Assignment 2

Applied Bayesian Statistics

Archit Sharma

S3755732

Contents

[Table of Figures 2](#_Toc51512811)

[Introduction 3](#_Toc51512812)

[About Data 3](#_Toc51512813)

[Descriptive Statistics 3](#_Toc51512814)

[Response vs. Predictors 3](#_Toc51512815)

[Correlation between Predictors 5](#_Toc51512816)

[Exploring Response Variable 5](#_Toc51512817)

[Methodology 6](#_Toc51512818)

[Result 7](#_Toc51512819)

[Defining JAGS Model 7](#_Toc51512820)

[Defining Priors 8](#_Toc51512821)

[Compiling Model 8](#_Toc51512822)

[Parameter Tuning 9](#_Toc51512823)

[1st Run 9](#_Toc51512824)

[2nd Run 10](#_Toc51512825)

[3rd Run 11](#_Toc51512826)

[4th Run 12](#_Toc51512827)

[5th Run 13](#_Toc51512828)

[Exploring Posterior Distribution 20](#_Toc51512829)

[Posterior for Predictors 21](#_Toc51512830)

[Posterior for Predictions 22](#_Toc51512831)

[Predictive Check 23](#_Toc51512832)

[Conclusion 23](#_Toc51512833)

[Appendix 25](#_Toc51512834)

# Table of Figures

[Figure 1: Descriptive statistics for predictors and response 3](#_Toc51512748)

[Figure 2: Scatter plot of Response vs. Area of Property 4](#_Toc51512749)

[Figure 3: Scatter plot for Response vs. Ordinal and Nominal Predictors 4](#_Toc51512750)

[Figure 4: Correlation matrix for predictors 5](#_Toc51512751)

[Figure 5: Histogram for Response 5](#_Toc51512752)

[Figure 6: Density plot for Response 6](#_Toc51512753)

[Figure 7: JAGS model diagram 8](#_Toc51512754)

[Figure 8: Model compiled in the software 9](#_Toc51512755)

[Figure 9: Diagnostics for tau 10](#_Toc51512756)

[Figure 10: Diagnostics for beta[5] 11](#_Toc51512757)

[Figure 11: Diagnostics for tau 12](#_Toc51512758)

[Figure 12: Diagnostics for beta[4] 13](#_Toc51512759)

[Figure 13: Diagnostics for beta[0] 14](#_Toc51512760)

[Figure 14: Diagnostics for beta[1] 15](#_Toc51512761)

[Figure 15: Diagnostics for beta[2] 16](#_Toc51512762)

[Figure 16: Diagnostics for beta[3] 17](#_Toc51512763)

[Figure 17: Diagnostics for beta[4] 18](#_Toc51512764)

[Figure 18: Diagnostics for beta[5] 19](#_Toc51512765)

[Figure 19: Diagnostics for tau 20](#_Toc51512766)

[Figure 20: Posterior distribution for intercept 21](#_Toc51512767)

[Figure 21: Posterior distribution for predictors 21](#_Toc51512768)

[Figure 22: Posterior distribution for Prediction from 1st and 2nd configurations 22](#_Toc51512769)

[Figure 23: Posterior distribution for Prediction from 3rd, 4th and 5th configurations 22](#_Toc51512770)

[Figure 24: Density plot for observed vs. predicted values 23](#_Toc51512771)

[Table 1: 1st Run for MCMC model 9](#_Toc51512772)

[Table 2: 2nd Run for MCMC model 10](#_Toc51512773)

[Table 3: 3rd Run for MCMC model 11](#_Toc51512774)

[Table 4: 4th Run for MCMC model 12](#_Toc51512775)

[Table 5: 5th Run for MCMC model 13](#_Toc51512776)

[Table 6: Inferences from posterior distributions of predictions 23](#_Toc51512777)

[Table 7: Chosen parameters for simulating MCMC model 23](#_Toc51512778)

[Table 8: Point estimates and 95% HDI for predictions 24](#_Toc51512779)

# Introduction

Before COVID-19, Australia’s real estate was presumed as a constantly booming industry, for a significantly long period of time. As part of the development plans for its major cities, assessment and prediction of volatility in property prices has been a hot topic amongst data scientists ever since. The analysts have been putting in tremendous efforts to collect and examine property prices in the city of Melbourne, which is a major contributor in Australia’s economy.

The core procedure executed for this analysis is Markov Chain Monte Carlo (MCMC) Bayesian analysis, on a dataset comprising of area of property, number of bedrooms, bathrooms, carparks and type, along with corresponding property prices. The assessment was carried out to understand the statistical properties of the data collected and build a multiple linear model for predicting property prices in Melbourne.

# About Data

The dataset comprises of 10,000 observations for 5 predictors and 1 response variable (Y), i.e. sale price of Melbourne’s properties in 100,000 of AUD. The predictors are as follows:

1. Area (X1): Land size in m2 of the sold property
2. Bedrooms (X2): The number of bedrooms
3. Bathrooms (X3): The number of bathrooms
4. CarParks (X4): The number of car parks
5. PropertyType (X5): The type of the property (0: House, 1: Unit)

## Descriptive Statistics

The descriptive statistics if dataset are as follows:

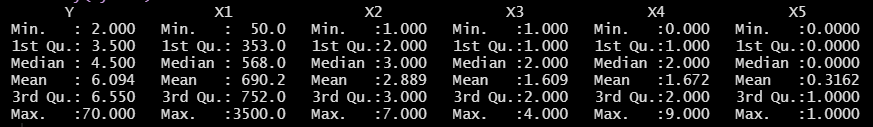


Figure 1: Descriptive statistics for predictors and response

From a classical statistics viewpoint, the mean and sample variance of response variable were computed to be approximately 6.1 and 26.24, respectively. X1 variable was observed to be continuous in nature, whereas X2, X3 and X4 were found to be ordinal. Variable X5 had values 0 (House) and 1 (Unit), which indicated nominal nature of variable.

## Response vs. Predictors

To understand the interaction between predictors and response variables, 2-dmensional scatter plots were obtained:

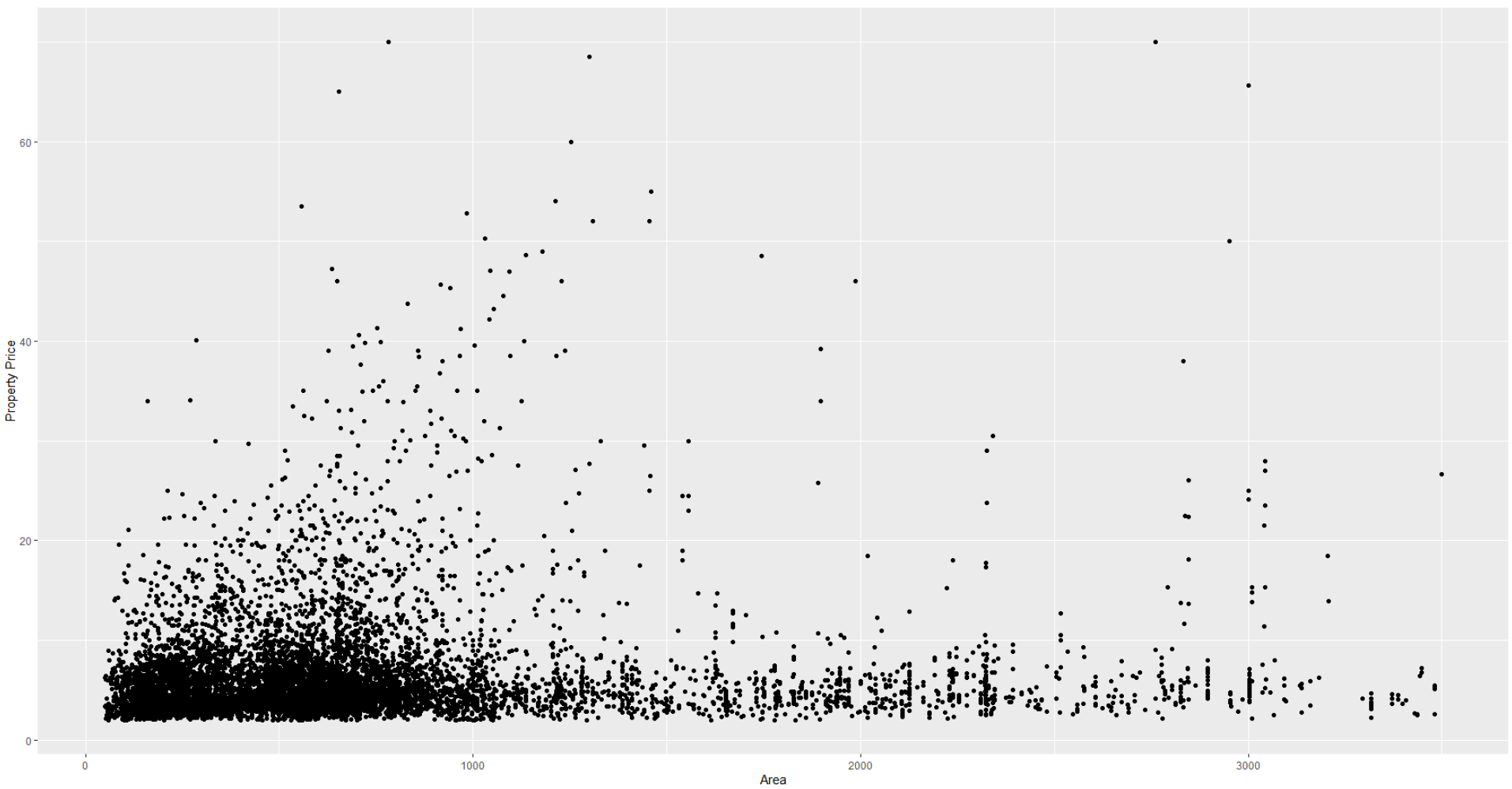


Figure 2: Scatter plot of Response vs. Area of Property

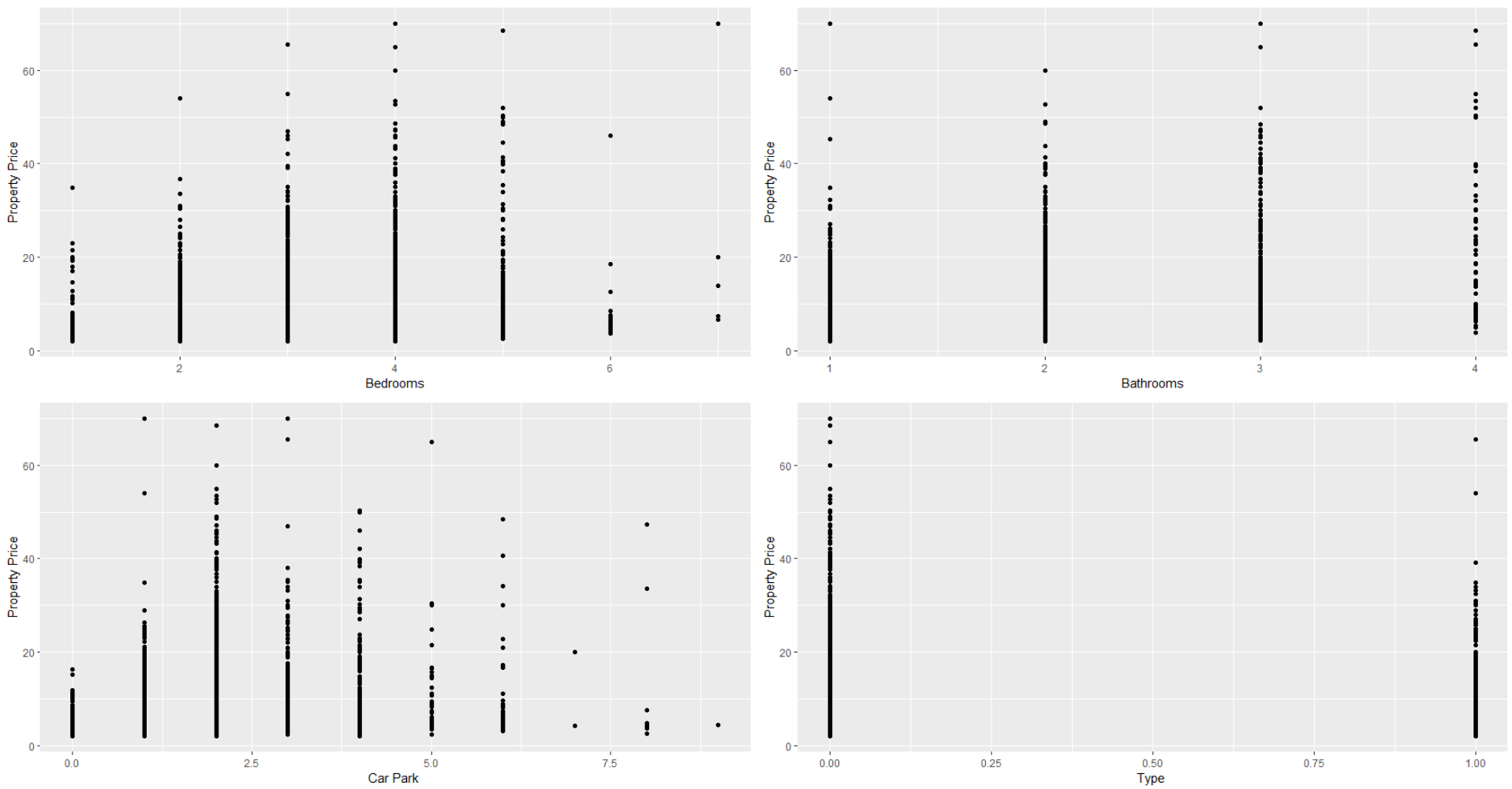


Figure 3: Scatter plot for Response vs. Ordinal and Nominal Predictors

The following inferences were drawn from the scatter plots:

