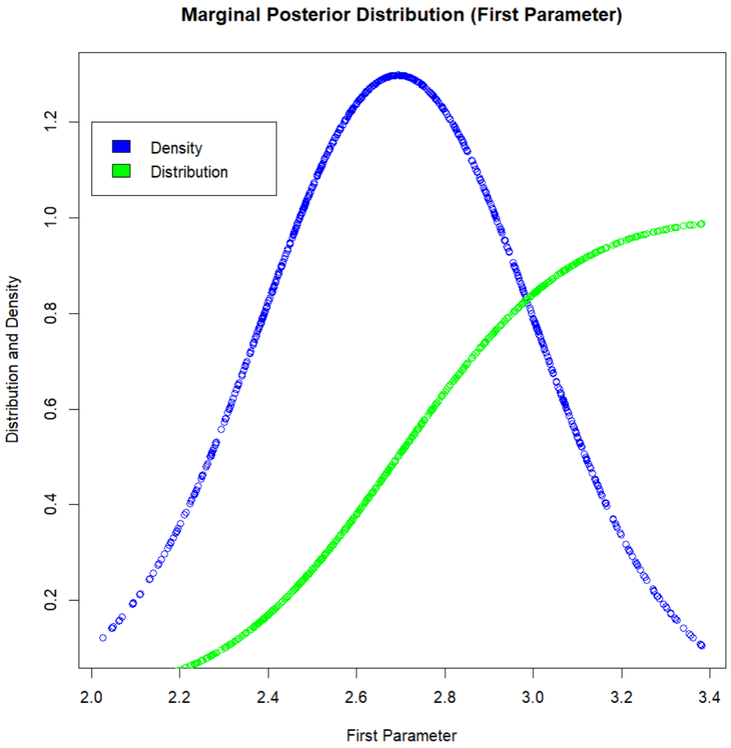

Figure 45: Marginal posterior distribution (first parameter)

For the marginal posterior distribution, density, and cumulative density plots for the first parameter are displayed in Figure 45. The density plot is nearly bell-shaped, but it leans slightly to the left.

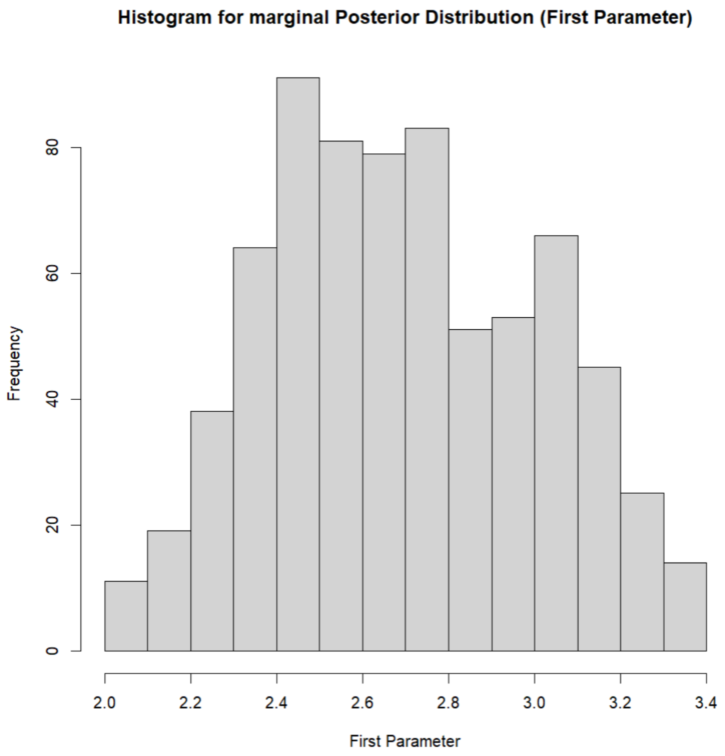

Figure 46: Histogram for marginal posterior distribution (first parameter)

Figure 46 shows the histogram for marginal posterior distribution, for the first parameter.

Marginal posterior distribution data for first parameter seems to be not normally distributed.

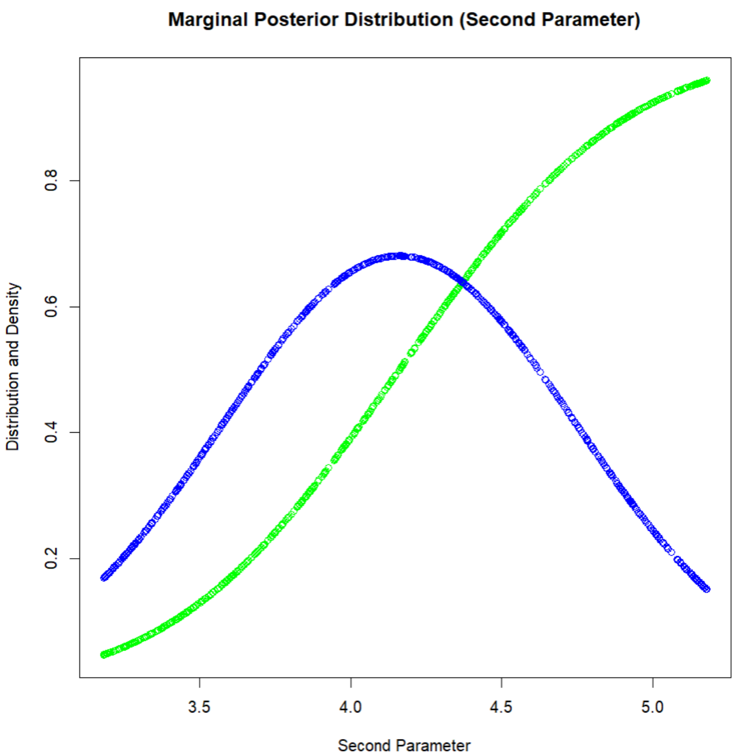

Figure 47: Marginal posterior distribution (second parameter)

For the marginal posterior distribution, density, and cumulative density plots for the second parameter are displayed in Figure 47. The density plot is flat bell-shaped.

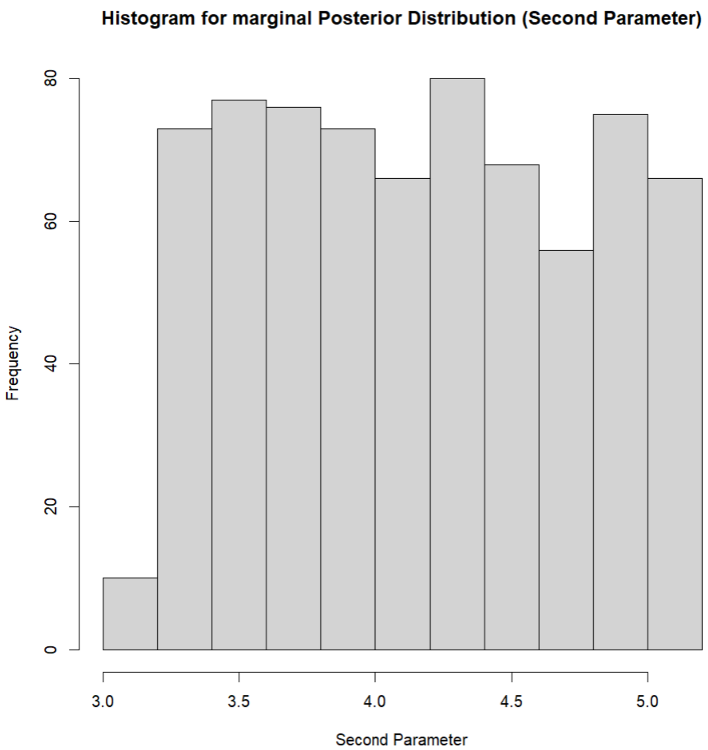

Figure 48: : Histogram for marginal posterior distribution (second parameter)

Figure 48 shows the histogram for marginal posterior distribution, for the second parameter.

Marginal posterior distribution data for second parameter seems to be not at all normally distributed.

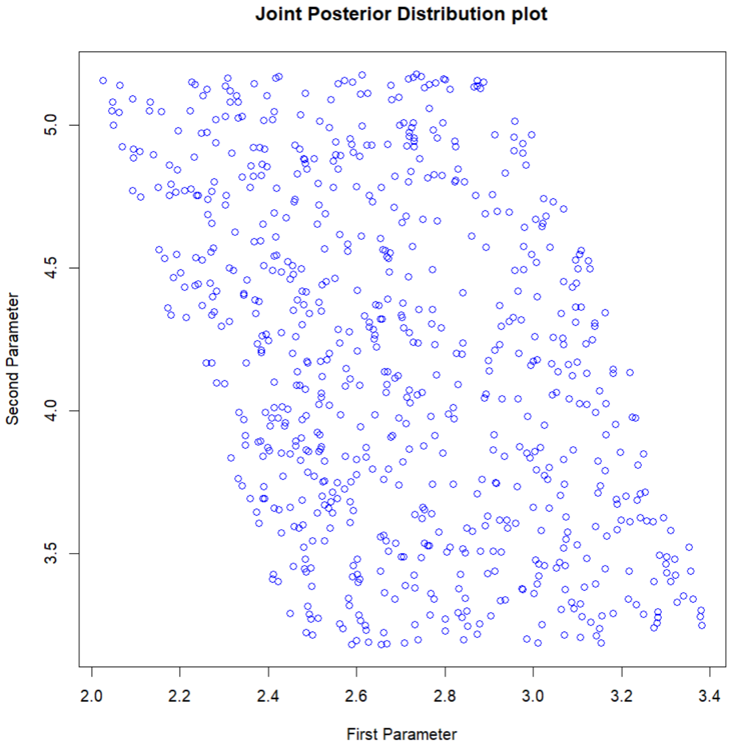

Figure 49: Joint posterior distribution

Figure 49 displays joint posterior distribution for first parameter and second parameter.