* Y vs. Area: There seems to be a slight presence of a positive linear relationship. However, there is a heavy presence of outliers. This might be reason behind the trend being distorted.
* Y vs. Ordinals: As the ordinal levels progress, there seems to be a positive relationship between the 2 variables. However, after a point, a negative relationship can be observed, as the property prices dip with the increasing ordinal value.
* Y vs. Type: It can be observed that a significant number of Houses (represented by 0) had a property price greater than Unit (represented by 1).

## Correlation between Predictors

A prerequisite for multiple linear regression modelling is that the predictors should not be correlated. High correlation might result in skewed modelling and subsequent predictions. The correlations matrix was obtained for the predictors:

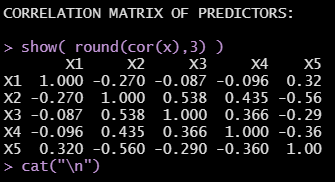


Figure 4: Correlation matrix for predictors

The correlation between number of bedrooms and bathrooms had a substantial magnitude of 0.538. The rest of the variables were not found to be highly correlated. The correlation value of ≈0.5 was still considered safe and accepted for analysis.

## Exploring Response Variable

To understand the nature and distribution of property prices, a histogram was visualized:

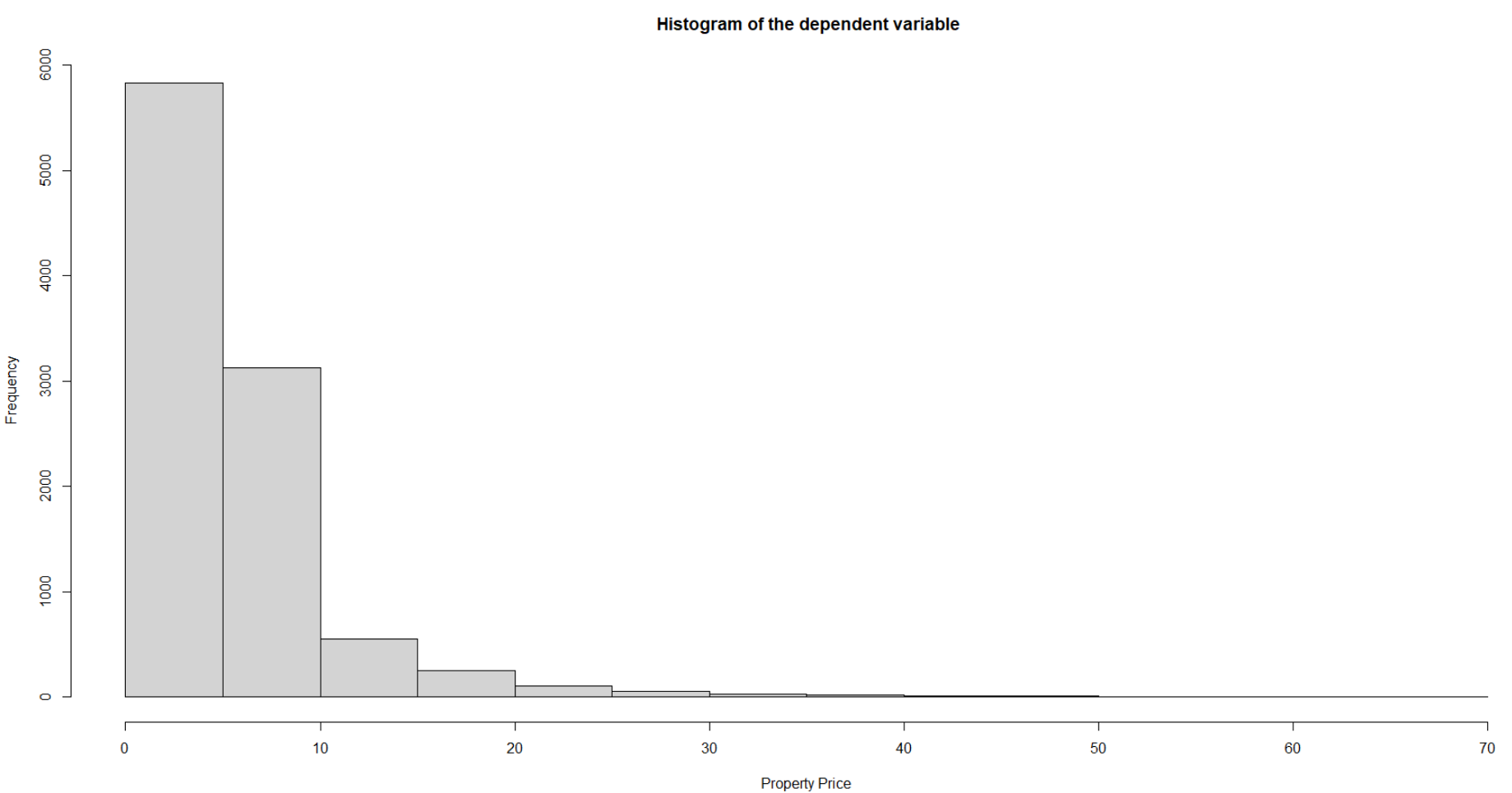


Figure 5: Histogram for Response

It can be seen that majority of the observations lie between $0 to $1,000,000. A downward trend is clearly visible. The graph denotes a heavily right-skewed distribution of observations. To understand the distribution of variable, a density plot was obtained for the same:

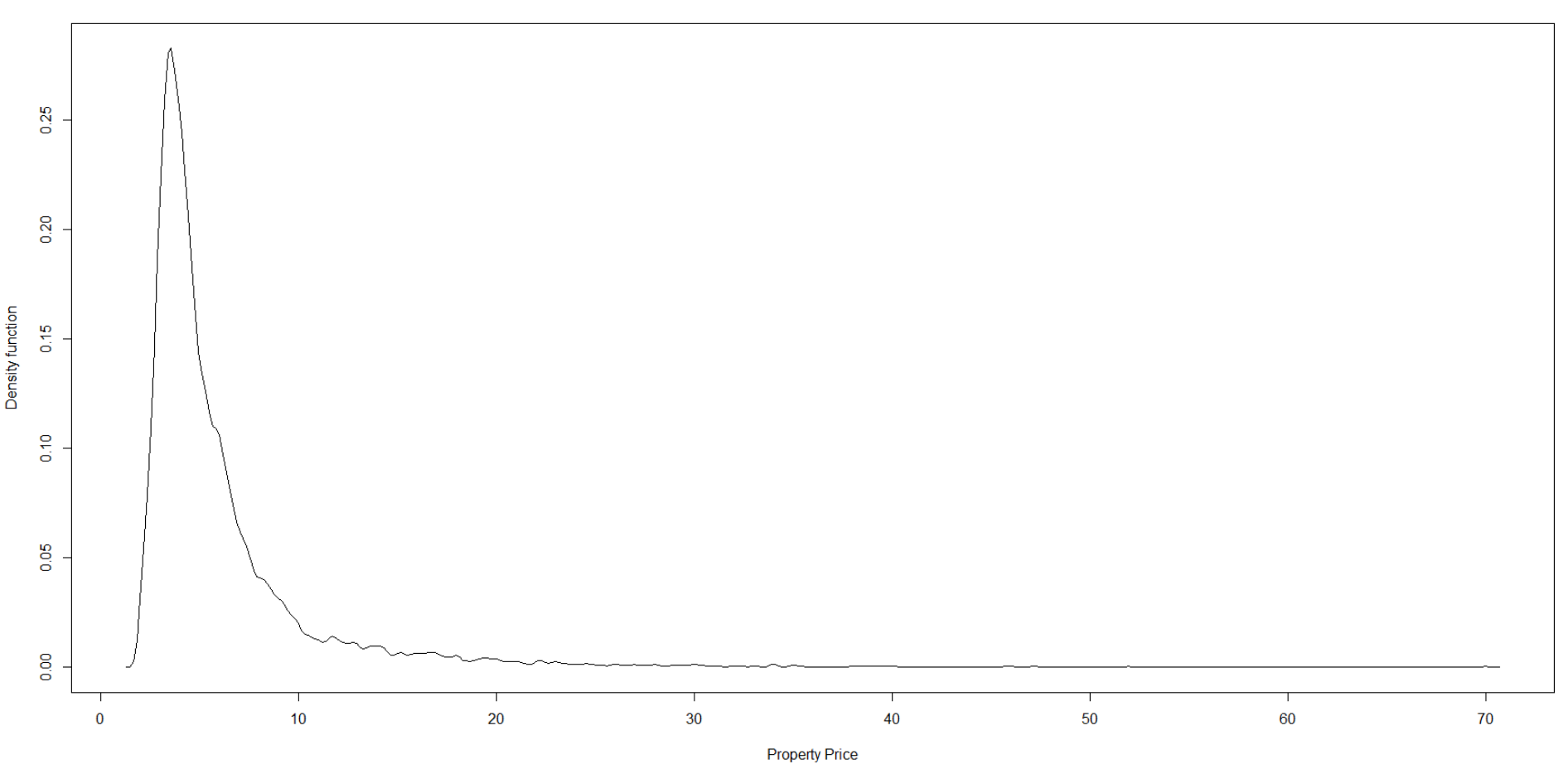


Figure 6: Density plot for Response

It can be seen that the variable follows a Gamma distribution. This is an indication of a dominant gamma distributed likelihood.

# Methodology

The analysis was carried out using Just Another Gibbs Sampler(JAGS) and R programming software. After obtaining the descriptive statistics of the data, the next steps involve the following:

* Creation of JAGS model diagram for the setting
  + Assess the descriptive statistics and graphical representations of variables
  + Determine the distribution of likelihood and prior
  + Visualize the distributions into a model diagram
* Specification of prior distributions as per the following expert knowledge provided in the problem statement
  + Area (X1): Every m2 increase in land size increases the sales price by 90 AUD. This is a very strong expert knowledge.
  + Bedrooms (X2): Every additional bedroom increases the sales price by 100,000 AUD. This is a weak expert knowledge.
  + Bathrooms (X3): There is no expert knowledge on the number of bathrooms.
  + CarParks (X4): Every additional car space increases the sales price by 120,000 AUD. This is a strong expert knowledge.
  + PropertyType (X5): If the property is a unit, the sale price will be 150,000 AUD less than that of a house on the average. This is a very strong expert knowledge.
* Compile the model using R programming and JAGS, and perform MCMC diagnostics to assess the effect of:
  + Number of chains
  + Burn-in period
  + Thinning
  + Number of saved steps
* After obtaining suitable diagnostics, visualize the posterior distribution for each of the variables and draw relevant inferences
* Obtain prediction model using the Bayesian point estimates
* Find predictions of sale prices for the information provided in problem statement

# Result

The aforementioned methodology was carried out, and the following subsections state and discuss the results of MCMC Bayesian analysis and prediction of data.