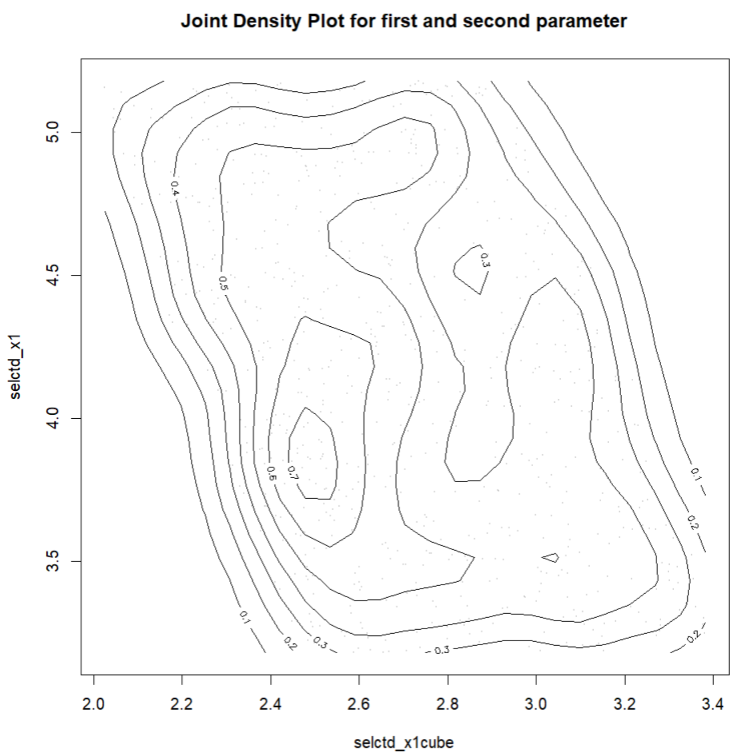

Figure 50: Joint posterior density plot

Figure 50 displays joint posterior density plot for first parameter and second parameter.

Code for above task is given in Appendix 3.

4. References

Lecture and lab notes by module leader: Dr. Fei He
Dalgaard, P. (2008). Introductory statistics with R (2nd ed.). New York: Springer.
James, G., Witten, Daniela, author, Hastie, Trevor, author, & Tibshirani, Robert, author. (2022). An introduction to statistical learning : With applications in R (Second ed., Springer texts in statistics).

https://www.geeksforgeeks.org/how-to-create-a-scatterplot-with-a-regression-line-in-r/

https://www.statology.org/test-for-normality-in-r/

https://www.statology.org/interpret-log-likelihood/

https://www.statology.org/train-test-split-r/

https://statisticsglobe.com/draw-plot-with-confidence-intervals-in-r

https://www.sciencedirect.com/science/article/pii/S175543651930026X https://towardsdatascience.com/the-abcs-of-approximate-bayesian-computation-bfe11b8ca341

https://www.sciencedirect.com/science/article/pii/S2215016120300698

https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1002803

https://www.rdocumentation.org/packages/LaplacesDemon/versions/16.1.6/topics/joint.density.plot

5. Appendix

Code for Task 1

Appendix 1

```
################ Read data ################
#X.csv file contains EEG signals.
#'x1' is prefrontal cortex and 'x2' is auditory cortex.
data_eeg = read.csv("X.csv")
data_eeg = data.matrix(data_eeg)
#Rename columns
colnames(data_eeg)=c("prefrontal","auditory")
#check if null values present
cat("Sum of null values in EEG signal data:",sum(is.na(data_eeg)))
print("First five rows in EEG signals file:")
print(data_eeg[1:5,])

#y.csv file contains sound signal 'y'.
data_sound = read.csv("y.csv")
data_sound = data.matrix(data_sound)
#Rename columns
colnames(data_sound)=c("sound_signal")
#check if null values present
cat("Sum of null values in Sound signal data:",sum(is.na(data_sound)))
print("First five rows in Sound signal file:")
print(data_sound[1:5,])

#time.csv contains sampling time of all three signals in 'seconds'.
#2 minutes of signal with sampling frequency of 20Hz.
data_time = read.csv("time.csv")
data_time = data.matrix(data_time)
cat("Sum of null values in Time data:",sum(is.na(data_time)))
print("First five rows in time data file:")
print(data_time[1:5,])

#All signals are subject to additive noise(assume independent
#and identically distributed Gaussian with zero mean)
#with unknown variance due to distortions during recording.
```

################ Preliminary data analysis ################

Appendix 1.1 (Task 1.1)

```
#Time series plots (audio and EEG signals)
#Prefrontal
plot(data_time,data_eeg[,"prefrontal"], type="l",
    xlab="Time in seconds (Sampling frequency of 20Hz)",
    ylab="EEG signal (Prefrontal)",
    main="Time series plot for EEG signal (Prefrontal)")
#Auditory
plot(data_time,data_eeg[,"auditory"], type="l",
    xlab="Time in seconds (Sampling frequency of 20Hz)",
    ylab="EEG signal (Auditory)",
    main="Time series plot for EEG signal (Auditory)")
#Sound
plot(data_time,data_sound, type="l",
    xlab="Time in seconds (Sampling frequency of 20Hz)",
    ylab="Sound signal",
    main="Time series plot for Sound signal")
```