## Defining JAGS Model

In order to perform multiple linear regression on data, the mathematical model was defined as:

**Y =** **β0 + β1 xi1 + β2 xi2+ β3 xi3 + β4 xi4 + β5 xi5 + ε**

*where, ε ~ Gamma(µ2/σ2 , µ/σ2)*

Based on the density plot of response variable, it was safely assumed that it had a significantly dominating gamma distribution. The gamma-model for response variable was written as:

**yi ~ Gamma(µi2/σ2 , µi/σ2)**

Furthermore, variance **σ** was also assumed to be gamma distributed as:

**σ ~ Gamma(A ,B)**

The β priors were assumed to be normally distributed, as shown below:

**βi ~ Normal(Mi , Si)**

*where, M and S represent mean and variance*

Finally, the JAGS diagram model was visualized as:

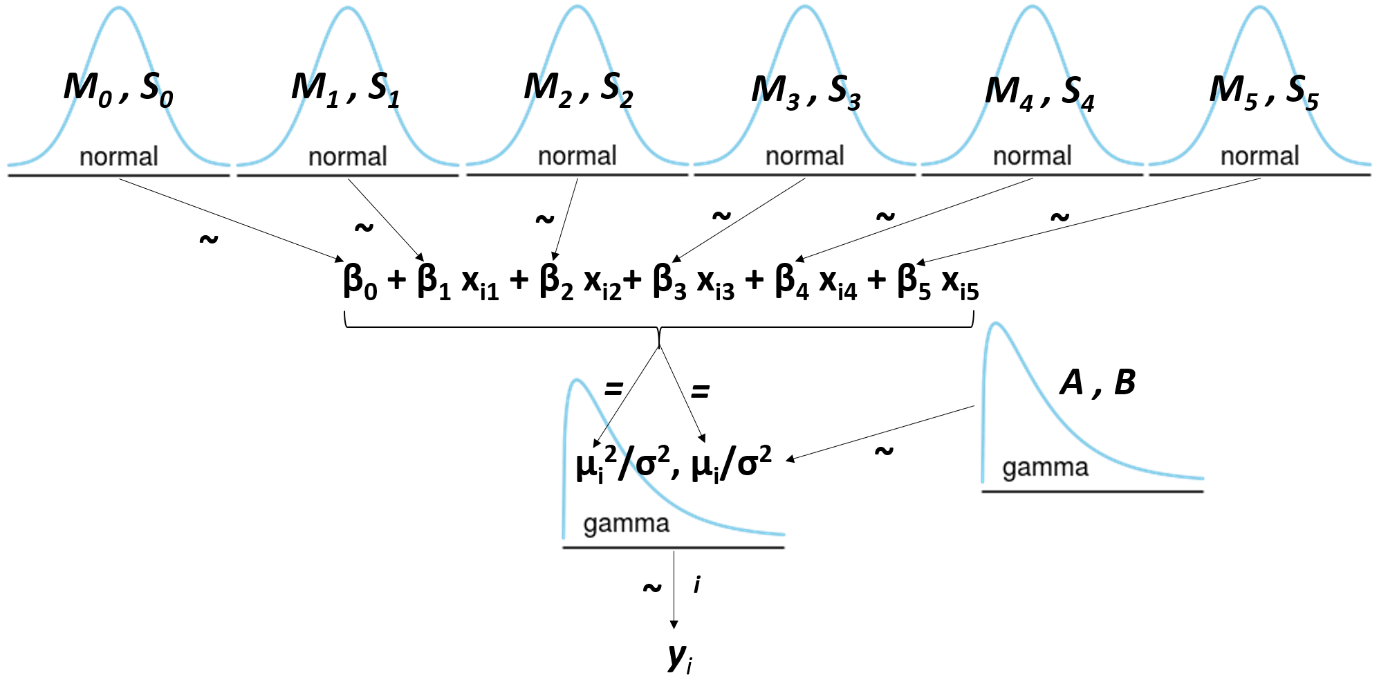


Figure 7: JAGS model diagram

## Defining Priors

As per the expert knowledge and degree of belief provided in problem statement, the following assumptions were made for defining mean (Mi), variance (Si), shape (A) and scale/rate (B) of priors:

* Intercept: β0  ~ Normal(0 , 4)
* Area: β1  ~ Normal(90/100000 , 0.1) 🡪 0.1 indicates very strong knowledge
* Bedrooms: β2  ~ Normal(100000/100000 , 4) 🡪 4 indicates weak knowledge
* Bathrooms: β3  ~ Normal(0 , 4) 🡪 4 indicates no knowledge
* Carparks: β4  ~ Normal(120000/100000 , 1) 🡪 1 indicates strong knowledge
* Type: β5  ~ Normal(150000/100000, 0.1) 🡪 0.1 indicates strong knowledge
* Variance: σ ~ Gamma(0.1 , 0.1)

The variance in β priors was assumed as per the belief that very strong knowledge denotes least variance and weak or no knowledge indicates high variance in amongst the observations. The unit changes in response variable were introduced in the form of mean for all the normally distributed variables.

## Compiling Model

Before proceeding further, the variables were scaled to ensure efficient and reliable outcome. Hence, standard deviation of each variable was introduced in the model and reflected in all the presumed expressions of distribution. Below is a snapshot of simulated model:

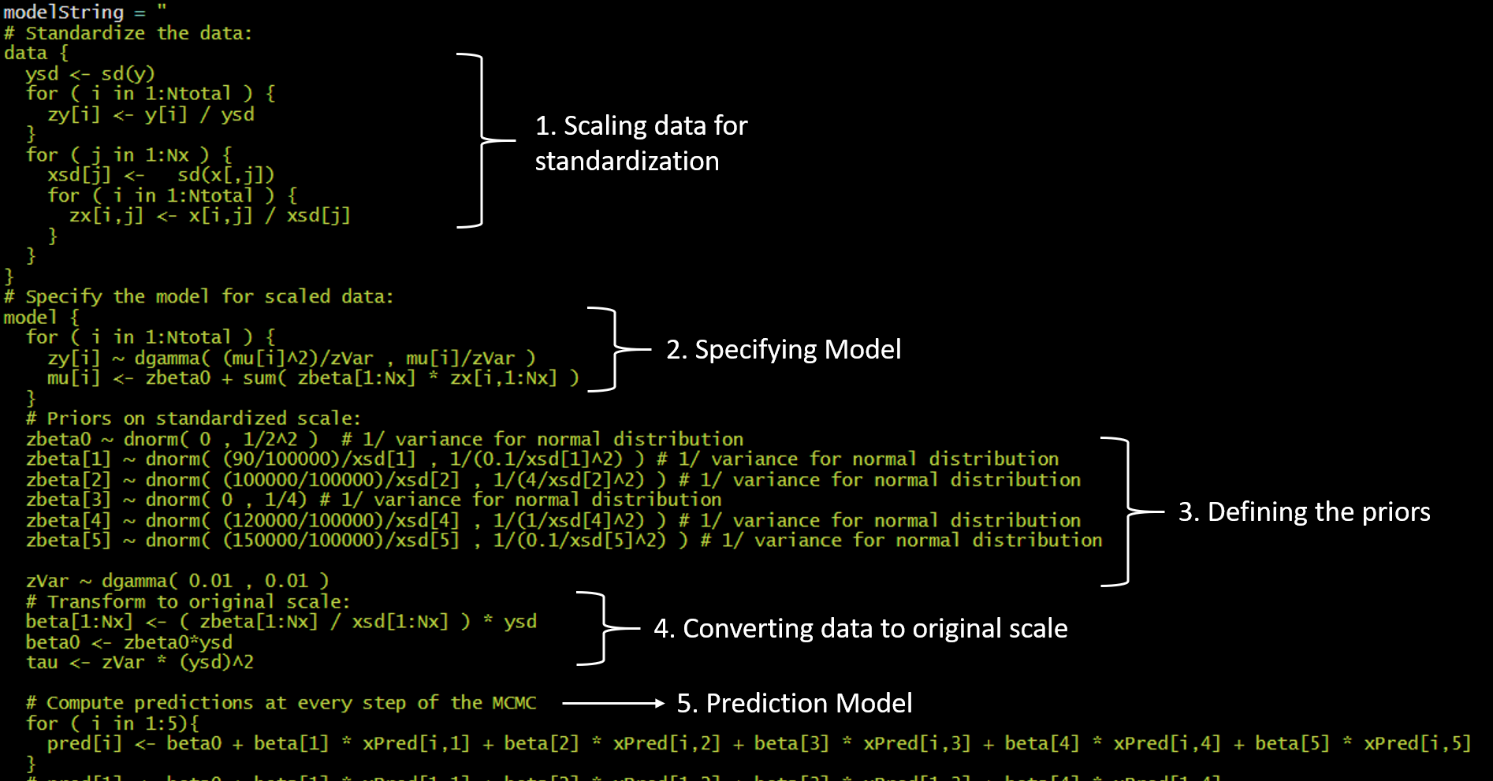


Figure 8: Model compiled in the software

## Parameter Tuning

The following parameters were toggled for obtaining an efficient and accurate representation of data:

* Number of chains
* Burn-in period
* Thinning
* Number of saved steps
* Adapt steps

For every run, the diagnostics of β and tau were examined.

### 1st Run

|  |  |
| --- | --- |
| Number of Chains | 3 |
| Burn-in Period | 2000 |
| Thinning | 2 |
| Number of Saved Steps | 4000 |
| Adapt Steps | 1000 |

Table 1: 1st Run for MCMC model

Additionally, the initial list of starting points for each variable was not specified at first. This was done to check if the model was able to identify efficient starting points on its own.

The model was successfully able to identify the starting points on its own. However, the diagnostics turned out to be pretty bad. For example, below is a screenshot of the diagnostics obtained for tau:

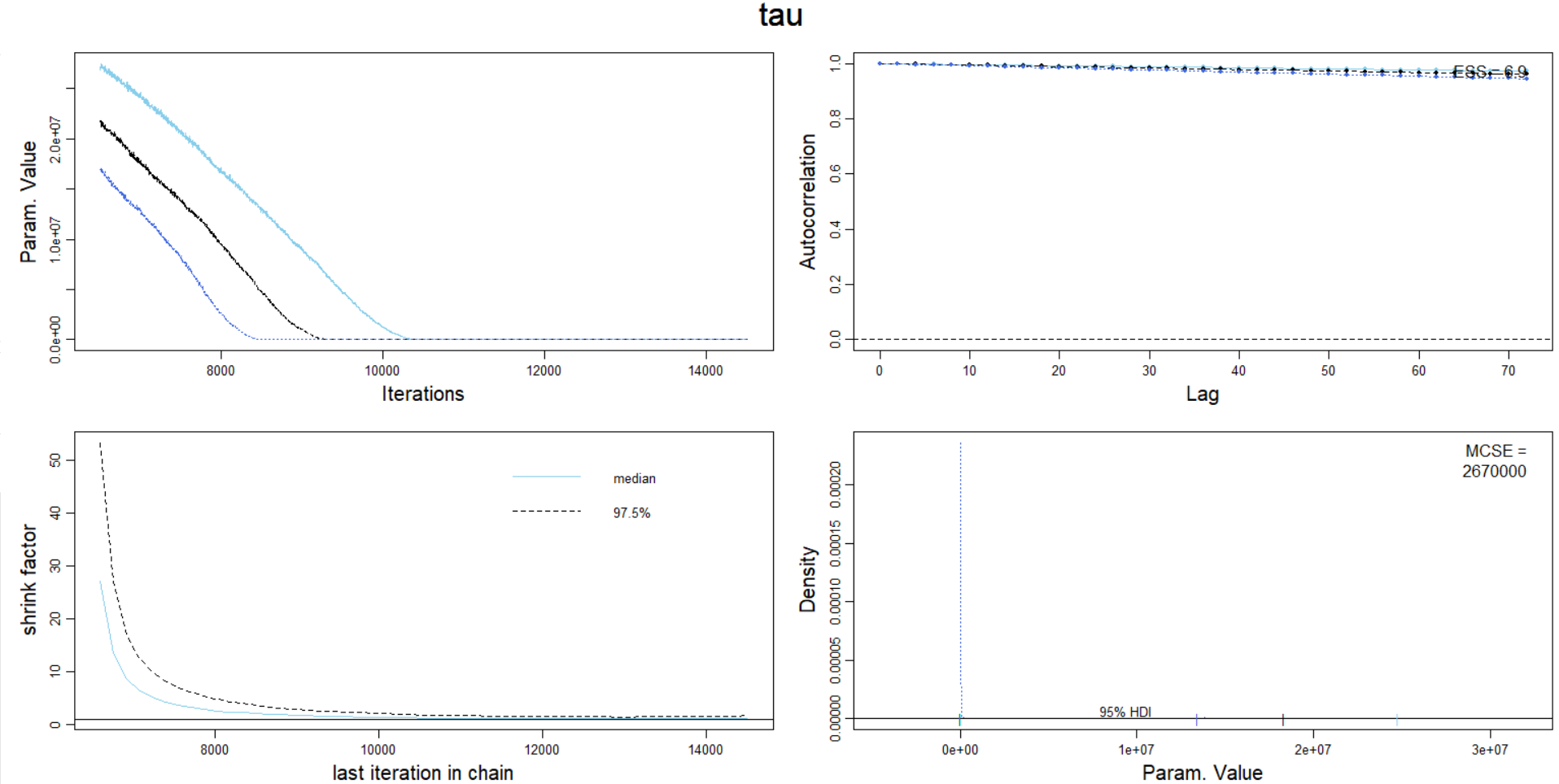


Figure 9: Diagnostics for tau

It can be clearly observed that:

* The chains do not converge in the trace plots
* Representativeness 🡪 Very Bad
  + Shrink factor >> 1.2, which indicates bad representativeness
* Accuracy 🡪 Very Bad
  + Very high autocorrelation observed in observations
  + Very high MCSE (2670000)
  + HDIs in density plot do not overlap

In conclusion, the following inferences were drawn for the diagnostics:

* Acceptable 🡪 None
* Unacceptable 🡪 tau, β1 ,β0 , β2 , β3 , β4 , β5

### 2nd Run

|  |  |
| --- | --- |
| Number of Chains | 3 |
| Burn-in Period | 3000 |
| Thinning | 3 |
| Number of Saved Steps | 4000 |
| Adapt Steps | 1500 |

Table 2: 2nd Run for MCMC model

In order to solve the issue of divergence in chains, adapt steps and burn-in period were increased. Thinning was also increase to 3, in order to the problem of substantial autocorrelation.

It was observed that the chains were finally converging. However, the diagnostics were still pretty bad. Below is a snapshot of diagnostics for β5:

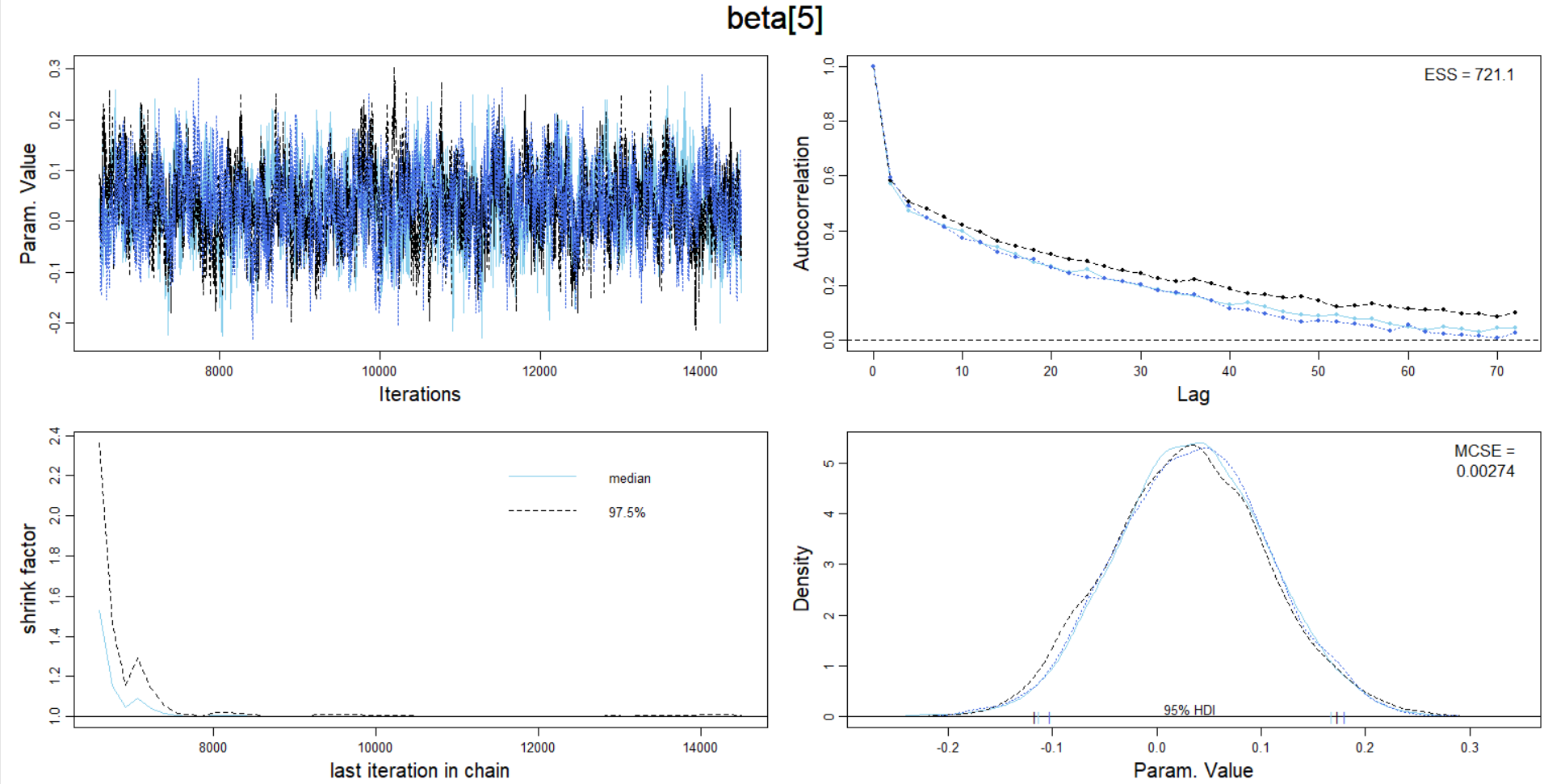


Figure 10: Diagnostics for beta[5]

It can be clearly observed that:

* The chains converge in the trace plots, which can still be improved
* Representativeness 🡪 Very Bad
  + Shrink factor >> 1.2, which indicates bad representativeness
* Accuracy 🡪 Bad
  + High autocorrelation observed in observations, but better than the previous run
  + Low MCSE (0.00274)
  + HDIs in density plot do not overlap, but are very close

In conclusion, the following inferences were drawn for the diagnostics:

Acceptable 🡪 tau, β1

Unacceptable 🡪 β0 , β2 , β3 , β4 , β5

### 3rd Run

|  |  |
| --- | --- |
| Number of Chains | 3 |
| Burn-in Period | 4000 |
| Thinning | 5 |
| Number of Saved Steps | 4000 |
| Adapt Steps | 1500 |

Table 3: 3rd Run for MCMC model

In order to obtain better convergence between chains, and get rid of autocorrelation, all the parameters were increased, except number of chains. Number of chains was left unaltered to maintain efficiency of MCMC process.

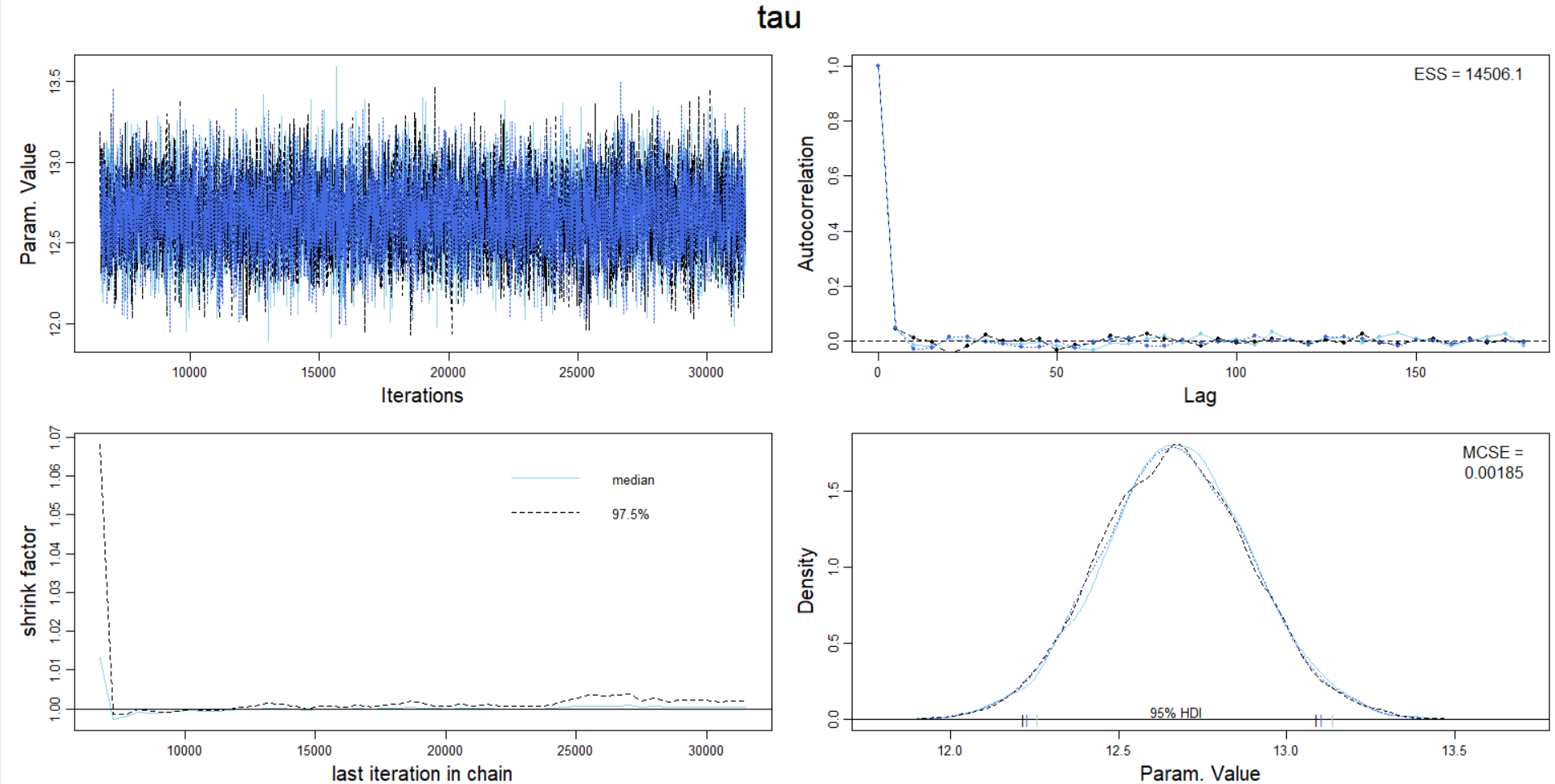
A much better convergence was achieved, with lesser autocorrelations. The shrink factor was also reduced significantly and HDIs were found to be overlapping in density plot. Below is a screenshot of diagnostics for tau:

Figure 11: Diagnostics for tau

In conclusion, the following inferences were drawn for the diagnostics:

Acceptable 🡪 tau, β1

Unacceptable 🡪 β0 , β2 , β3 , β4 , β5

### 4th Run

|  |  |
| --- | --- |
| Number of Chains | 3 |
| Burn-in Period | 5000 |
| Thinning | 15 |
| Number of Saved Steps | 5000 |
| Adapt Steps | 1500 |

Table 4: 4th Run for MCMC model

In order to obtain a lower shrink factor and better convergence of chains, the burn-in period was further increased. To tackle the autocorrelation, thinning was tripled, and number of saved steps was also increased.