Appendix 1.2 (Task 1.2)

```
#Distribution for each signal
func_density=function(prm_data, x_label, y_label)
{
  dt_mean=mean(prm_data)
  dt_sd=sd(prm_data)
  dt_densty=dnorm(prm_data,mean=dt_mean,sd=dt_sd)
  plot(prm_data,dt_densty,lty=1,xlab=x_label,ylab=y_label,
      main="Standard Normal Density")
}
func_distribution=function(prm_data, x_label, y_label)
{
  dt_mean=mean(prm_data)
  dt_sd=sd(prm_data)
  dt_dstrbtn=pnorm(prm_data,mean=dt_mean,sd=dt_sd)
  plot(prm_data,dt_dstrbtn,lty=1,xlab=x_label,ylab=y_label,
      main="Standard Normal Distribution")
}
#Prefrontal Distribution
func_density(data_eeg[,"prefrontal"],"Prefrontal","Density")
func_distribution(data_eeg[,"prefrontal"],"Prefrontal","Distribution")
```

```
#Auditory Distribution
func_density(data_eeg[,"auditory"],"Auditory","Density")
func_distribution(data_eeg[,"auditory"],"Auditory","Distribution")
#Sound Distribution
func_density(data_sound,"Sound","Density")
func_distribution(data_sound,"Sound","Distribution")

#Central Tendencies, Scale, Skewness
fun_central_tend=function(prm_data,x_name)
{
  dt_mean=mean(prm_data)
  dt_median=median(prm_data)
  hist(prm_data,breaks=10,xlab=x_name,main=paste("Histogram for",x_name))
  abline(v = dt_mean, lwd = 5 , col="cyan")
  abline(v = dt_median, lwd = 3 , col = "orange")
  dt_mode = unique(prm_data)
  dt_mode = dt_mode[which.max(tabulate(match(prm_data, dt_mode)))]
  abline(v = dt_mode, lwd = 3 , col="blue")
  cat("Mean: ",dt_mean)
  cat("\nMedian: ",dt_median)
  cat("\nMode: ",dt_mode)
}
fun_scale=function(prm_data)
{
  dt_max = max(prm_data)
  dt_min = min(prm_data)
  dt_range = dt_max - dt_min
  dt_variance = var(prm_data)
  dt_std = sd(prm_data)
  cat("Sample Maximum: ",dt_max,"\nSample Minimum: ",dt_min,"\nSample
Range: ",dt_range)
  cat("Sample Variance: ",dt_variance,"\nSample Standard Deviation: ",dt_std)
}
fun_skewness=function(prm_data)
{
  n = length(prm_data)
  dt_skewness = (sqrt(n) * sum( (prm_data - mean(prm_data))^3 ) ) / (sqrt( sum(
(prm_data - mean(prm_data))^2 ) ))^3
  cat("Skewness: %s",dt_skewness)}
```

```
#Prefrontal
fun_central_tend(data_eeg[,"prefrontal"],"Prefrontal")
fun_scale(data_eeg[,"prefrontal"])
fun_skewness(data_eeg[,"prefrontal"])

#Auditory
fun_central_tend(data_eeg[,"auditory"],"Auditory")
fun_scale(data_eeg[,"auditory"])
fun_skewness(data_eeg[,"auditory"])

#Sound
fun_central_tend(data_sound,"Sound")
fun_scale(data_sound)
fun_skewness(data_sound)
```

Appendix 1.1.a

```
#Box plot
boxplot(data_eeg)
#outliers for prefrontal
boxplot.stats(data_eeg[,"prefrontal"])$out
#outliers for auditory
boxplot.stats(data_eeg[,"auditory"])$out
#boxplot for sound
boxplot(data_sound)
boxplot.stats(data_sound)$out
```

Appendix 1.3 (Task 1.3)

```
#Correlation and scatter plots
#scatter plot prefrontal~sound
plot(data_eeg[,"prefrontal"],data_sound,
    ylab="Sound signal",xlab="EEG signal (Prefrontal)",
    main="Correlation and scatter plot for EEG signal (Prefrontal) and Sound signal")
#regression line lm(y~x,data=data)
abline(lm(data_sound~data_eeg[,"prefrontal"]),col="red", lwd=2)
cat("Correlation between EEG signal (Prefrontal) and Sound signal",cor(data_eeg[,"prefrontal"],data_sound))
```

```
#scatter plot Auditory~sound
plot(data_eeg[,"auditory"],data_sound,
    ylab="Sound signal",xlab="EEG signal (Auditory)",
    main="Correlation and scatter plot for EEG signal (Auditory) and Sound
signal")
#regression line lm(y~x,data=data)
abline(lm(data_sound~data_eeg[,"auditory"]),col="blue", lwd=2)
cat("Correlation between EEG signal (Auditory) and Sound
signal",cor(data_eeg[,"auditory"],data_sound))
```

Code for Task 2

Appendix 2 (Task 2)

```
#Determine a suitable mathematical model in explaining the relationship

#between the audio signal y and the two brain signals x1 and x2

#X.csv file contains EEG signals.

#'x1' is prefrontal cortex and 'x2' is auditory cortex.

data_eeg_x = read.csv("X.csv")

data_eeg_x = data.matrix(data_eeg_x)

cat("No. of rows and columns in brain signal
data:",nrow(data_eeg_x),",",ncol(data_eeg_x))

#y.csv file contains sound signal 'y'.

data_sound_y = read.csv("y.csv")

data_sound_y = data.matrix(data_sound_y)

cat("No. of rows and columns in sound signal
data:",nrow(data_sound_y),",",ncol(data_sound_y))

#time.csv contains sampling time of all three signals in 'seconds'.

#2 minutes of signal with sampling frequency of 20Hz.

data_time = read.csv("time.csv")

data_time = data.matrix(data_time)
```

######################### Model 1 y=θ1*x1^3+θ2*x2^5+θbias+ε #########################