The following improvements were observed for all the priors:

* A shrink factor of less than 1.2 was successfully achieved
* The chains were observed to be converging very well
* The HDIs in density plots were either extremely close or overlapping
* The MCSE was observed to be very close to 0
* ESS was observed to be decently high, but could be better

The issue observed was the existence of a few autocorrelations. Below is a screenshot of :

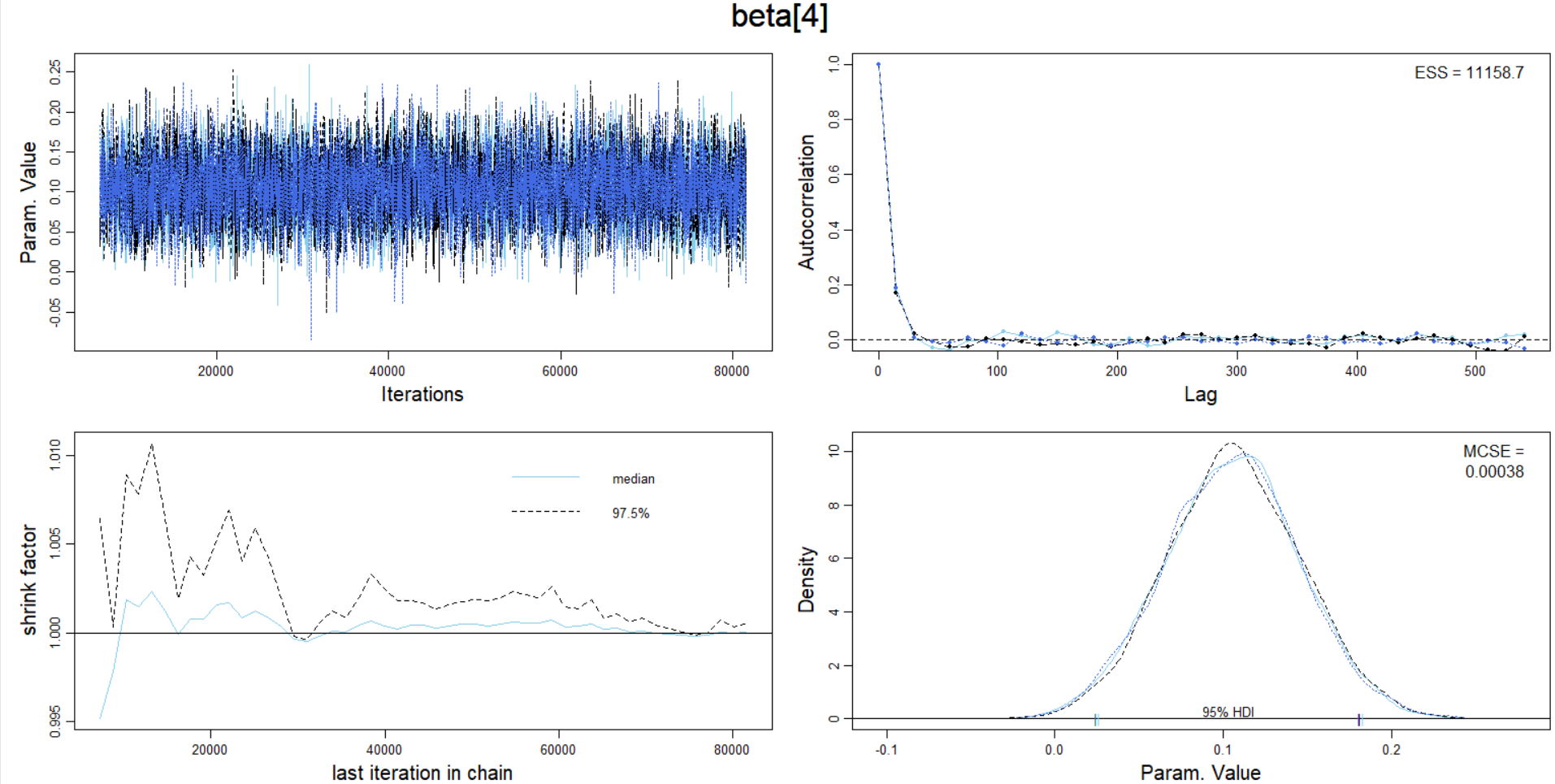


Figure 12: Diagnostics for beta[4]

In conclusion, the following inferences were drawn for the diagnostics:

Acceptable 🡪 tau, β1

Unacceptable 🡪 β0 , β2 , β3 , β4 , β5

### 5th Run

|  |  |
| --- | --- |
| Number of Chains | 3 |
| Burn-in Period | 5000 |
| Thinning | 30 |
| Number of Saved Steps | 5000 |
| Adapt Steps | 1500 |

Table 5: 5th Run for MCMC model

Finally, the thinning was increased to 30 for tackling the existence of autocorrelation.

Below are the snapshot of diagnostics and their inferences:

**Intercept β0 :**

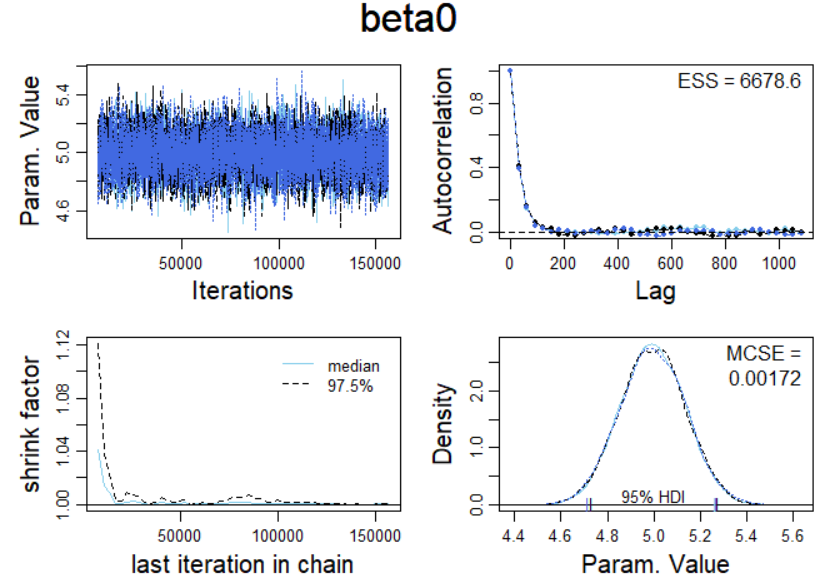


Figure 13: Diagnostics for beta[0]

* Chains converge very well
* Shrink factor < 1.2
* High ESS
* MCSE close to 0
* HDIs overlap
* A few autocorrelations

**Area β1 :**

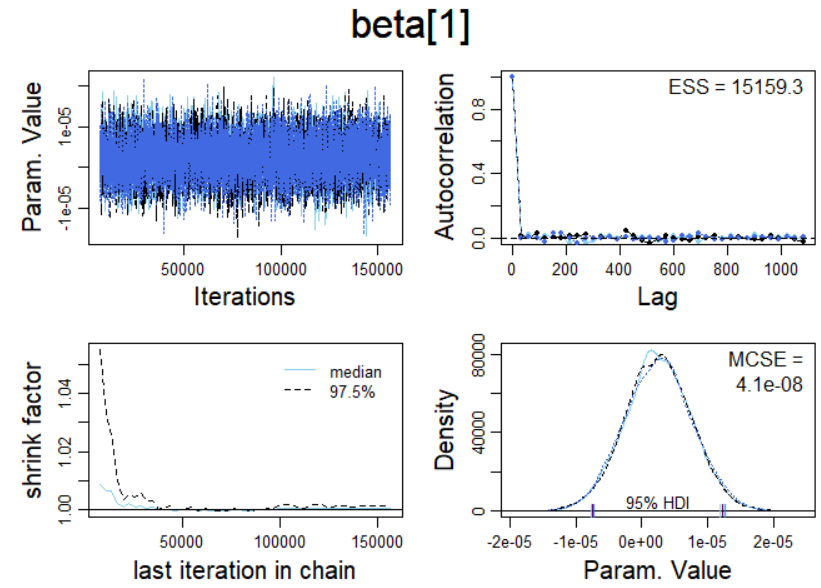


Figure 14: Diagnostics for beta[1]

* Chains converge very well
* Shrink factor < 1.2
* High ESS
* MCSE close to 0
* HDIs overlap
* No autocorrelations

**Bedrooms β2 :**

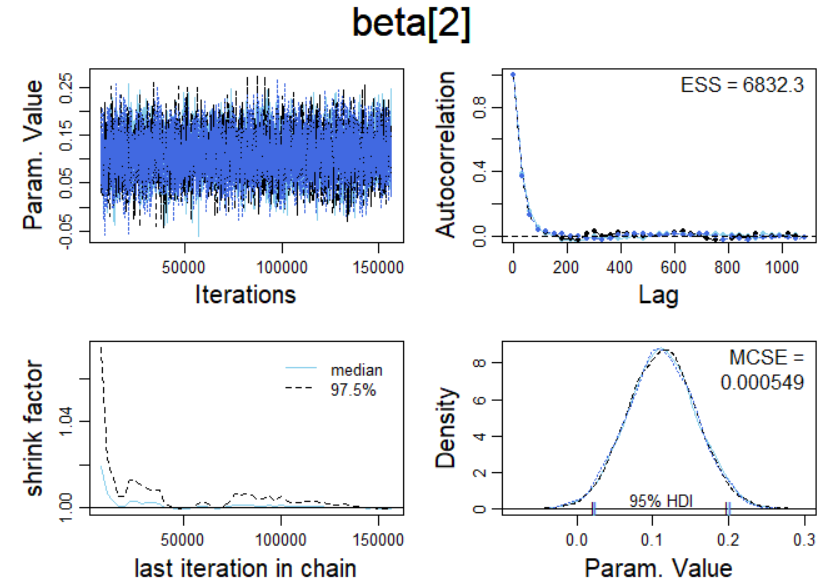


Figure 15: Diagnostics for beta[2]

* Chains converge very well
* Shrink factor < 1.2
* High ESS
* MCSE close to 0
* HDIs overlap
* A few autocorrelations

**Bathrooms β3 :**

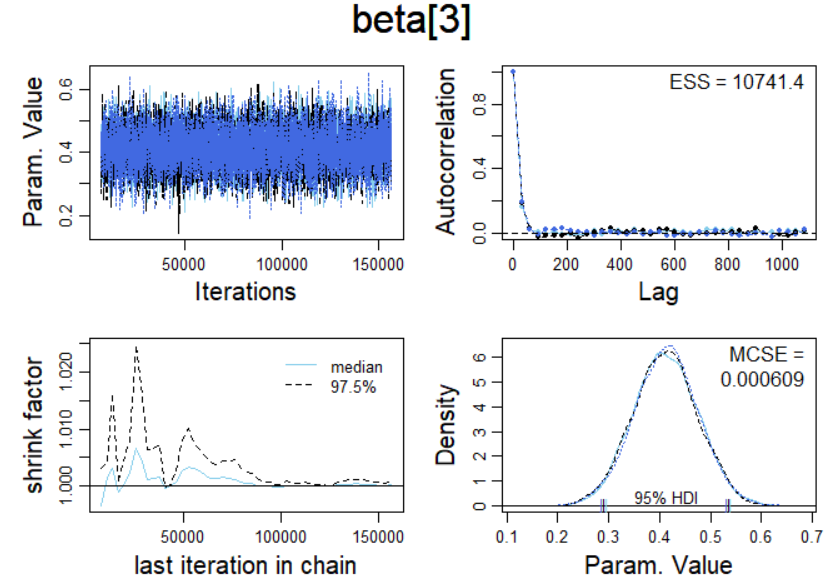


Figure 16: Diagnostics for beta[3]

* Chains converge very well
* Shrink factor < 1.2
* High ESS
* MCSE close to 0
* HDIs overlap
* A few autocorrelations

**Carparks β4 :**

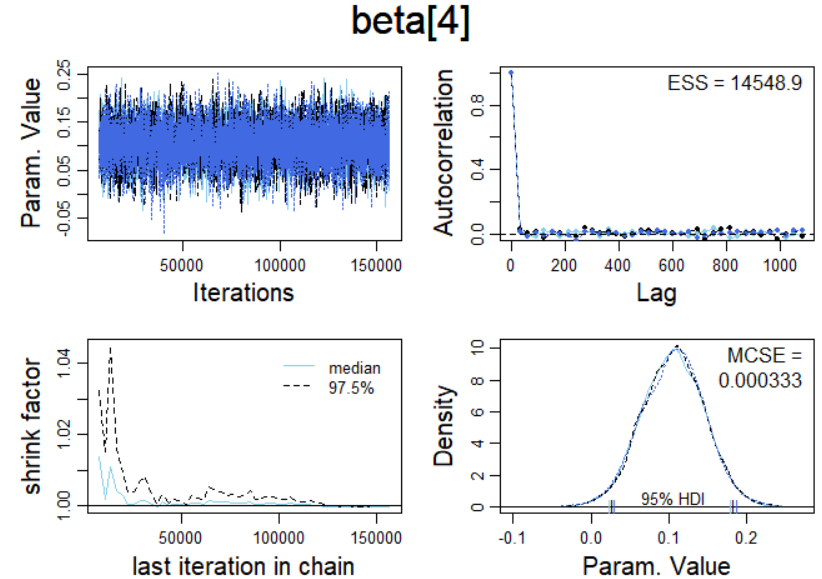


Figure 17: Diagnostics for beta[4]

* Chains converge very well
* Shrink factor < 1.2
* High ESS
* MCSE close to 0
* HDIs overlap
* No autocorrelations

**Type β5 :**

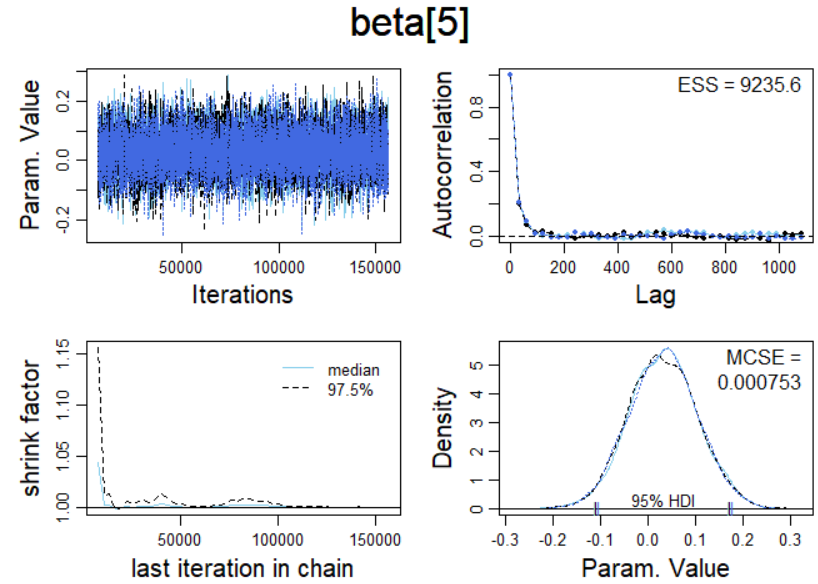


Figure 18: Diagnostics for beta[5]

* Chains converge very well
* Shrink factor < 1.2
* High ESS
* MCSE close to 0
* HDIs overlap
* A few autocorrelations

**Tau σ :**

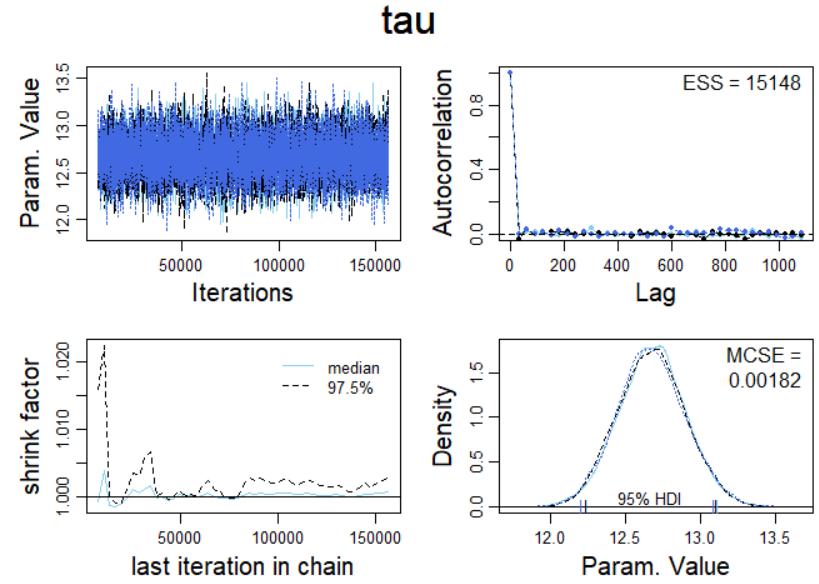


Figure 19: Diagnostics for tau

* Chains converge very well
* Shrink factor < 1.2
* High ESS
* MCSE close to 0
* HDIs almost overlap
* No autocorrelations

## Exploring Posterior Distribution

The 5th run was accepted as the best representation for the data provided. The posteriors distributions were obtained for all parameters and discussed in this section.

### Posterior for Predictors

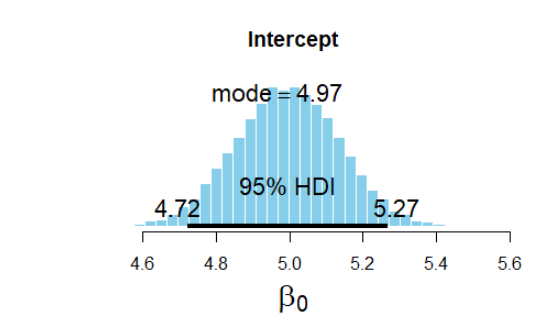


Figure 20: Posterior distribution for intercept

* β0 : The probability of intercept being between (4.72,5.27) was 0.95. The point estimate was computed as 4.97, which represents the base property price for a setup having all predictors = 0

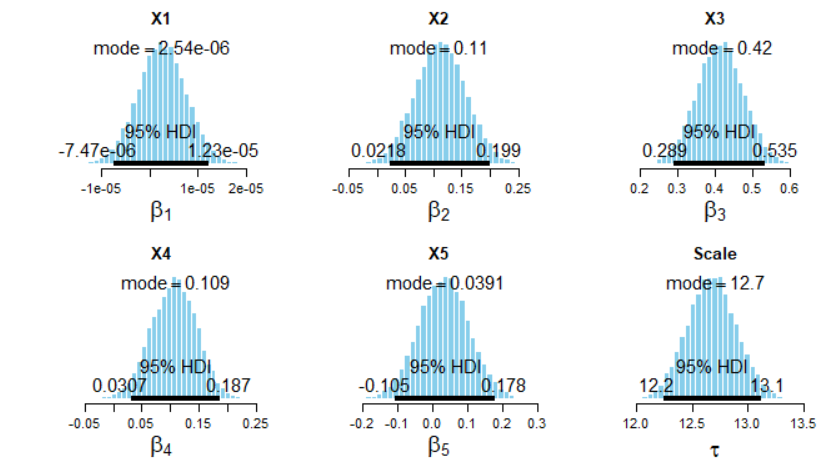


Figure 21: Posterior distribution for predictors

* β1 : The probability of area being between (-0.747x10-6, 1.23x10-5) was 0.95. This 95% HDI also includes 0 value. Hence, this parameter may be considered insignificant for further tuning and prediction. When considered significant, the point estimate was computed as 2.54x10-6, which represents the increase in property price for a unit increase in area of property.
* Β2 : The probability of number of bedrooms being between (0.0218, 0.199) was 0.95. This 95% HDI does not include 0 value. Hence, this parameter coefficient was considered significant for further tuning and prediction. The point estimate was computed as 0.11, which represents the increase in property price for a unit increase in number of bedrooms.
* Β3 : The probability of number of bathrooms being between (0.289, 0.535) was 0.95. This 95% HDI does not include 0 value. Hence, this parameter coefficient was considered significant for further tuning and prediction. The point estimate was computed as 0.42, which represents the increase in property price for a unit increase in number of bathrooms.
* Β4 : The probability of number of carparks being between (0.0307, 0.187) was 0.95. This 95% HDI also includes 0 value, but it was observed near the left tail of distributions. In case of 90% HDI, 0 might be get excluded from the HDI region. Hence, this parameter was considered significant for further tuning and prediction. The point estimate was computed as 0.109, which represents the increase in property price for a unit increase in number of carparks.
* Β5 : The probability for coefficient of property type variable being between (-0.105, 0.178) was 0.95. This 95% HDI also includes 0 value. Hence, this parameter may be considered insignificant for further tuning and prediction. When considered significant, the point estimate was computed as 0.0391, which represents the difference in property price between house(0) and unit(1). If considered insignificant, the prediction model would indirectly and unknowingly assume the property to be a house.

### Posterior for Predictions

As per the 5 configurations mentioned in problem statement, the posteriors for predictions were obtained:

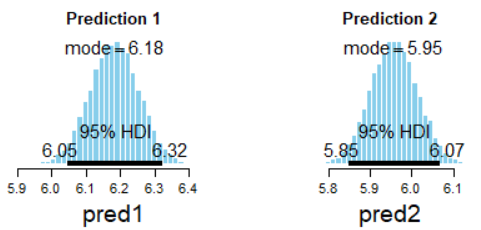


Figure 22: Posterior distribution for Prediction from 1st and 2nd configurations

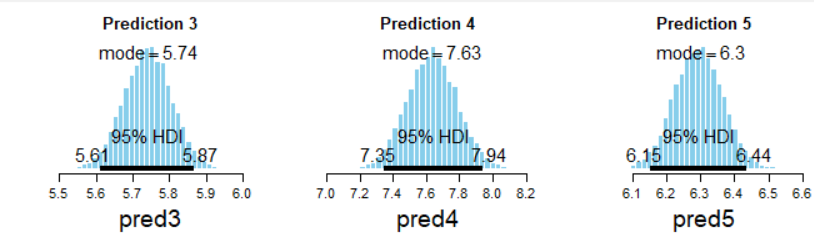


Figure 23: Posterior distribution for Prediction from 3rd, 4th and 5th configurations

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Prediction** | **Area** | **Bedrooms** | **Bathrooms** | **Carparks** | **Type** | **HDI for P(0.95)** | **Point Estimate** |
| 1 | 600 | 2 | 2 | 1 | 1 | (6.05,6.32) | 6.18 |
| 2 | 800 | 3 | 1 | 2 | 0 | (5.85,6.07) | 5.95 |
| 3 | 1500 | 2 | 1 | 1 | 0 | (5.61,5.87) | 5.74 |
| 4 | 2500 | 5 | 4 | 4 | 0 | (7.35,7.94) | 7.63 |
| 5 | 250 | 3 | 2 | 1 | 1 | (6.15,6.44) | 6.3 |

Table 6: Inferences from posterior distributions of predictions

## Predictive Check

After accepting the prediction model, a density plot was obtained to compare the observed and predicted values of response variable.