Appendix 2.1.1

#Task 2.1#Estimate model parameters

```
matrix_of_ones = matrix(1, nrow(data_eeg_x), 1)

model1_X = cbind(data_eeg_x[,"x1"]^3, data_eeg_x[,"x2"]^5,matrix_of_ones)

model1_theta_hat = solve(t(model1_X) %*% model1_X) %*% t(model1_X) %*% data_sound_y[,"y"]

cat("Model 1 parameter values for θ1, θ2 and θbias:", model1_theta_hat)
```

Appendix 2.2.1

#Task 2.2#RSS

```
model1_y_hat = model1_X %*% model1_theta_hat

model1_error = data_sound_y[,"y"] - model1_y_hat

model1_rss = sum(model1_error^2)

cat("Model 1 residual sum of squared errors:",model1_rss)
```

Appendix 2.3.1

#Task 2.3#log-likelihood

```
model1_num_samples=nrow(data_eeg_x)

model1_resdl_varnc=model1_rss/(model1_num_samples-1)

model1_log_liklhd=-(model1_num_samples/2)*log(2*pi)-(model1_num_samples/2)*log(model1_resdl_varnc)-(1/(2*model1_resdl_varnc))*model1_rss

cat("Model 1 log-likelihood: ",model1_log_liklhd)
```

Appendix 2.4.1

#Task 2.4#Aic & Bic

```
no_estmtd_params=3

model1_aic=(2*no_estmtd_params)-(2*model1_log_liklhd)

model1_bic=(no_estmtd_params*log(model1_num_samples))-
(2*model1_log_liklhd)

cat("Model 1 Akaike information criterion: ",model1_aic)

cat("Model 1 Bayesian information criterion: ",model1_bic)
```

Appendix 2.5.1

#Task 2.5

```
#Q-Q plot

qqnorm(model1_error, main = "Q-Q Plot for Model 1 residual",

       xlab = "Theoretical Quantiles", ylab = "Empirical Quantiles",

       plot.it = TRUE, datax = FALSE,col="blue")

qqline(model1_error,datax = FALSE, distribution = qnorm,col="red")

#histogram

hist(model1_error,breaks=10,col="blue",xlab ="Model 1 residual",

     main="Histogram plot for Model 1 residual")

#Shapiro-Wilk test

shapiro.test(model1_error)

#Kolmogorov-Smirnov test

ks.test(model1_error,"pnorm")
```

########################## Model 2#y=θ1*x1^4 + θ2*x2^2 + θbias + ε ##########################

Appendix 2.1.2

#Task 2.1#Estimate model parameters

```
model2_X = cbind(data_eeg_x[,"x1"]^4, data_eeg_x[,"x2"]^2, matrix_of_ones)

model2_theta_hat = solve(t(model2_X) %*% model2_X) %*% t(model2_X) %*% data_sound_y[,"y"]

cat("Model 2 parameter values for θ1, θ2 and θbias:", model2_theta_hat)
```

Appendix 2.2.2

#Task 2.2#RSS

```
model2_y_hat = model2_X %*% model2_theta_hat

model2_error = data_sound_y[,"y"] - model2_y_hat

model2_rss = sum(model2_error^2)

cat("Model 2 residual sum of squared errors:",model2_rss)
```

Appendix 2.3.2

#Task 2.3#log-likelihood

```
model2_num_samples=nrow(data_eeg_x)

model2_resdl_varnc=model2_rss/(model2_num_samples-1)

model2_log_liklhd=-(model2_num_samples/2)*log(2*pi)-(model2_num_samples/2)*log(model2_resdl_varnc)-(1/(2*model2_resdl_varnc))*model2_rss

cat("Model 2 log-likelihood: ",model2_log_liklhd)
```

Appendix 2.4.2

#Task 2.4#Aic & Bic

```
no_estmtd_params=3
model2_aic=(2*no_estmtd_params)-(2*model2_log_liklhd)
model2_bic=(no_estmtd_params*log(model2_num_samples))-(2*model2_log_liklhd)
cat("Model 2 Akaike information criterion: ",model2_aic)
cat("Model 2 Bayesian information criterion: ",model2_bic)
```

Appendix 2.5.2

#Task 2.5

```
#Q-Q plot
qqnorm(model2_error, main = "Q-Q Plot for Model 2 residual",
    xlab = "Theoretical Quantiles", ylab = "Empirical Quantiles",
    plot.it = TRUE, datax = FALSE,col="blue")
qqline(model2_error,datax = FALSE, distribution = qnorm,col="red")
#histogram
hist(model2_error,breaks=10,col="blue",xlab ="Model 2 residual",
   main="Histogram plot for Model 2 residual")
#Shapiro-Wilk test
shapiro.test(model2_error)
#Kolmogorov-Smirnov test
ks.test(model2_error,"pnorm")
```

######################## Model 3 #y=θ1*x1^3 + θ2*x2 + θ3*x1 + θbias + ε ##########################

Appendix 2.1.3

#Task 2.1#Estimate model parameters

model3_X = cbind(data_eeg_x[,"x1"]^3, data_eeg_x[,"x2"], data_eeg_x[,"x1"], matrix_of_ones)

model3_theta_hat = solve(t(model3_X) %*% model3_X) %*% t(model3_X) %*% data_sound_y[,"y"]

cat("Model 3 parameter values for θ1, θ2, θ3 and θbias:", model3_theta_hat)

Appendix 2.2.3

#Task 2.2#RSS

model3_y_hat = model3_X %*% model3_theta_hat

model3_error = data_sound_y[,"y"] - model3_y_hat

model3_rss = sum(model3_error^2)

cat("Model 3 residual sum of squared errors:",model3_rss)

Appendix 2.3.3

#Task 2.3#log-likelihood

model3_num_samples=nrow(data_eeg_x)

model3_resdl_varnc=model3_rss/(model3_num_samples-1)

model3_log_liklhd=-(model3_num_samples/2)*log(2*pi)-(model3_num_samples/2)*log(model3_resdl_varnc)-(1/(2*model3_resdl_varnc))*model3_rss

cat("Model 3 log-likelihood: ",model3_log_liklhd)

Appendix 2.4.3

#Task 2.4#Aic & Bic

no_estmtd_params=4

model3_aic=(2*no_estmtd_params)-(2*model3_log_liklhd) model3_bic=(no_estmtd_params*log(model3_num_samples))-(2*model3_log_liklhd)

cat("Model 3 Akaike information criterion: ",model3_aic)

cat("Model 3 Bayesian information criterion: ",model3_bic)

Appendix 2.5.3

#Task 2.5

#Q-Q plot

qqnorm(model3_error, main = "Q-Q Plot for Model 3 residual",

xlab = "Theoretical Quantiles", ylab = "Empirical Quantiles",

plot.it = TRUE, datax = FALSE,col="blue")

qqline(model3_error,datax = FALSE, distribution = qnorm,col="red")

#histogram

hist(model3_error,breaks=10,col="blue",xlab ="Model 3 residual",

main="Histogram plot for Model 3 residual")

#Shapiro-Wilk test

shapiro.test(model3_error)

#Kolmogorov-Smirnov test

ks.test(model3_error,"pnorm")

###

########################## Model 4 #y=θ1*x1 + θ2*x1^2 + θ3*x1^3 + θ4*x2^3 + θbias + ε ########################

Appendix 2.1.4

#Task 2.1#Estimate model parameters

model4_X = cbind(data_eeg_x[,"x1"], data_eeg_x[,"x1"]^2, data_eeg_x[,"x1"]^3, data_eeg_x[,"x2"]^3, matrix_of_ones)

model4_theta_hat = solve(t(model4_X) %*% model4_X) %*% t(model4_X) %*% data_sound_y[,"y"]

cat("Model 4 parameter values for θ1, θ2, θ3, θ4 and θbias:", model4_theta_hat)

Appendix 2.2.4

#Task 2.2#RSS

```
model4_y_hat = model4_X %*% model4_theta_hat

model4_error = data_sound_y[,"y"] - model4_y_hat

model4_rss = sum(model4_error^2)

cat("Model 4 residual sum of squared errors:",model4_rss)
```

Appendix 2.3.4

#Task 2.3#log-likelihood

```
model4_num_samples=nrow(data_eeg_x)

model4_resdl_varnc=model4_rss/(model4_num_samples-1)

model4_log_liklhd=-(model4_num_samples/2)*log(2*pi)-
(model4_num_samples/2)*log(model4_resdl_varnc)-
(1/(2*model4_resdl_varnc))*model4_rss

cat("Model 4 log-likelihood: ",model4_log_liklhd)
```

Appendix 2.4.4

#Task 2.4#Aic & Bic

```
no_estmtd_params=5

model4_aic=(2*no_estmtd_params)-(2*model4_log_liklhd)

model4_bic=(no_estmtd_params*log(model4_num_samples))-
(2*model4_log_liklhd)

cat("Model 4 Akaike information criterion: ",model4_aic)

cat("Model 4 Bayesian information criterion: ",model4_bic)
```

Appendix 2.5.4

#Task 2.5#Q-Q plot

```
qqnorm(model4_error, main = "Q-Q Plot for Model 4 residual",
      xlab = "Theoretical Quantiles", ylab = "Empirical Quantiles",
      plot.it = TRUE, datax = FALSE,col="blue")
```

```
qqline(model4_error,datax = FALSE, distribution = qnorm,col="red")

#histogram

hist(model4_error,breaks=10,col="blue",xlab ="Model 4 residual",
    main="Histogram plot for Model 4 residual")

#Shapiro-Wilk test

shapiro.test(model4_error)

#Kolmogorov-Smirnov test

ks.test(model4_error,"pnorm")

######################### Model 5 #y=θ1*x1^3 + θ2*x1^4 + θ3*x2 +
θbias + ε #########################
```

Appendix 2.1.5

#Task 2.1#Estimate model parameters

```
model5_X = cbind(data_eeg_x[,"x1"]^3, data_eeg_x[,"x1"]^4,
data_eeg_x[,"x2"], matrix_of_ones)

model5_theta_hat = solve(t(model5_X) %*% model5_X) %*% t(model5_X)
%*% data_sound_y[,"y"]

cat("Model 5 parameter values for θ1, θ2, θ3 and θbias:", model5_theta_hat)
```