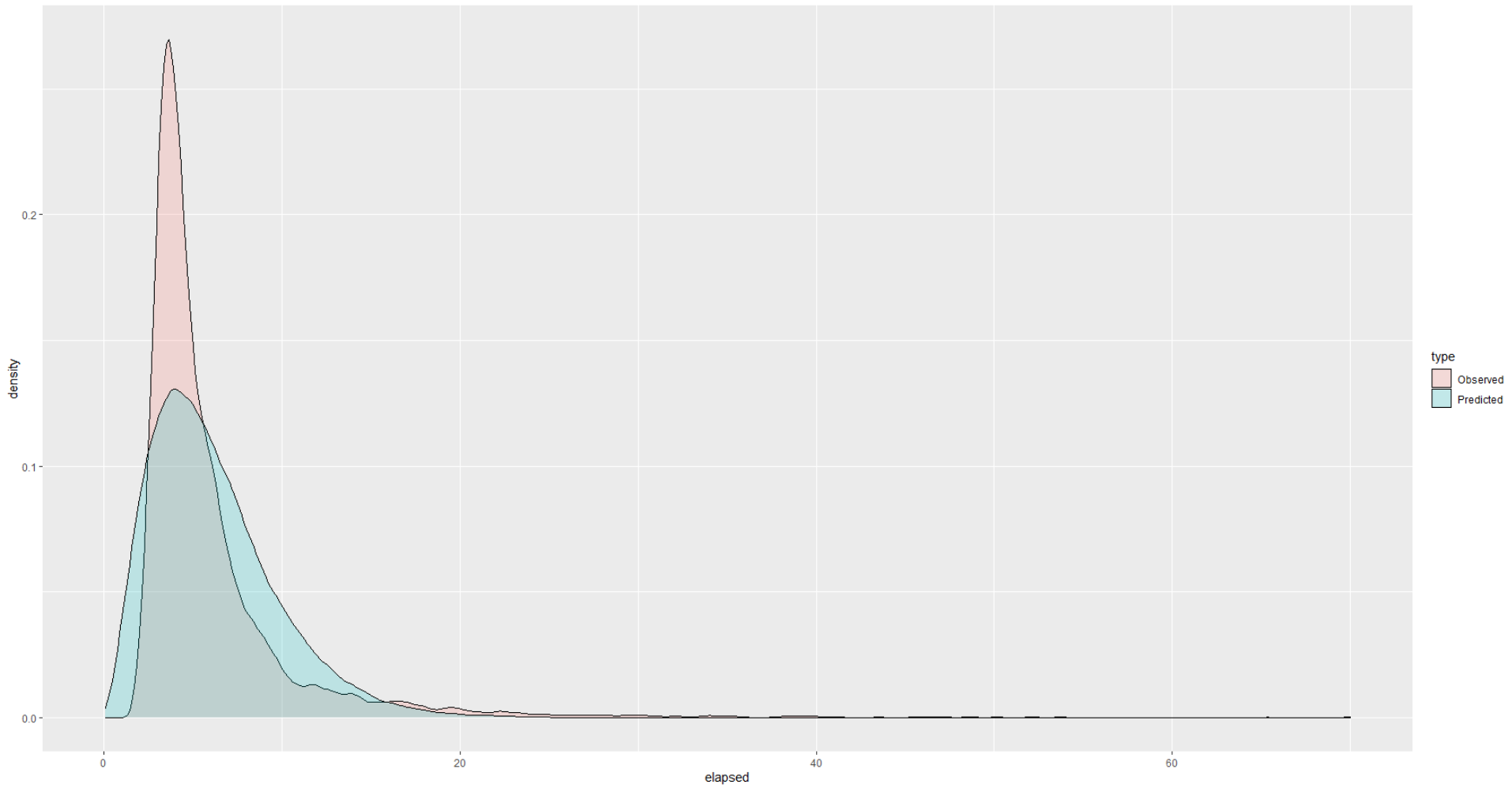


Figure 24: Density plot for observed vs. predicted values

It can be observed that an acceptable fit was achieved along both the tails of the plot, but the predicted model was unable to capture the peak density of observed values. The failure of model might be attributed to the misinterpretation of relations

# Conclusion

Based on the data and expert knowledge provided for each of the 5 predictors, a multiple linear model was successfully simulated to predict property prices in Melbourne. The best diagnostics for the MCMC model were obtained using the following parameters:

|  |  |
| --- | --- |
| Number of Chains | 3 |
| Burn-in Period | 5000 |
| Thinning | 30 |
| Number of Saved Steps | 5000 |
| Adapt Steps | 1500 |

Table 7: Chosen parameters for simulating MCMC model

With the abovementioned configuration, satisfactory diagnostics were achieved for all beta priors, except for a few autocorrelations in intercept, type of property, number of bedrooms and bathrooms.

The β prior of area and type of property were found to be insignificant but were still considered for formulating the predictive model because their point estimates were extremely small. The final predictive model is given below:

**Y =** **4.97 + 2.54x10-6 xi1 + 0.11 xi2+ 0.42 xi3 + 0.109 xi4 + 0.0391 xi5**

Using the configuration provided in problem statement, the point estimates of predictions, along with 95% probability for response variable’s values were obtained using the predictive model:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Prediction** | **Area** | **Bedrooms** | **Bathrooms** | **Carparks** | **Type** | **HDI for P(0.95)** | **Point Estimate** |
| 1 | 600 | 2 | 2 | 1 | 1 | (6.05,6.32) | 6.18 |
| 2 | 800 | 3 | 1 | 2 | 0 | (5.85,6.07) | 5.95 |
| 3 | 1500 | 2 | 1 | 1 | 0 | (5.61,5.87) | 5.74 |
| 4 | 2500 | 5 | 4 | 4 | 0 | (7.35,7.94) | 7.63 |
| 5 | 250 | 3 | 2 | 1 | 1 | (6.15,6.44) | 6.3 |

Table 8: Point estimates and 95% HDI for predictions

The goodness-of-fit of predictive model was examined by plotting a density plot of observed vs. predicted values. It was observed that the model failed to capture almost 50% of the high-density peak of observed values. This issue could be attributed to the following:

* Presence of possible outliers in data, which led to distortion of trend and subsequent misinterpretation of relationship between response and predictors
* Correlation between predictors, even though small, might have led to issues in modelling
* Considering beta priors of area and property type as significant, which could be handled by completely dropping them from the model
* Presence of a few autocorrelations in some beta priors’ diagnostics

# Appendix

graphics.off() # This closes all of R's graphics windows.

rm(list=ls()) # Careful! This clears all of R's memory!

library(ggplot2)

library(ggpubr)

library(ks)

library(rjags)

library(runjags)

source("DBDA2E-utilities.R")

#===============PRELIMINARY FUNCTIONS FOR POSTERIOR INFERENCES====================

smryMCMC\_HD = function( codaSamples , compVal = NULL, saveName=NULL) {

summaryInfo = NULL

mcmcMat = as.matrix(codaSamples,chains=TRUE)

paramName = colnames(mcmcMat)

for ( pName in paramName ) {

if (pName %in% colnames(compVal)){

if (!is.na(compVal[pName])) {

summaryInfo = rbind( summaryInfo , summarizePost( paramSampleVec = mcmcMat[,pName] ,

compVal = as.numeric(compVal[pName]) ))

}

else {

summaryInfo = rbind( summaryInfo , summarizePost( paramSampleVec = mcmcMat[,pName] ) )

}

} else {

summaryInfo = rbind( summaryInfo , summarizePost( paramSampleVec = mcmcMat[,pName] ) )

}

}

rownames(summaryInfo) = paramName

# summaryInfo = rbind( summaryInfo ,

# "tau" = summarizePost( mcmcMat[,"tau"] ) )

if ( !is.null(saveName) ) {

write.csv( summaryInfo , file=paste(saveName,"SummaryInfo.csv",sep="") )

}

return( summaryInfo )

}

#===============================================================================

plotMCMC\_HD = function( codaSamples , data , xName="x" , yName="y" ,

showCurve=FALSE , pairsPlot=FALSE , compVal = NULL,

saveName=NULL , saveType="jpg" ) {

# showCurve is TRUE or FALSE and indicates whether the posterior should

# be displayed as a histogram (by default) or by an approximate curve.

# pairsPlot is TRUE or FALSE and indicates whether scatterplots of pairs

# of parameters should be displayed.

#-----------------------------------------------------------------------------

y = data[,yName]

x = as.matrix(data[,xName])

mcmcMat = as.matrix(codaSamples,chains=TRUE)

chainLength = NROW( mcmcMat )

zbeta0 = mcmcMat[,"zbeta0"]

zbeta = mcmcMat[,grep("^zbeta$|^zbeta\\[",colnames(mcmcMat))]

if ( ncol(x)==1 ) { zbeta = matrix( zbeta , ncol=1 ) }

zVar = mcmcMat[,"zVar"]

beta0 = mcmcMat[,"beta0"]

beta = mcmcMat[,grep("^beta$|^beta\\[",colnames(mcmcMat))]

if ( ncol(x)==1 ) { beta = matrix( beta , ncol=1 ) }

tau = mcmcMat[,"tau"]

pred1 = mcmcMat[,"pred[1]"] # Added by Demirhan

pred2 = mcmcMat[,"pred[2]"] # Added by Demirhan

pred3 = mcmcMat[,"pred[3]"] # Added by Demirhan

pred4 = mcmcMat[,"pred[4]"] # Added by Demirhan

pred5 = mcmcMat[,"pred[5]"] # Added by Demirhan

#-----------------------------------------------------------------------------

# Compute R^2 for credible parameters:

YcorX = cor( y , x ) # correlation of y with each x predictor

Rsq = zbeta %\*% matrix( YcorX , ncol=1 )

#-----------------------------------------------------------------------------

if ( pairsPlot ) {

# Plot the parameters pairwise, to see correlations:

openGraph()

nPtToPlot = 1000

plotIdx = floor(seq(1,chainLength,by=chainLength/nPtToPlot))

panel.cor = function(x, y, digits=2, prefix="", cex.cor, ...) {

usr = par("usr"); on.exit(par(usr))

par(usr = c(0, 1, 0, 1))

r = (cor(x, y))

txt = format(c(r, 0.123456789), digits=digits)[1]

txt = paste(prefix, txt, sep="")

if(missing(cex.cor)) cex.cor <- 0.8/strwidth(txt)

text(0.5, 0.5, txt, cex=1.25 ) # was cex=cex.cor\*r

}

pairs( cbind( beta0 , beta , tau )[plotIdx,] ,

labels=c( "beta[0]" ,

paste0("beta[",1:ncol(beta),"]\n",xName) ,

expression(tau) ) ,

lower.panel=panel.cor , col="skyblue" )

if ( !is.null(saveName) ) {

saveGraph( file=paste(saveName,"PostPairs",sep=""), type=saveType)

}

}

#-----------------------------------------------------------------------------

# Marginal histograms:

decideOpenGraph = function( panelCount , saveName , finished=FALSE ,

nRow=2 , nCol=3 ) {

# If finishing a set:

if ( finished==TRUE ) {

if ( !is.null(saveName) ) {

saveGraph( file=paste0(saveName,ceiling((panelCount-1)/(nRow\*nCol))),

type=saveType)

}

panelCount = 1 # re-set panelCount

return(panelCount)

} else {

# If this is first panel of a graph:

if ( ( panelCount %% (nRow\*nCol) ) == 1 ) {

# If previous graph was open, save previous one:

if ( panelCount>1 & !is.null(saveName) ) {

saveGraph( file=paste0(saveName,(panelCount%/%(nRow\*nCol))),

type=saveType)

}

# Open new graph

openGraph(width=nCol\*7.0/3,height=nRow\*2.0)

layout( matrix( 1:(nRow\*nCol) , nrow=nRow, byrow=TRUE ) )

par( mar=c(4,4,2.5,0.5) , mgp=c(2.5,0.7,0) )

}

# Increment and return panel count:

panelCount = panelCount+1

return(panelCount)