Appendix 2.2.5

#Task 2.2#RSS

```
model5_y_hat = model5_X %*% model5_theta_hat

model5_error = data_sound_y[,"y"] - model5_y_hat

model5_rss = sum(model5_error^2)

cat("Model 5 residual sum of squared errors:",model5_rss)
```

Appendix 2.3.5

#Task 2.3#log-likelihood

```
model5_num_samples=nrow(data_eeg_x)

model5_resdl_varnc=model5_rss/(model5_num_samples-1)
```

```
model5_log_liklhd=-(model5_num_samples/2)*log(2*pi)-
(model5_num_samples/2)*log(model5_resdl_varnc)-
(1/(2*model5_resdl_varnc))*model5_rss

cat("Model 5 log-likelihood: ",model5_log_liklhd)
```

Appendix 2.4.5

#Task 2.4#Aic & Bic

```
no_estmtd_params=4

model5_aic=(2*no_estmtd_params)-(2*model5_log_liklhd)

model5_bic=(no_estmtd_params*log(model5_num_samples))-
(2*model5_log_liklhd)

cat("Model 5 Akaike information criterion: ",model5_aic)

cat("Model 5 Bayesian information criterion: ",model5_bic)
```

Appendix 2.5.5

#Task 2.5

```
#Q-Q plot

qqnorm(model5_error, main = "Q-Q Plot for Model 5 residual",

      xlab = "Theoretical Quantiles", ylab = "Empirical Quantiles",

      plot.it = TRUE, datax = FALSE,col="blue")

qqline(model5_error,datax = FALSE, distribution = qnorm,col="red")

#histogram

hist(model5_error,breaks=10,col="blue",xlab ="Model 5 residual",

    main="Histogram plot for Model 5 residual")

#Shapiro-Wilk test

shapiro.test(model5_error)

#Kolmogorov-Smirnov test

ks.test(model5_error,"pnorm")

#####################################################
```

Appendix 2.6

#Task 2.6#select best regression model

cat("Model 3 is the best performing model among the five given models based on the results.")

Appendix 2.7

#Task 2.7#train-test-split

```
#install.packages("dplyr")

library(dplyr)

set.seed(1)

#create id column to split by id

matrix_id=matrix(1:nrow(data_eeg_x), nrow(data_eeg_x), 1)

colnames(matrix_id)="Id"

data_eeg_sound=cbind(matrix_id,data_eeg_x,data_sound_y)

data_eeg_sound=data.frame(data_eeg_sound)

#get 70% data to train

data_train=data_eeg_sound%>%dplyr::sample_frac(0.70)

#get 30% data to test

data_test=dplyr::anti_join(data_eeg_sound,data_train,by="Id")

cat("No. of rows and columns in original dataset: ",dim(data_eeg_sound))

cat("No. of rows and columns in train dataset: ",dim(data_train))

cat("No. of rows and columns in test dataset: ",dim(data_test))
```

Appendix 2.7.1

#Task 2.7.1#Estimate model parameters using the training dataset

```
#Best model#Model 3 #y=θ1*x1^3 + θ2*x2 + θ3*x1 + θbias + ε

matrix_of_ones = matrix(1, nrow(data_train), 1)

modelBest_X = cbind(data_train[,"x1"]^3, data_train[,"x2"], data_train[,"x1"], matrix_of_ones)
```

```
modelBest_theta_hat = solve(t(modelBest_X) %*% modelBest_X) %*% t(modelBest_X) %*% data_train[,"y"]

cat("Best Model (train data) parameter values for θ1, θ2, θ3 and θbias:", modelBest_theta_hat)
```

Appendix 2.7.2

```
#Task 2.7.2#Compute the model's prediction on the testing data

matrix_of_ones_test = matrix(1, nrow(data_test), 1)

modelBest_X_test = cbind(data_test[,"x1"]^3, data_test[,"x2"], data_test[,"x1"], matrix_of_ones_test)

modelBest_y_hat = modelBest_X_test %*% modelBest_theta_hat

modelBest_error = data_test[,"y"] - modelBest_y_hat

modelBest_rss = sum(modelBest_error^2)

cat("Best Model (Test data) residual sum of squared errors:",modelBest_rss)

cat("Predicted Values:", modelBest_y_hat)
```