}

}

# Original scale:

panelCount = 1

if (!is.na(compVal["beta0"])){

panelCount = decideOpenGraph( panelCount , saveName=paste0(saveName,"PostMarg") )

histInfo = plotPost( beta0 , cex.lab = 1.75 , showCurve=showCurve ,

xlab=bquote(beta[0]) , main="Intercept", compVal = as.numeric(compVal["beta0"] ))

} else {

histInfo = plotPost( beta0 , cex.lab = 1.75 , showCurve=showCurve ,

xlab=bquote(beta[0]) , main="Intercept")

}

for ( bIdx in 1:ncol(beta) ) {

panelCount = decideOpenGraph( panelCount , saveName=paste0(saveName,"PostMarg") )

if (!is.na(compVal[paste0("beta[",bIdx,"]")])) {

histInfo = plotPost( beta[,bIdx] , cex.lab = 1.75 , showCurve=showCurve ,

xlab=bquote(beta[.(bIdx)]) , main=xName[bIdx],

compVal = as.numeric(compVal[paste0("beta[",bIdx,"]")]))

} else{

histInfo = plotPost( beta[,bIdx] , cex.lab = 1.75 , showCurve=showCurve ,

xlab=bquote(beta[.(bIdx)]) , main=xName[bIdx])

}

}

panelCount = decideOpenGraph( panelCount , saveName=paste0(saveName,"PostMarg") )

histInfo = plotPost( tau , cex.lab = 1.75 , showCurve=showCurve ,

xlab=bquote(tau) , main=paste("Scale") )

panelCount = decideOpenGraph( panelCount , saveName=paste0(saveName,"PostMarg") )

histInfo = plotPost( Rsq , cex.lab = 1.75 , showCurve=showCurve ,

xlab=bquote(R^2) , main=paste("Prop Var Accntd") )

panelCount = decideOpenGraph( panelCount , saveName=paste0(saveName,"PostMarg") )

histInfo = plotPost( pred1 , cex.lab = 1.75 , showCurve=showCurve ,

xlab="pred1" , main="Prediction 1" ) # Added by Demirhan

panelCount = decideOpenGraph( panelCount , finished=TRUE , saveName=paste0(saveName,"PostMarg") )

histInfo = plotPost( pred2 , cex.lab = 1.75 , showCurve=showCurve ,

xlab="pred2" , main="Prediction 2" ) # Added by Demirhan

panelCount = decideOpenGraph( panelCount , saveName=paste0(saveName,"PostMarg") )

histInfo = plotPost( pred3 , cex.lab = 1.75 , showCurve=showCurve ,

xlab="pred3" , main="Prediction 3" ) # Added by Demirhan

panelCount = decideOpenGraph( panelCount , saveName=paste0(saveName,"PostMarg") )

histInfo = plotPost( pred4 , cex.lab = 1.75 , showCurve=showCurve ,

xlab="pred4" , main="Prediction 4" ) # Added by Demirhan

panelCount = decideOpenGraph( panelCount , saveName=paste0(saveName,"PostMarg") )

histInfo = plotPost( pred5 , cex.lab = 1.75 , showCurve=showCurve ,

xlab="pred5" , main="Prediction 5" ) # Added by Demirhan

# Standardized scale:

panelCount = 1

panelCount = decideOpenGraph( panelCount , saveName=paste0(saveName,"PostMargZ") )

histInfo = plotPost( zbeta0 , cex.lab = 1.75 , showCurve=showCurve ,

xlab=bquote(z\*beta[0]) , main="Intercept" )

for ( bIdx in 1:ncol(beta) ) {

panelCount = decideOpenGraph( panelCount , saveName=paste0(saveName,"PostMargZ") )

histInfo = plotPost( zbeta[,bIdx] , cex.lab = 1.75 , showCurve=showCurve ,

xlab=bquote(z\*beta[.(bIdx)]) , main=xName[bIdx] )

}

panelCount = decideOpenGraph( panelCount , saveName=paste0(saveName,"PostMargZ") )

histInfo = plotPost( zVar , cex.lab = 1.75 , showCurve=showCurve ,

xlab=bquote(z\*tau) , main=paste("Scale") )

panelCount = decideOpenGraph( panelCount , saveName=paste0(saveName,"PostMargZ") )

histInfo = plotPost( Rsq , cex.lab = 1.75 , showCurve=showCurve ,

xlab=bquote(R^2) , main=paste("Prop Var Accntd") )

panelCount = decideOpenGraph( panelCount , finished=TRUE , saveName=paste0(saveName,"PostMargZ") )

#-----------------------------------------------------------------------------

}

#===============PRELIMINARY FUNCTIONS FOR POSTERIOR INFERENCES====================

myData <- read.csv("Assignment2PropertyPrices.csv")

head(myData)

summary(myData)

# Scatter plots

p1 <- ggplot(myData, aes(x=X1, y=Y)) +

geom\_point() +

xlab("Area") +

ylab("Property Price")

p2 <- ggplot(myData, aes(x=X2, y=Y)) +

geom\_point() +

xlab("Bedrooms") +

ylab("Property Price")

p3 <- ggplot(myData, aes(x=X3, y=Y)) +

geom\_point() +

xlab("Bathrooms") +

ylab("Property Price")

p4 <- ggplot(myData, aes(x=X4, y=Y)) +

geom\_point() +

xlab("Car Park ") +

ylab("Property Price")

p5 <- ggplot(myData, aes(x=X5, y=Y)) +

geom\_point() +

xlab("Type ") +

ylab("Property Price")

p1

figure <- ggarrange(p2, p3, p4, p5, nrow = 2, ncol = 2)

figure

# Histogram

hist(myData$Y, main= " Histogram of the dependent variable", xlab = "Property Price")

# Kernel density estimation

plot(kde(myData$Y), xlab = "Property Price") # with default settings

# THE DATA.

y = myData[,"Y"]

x = as.matrix(myData[,c("X1","X2","X3","X4","X5")])

# Some more descriptives

cat("\nCORRELATION MATRIX OF PREDICTORS:\n ")

show( round(cor(x),3) )

cat("\n")

xPred = array(NA, dim = c(5,5))

xPred[1,] = c(600, 2, 2, 1, 1)

xPred[2,] = c(800, 3, 1, 2, 0)

xPred[3,] = c(1500, 2, 1, 1, 0)

xPred[4,] = c(2500, 5, 4, 4, 0)

xPred[5,] = c(250, 3, 2, 1, 1)

# Specify the data in a list, for later shipment to JAGS:

dataList <- list(

x = x ,

y = y ,

xPred = xPred ,

Nx = dim(x)[2] ,

Ntotal = dim(x)[1]

)

# First run without initials!

initsList <- list(

zbeta0 = 2000,

zbeta = c(100, 1, 1, 0.5,1),

Var = 12000000

)

# WE WILL RUN THE MODEL WITH SCALING!

# THE MODEL.

modelString = "

# Standardize the data:

data {

ysd <- sd(y)

for ( i in 1:Ntotal ) {

zy[i] <- y[i] / ysd

}

for ( j in 1:Nx ) {

xsd[j] <- sd(x[,j])

for ( i in 1:Ntotal ) {

zx[i,j] <- x[i,j] / xsd[j]

}

}

}

# Specify the model for scaled data:

model {

for ( i in 1:Ntotal ) {

zy[i] ~ dgamma( (mu[i]^2)/zVar , mu[i]/zVar )

mu[i] <- zbeta0 + sum( zbeta[1:Nx] \* zx[i,1:Nx] )

}

# Priors on standardized scale:

zbeta0 ~ dnorm( 0 , 1/2^2 ) # 1/ variance for normal distribution

zbeta[1] ~ dnorm( (90/100000)/xsd[1] , 1/(0.1/xsd[1]^2) ) # 1/ variance for normal distribution

zbeta[2] ~ dnorm( (100000/100000)/xsd[2] , 1/(4/xsd[2]^2) ) # 1/ variance for normal distribution

zbeta[3] ~ dnorm( 0 , 1/4) # 1/ variance for normal distribution

zbeta[4] ~ dnorm( (120000/100000)/xsd[4] , 1/(1/xsd[4]^2) ) # 1/ variance for normal distribution

zbeta[5] ~ dnorm( (150000/100000)/xsd[5] , 1/(0.1/xsd[5]^2) ) # 1/ variance for normal distribution

zVar ~ dgamma( 0.01 , 0.01 )

# Transform to original scale:

beta[1:Nx] <- ( zbeta[1:Nx] / xsd[1:Nx] ) \* ysd

beta0 <- zbeta0\*ysd

tau <- zVar \* (ysd)^2

# Compute predictions at every step of the MCMC

for ( i in 1:5){

pred[i] <- beta0 + beta[1] \* xPred[i,1] + beta[2] \* xPred[i,2] + beta[3] \* xPred[i,3] + beta[4] \* xPred[i,4] + beta[5] \* xPred[i,5]

}

# pred[1] <- beta0 + beta[1] \* xPred[1,1] + beta[2] \* xPred[1,2] + beta[3] \* xPred[1,3] + beta[4] \* xPred[1,4]

# pred[2] <- beta0 + beta[1] \* xPred[2,1] + beta[2] \* xPred[2,2] + beta[3] \* xPred[2,3] + beta[4] \* xPred[2,4]

}

" # close quote for modelString

# Write out modelString to a text file

writeLines( modelString , con="TEMPmodel.txt" )

parameters = c( "zbeta0" , "zbeta" , "beta0" , "beta" , "tau", "zVar") # Here beta is a vector!

adaptSteps = 1500 # Number of steps to "tune" the samplers

burnInSteps = 5000

nChains = 3

thinSteps = 30 # First run for 3

numSavedSteps = 5000

nIter = ceiling( ( numSavedSteps \* thinSteps ) / nChains )

# Parallel run

runJagsOut <- run.jags( method="parallel" ,

model="TEMPmodel.txt" ,

monitor=c( "zbeta0" , "zbeta" , "beta0" , "beta" , "tau", "zVar", "pred") ,

data=dataList ,

n.chains=nChains ,

adapt=adaptSteps ,

burnin=burnInSteps ,

sample=numSavedSteps ,

thin=thinSteps , summarise=FALSE , plots=FALSE )

codaSamples = as.mcmc.list( runJagsOut )

#save.image(file="rEnvironment-5.RData")

#load(file="rEnvironment-5.RData") # Load the results with 124,000 iterations

diagMCMC( codaSamples , parName="beta0" )

diagMCMC( codaSamples , parName="beta[1]" )

diagMCMC( codaSamples , parName="beta[2]" )

diagMCMC( codaSamples , parName="beta[3]" )

diagMCMC( codaSamples , parName="beta[4]" )

diagMCMC( codaSamples , parName="beta[5]" )

diagMCMC( codaSamples , parName="tau" )

compVal <- data.frame("beta0" = NA, "beta[1]" = NA, "beta[2]" = NA, "beta[3]" = NA, "beta[4]" = NA, "beta[5]" = NA, "tau" = NA , check.names=FALSE)

summaryInfo <- smryMCMC\_HD( codaSamples = codaSamples , compVal = compVal )

print(summaryInfo)

plotMCMC\_HD( codaSamples = codaSamples , data = myData, xName=c("X1","X2","X3","X4", "X5") ,

yName="Y", compVal = compVal)

# ============ Predictive check ============

coefficients <- summaryInfo[8:13,3] # Get the model coefficients out

Variance <- summaryInfo[14,3] # Get the variance out

# Since we imposed the regression model on the mean of the gamma likelihood,

# we use the model (X\*beta) to generate the mean of gamma population for each

# observed x vector.

meanGamma <- as.matrix(cbind(rep(1,nrow(x)), x)) %\*% as.vector(coefficients)

# Generate random data from the posterior distribution. Here I take the

# reparameterisation back to alpha and beta.

randomData <- rgamma(n= 100000,shape=meanGamma^2/Variance, rate = meanGamma/Variance)

# Display the density plot of observed data and posterior distribution:

predicted <- data.frame(elapsed = randomData)

observed <- data.frame(elapsed = y)

predicted$type <- "Predicted"

observed$type <- "Observed"

dataPred <- rbind(predicted, observed)

ggplot(dataPred, aes(elapsed, fill = type)) + geom\_density(alpha = 0.2)