Appendix 2.7.3

```
#Task 2.7.3#Compute 95% confidence interval, plot them with error bars

#together with model prediction and testing data samples.

#Va(yhat)=sigma^2*x(X^T*X)^-1*x^T

modeltest_num_samples=nrow(data_test)

modeltest_resdl_varnc=modelBest_rss/(modeltest_num_samples-1)#variance

X_trnsp_X_invrs=solve(t(modelBest_X_test)%*%modelBest_X_test)

var_yhat=matrix(1,modeltest_num_samples,1)

for(i in 1:modeltest_num_samples)

{  x_i=matrix(modelBest_X_test[i,],1,4)

  #print(x_i)

  var_yhat[i,1]=modeltest_resdl_varnc*x_i%*%X_trnsp_X_invrs%*%t(x_i)

}
```

```
#conf_intrvl=yhat(+/-)1.96*sqrt(Var(yhat))

conf_intrvl=1.96*(sqrt(var_yhat))

cf_upper=modelBest_y_hat+conf_intrvl

cf_lower=modelBest_y_hat-conf_intrvl

#plot confidence interval

#split time data

data_time_id=cbind(matrix_id,data_time)

data_time_id=data.frame(data_time_id)

data_time_train=data_time_id%>%dplyr::sample_frac(0.70)

data_time_test=dplyr::anti_join(data_time_id,data_time_train,by="Id")

#all points

plot(x=data_time_test[,"time"],modelBest_y_hat,col="orange",

    main="Confidence interval (all data)",

    xlab="Sampling Time in seconds",ylab="Predicted values")#predicted

#points(x=data_time_test[,"time"],data_test[,"y"],col="green")#actual

segments(data_time_test[,"time"],cf_lower,data_time_test[,"time"],cf_upper,col
= "blue",lwd=2)#ci

legend(0,95,legend=c("Predicted values","Confidence
Interval"),fill=c("orange","blue"))

#first 50 points

rng=1:50

plot(x=data_time_test[rng,"time"],modelBest_y_hat[rng,],col="orange",lwd=2,

    main="Confidence interval (first 50 points)",

    xlab="Sampling Time in seconds",ylab="Predicted values")#predicted

points(x=data_time_test[rng,"time"],data_test[rng,"y"],col="green",lwd=2)#act
ual

segments(data_time_test[rng,"time"],cf_lower[rng],data_time_test[rng,"time"],c
f_upper[rng],

        col = "blue",lwd=2)#ci
```

```
legend(0,95,legend=c("Predicted values","Confidence Interval","Actual values"),
       fill=c("orange","blue","green"))
```

Appendix 3

Code for Task 3

```
#compute posterior distribution of best model using rejection abc
#Model 3: y = 2.715713*𝑥1^3+-3.15135*𝑥2+4.18139*𝑥1+-6.6514+ε
#The two parameters with largest absolute values are 2.715713 and 4.18139
set.seed(1)
prior_x1cube=runif(n=1500,min=2.715713-1,max=2.715713+1)
prior_x1=runif(n=1500,min=4.18139-1,max=4.18139+1)
#perform rejection abc
selctd_x1cube=matrix(1,modeltest_num_samples,1)
selctd_x1=matrix(1,modeltest_num_samples,1)
res_all=matrix(1,1500,3)
colnames(res_all)=c("x1cube","x1","varnc")

for(i in 1:1500)
{
  theta_abc=matrix(c(prior_x1cube[i],-3.15135,prior_x1[i],-6.6514),4,1)
  y_hat_abc = modelBest_X_test %*% theta_abc
  model_abc_error = data_test[,"y"] - y_hat_abc
  model_abc_rss = sum(model_abc_error^2)
  model_abc_varnc=model_abc_rss/(modeltest_num_samples-1)#variance
  res_all[i,]=c(prior_x1cube[i],prior_x1[i],model_abc_varnc)
}
```

```
c=0
for(i in 1:1500)
{
  if(res_all[i,"varnc"]<=modeltest_resdl_varnc+2.6)
  {
    c=c+1
    selctd_x1cube[c,1]=res_all[i,"x1cube"]
    selctd_x1[c,1]=res_all[i,"x1"]
  }
  if(c>=modeltest_num_samples)
    break
}
#print("c")
#print(c)
#plot joint and marginal posterior distribution
#selctd_x1cube#2.715713*x1^3
dt_mean=mean(selctd_x1cube)
dt_sd=sd(selctd_x1cube)
dt_densty1=dnorm(selctd_x1cube,mean=dt_mean,sd=dt_sd)
plot(selctd_x1cube,dt_densty1,lty=1,xlab="First Parameter",ylab="Distribution
and Density",col="blue",
     main="Marginal Posterior Distribution (First Parameter)")
dt_dstrbtn1=pnorm(selctd_x1cube,mean=dt_mean,sd=dt_sd)
points(selctd_x1cube,dt_dstrbtn1,lty=1,col="green")
legend(2,1.2,legend=c("Density","Distribution"),
       fill=c("blue","green"))
```

```
hist(selctd_x1cube,breaks=10,xlab="First Parameter",ylab="Frequency",
    main="Histogram for marginal Posterior Distribution (First Parameter)")

#selctd_x1[c,1]#4.18139*x1
dt_mean=mean(selctd_x1)
dt_sd=sd(selctd_x1)
dt_densty2=dnorm(selctd_x1,mean=dt_mean,sd=dt_sd)
dt_dstrbtn2=pnorm(selctd_x1,mean=dt_mean,sd=dt_sd)
plot(selctd_x1,dt_dstrbtn2,lty=1,xlab="Second Parameter",ylab="Distribution and Density",col="green",
    main="Marginal Posterior Distribution (Second Parameter)")
points(selctd_x1,dt_densty2,lty=1,col="blue")
legend(2,1.2,legend=c("Density","Distribution"),
      fill=c("blue","green"))
hist(selctd_x1,breaks=10,xlab="Second Parameter",ylab="Frequency",
    main="Histogram for marginal Posterior Distribution (Second Parameter)")

#Joint Posterior
plot(selctd_x1cube, selctd_x1,lty=1,xlab="First Parameter",ylab="Second Parameter",col="blue",
    main="Joint Posterior Distribution plot")
##########
#install.packages("LaplacesDemon")
library(LaplacesDemon)
joint.density.plot(selctd_x1cube, selctd_x1,
                Title="Joint Density Plot for first and second parameter",
                contour=TRUE, color=FALSE)
```