Illustration of the dataset

Quarterly car sales are estimated from the same sample used for the Monthly car sales to estimate preliminary and final car sales in Quebec .The data is an open source data from 2000 to 2016 and downloaded from this website: https://data.worldbank.org/

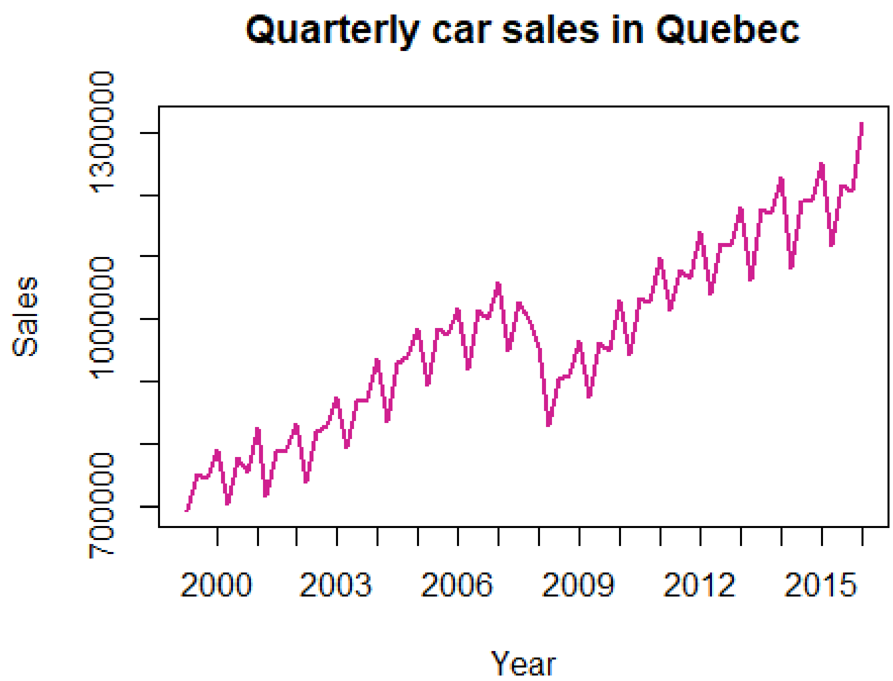
The table below shows the first 12 data (3 years), which are sorted by year and quarter.

```
## read data
data = read.csv("QUARTERLY CAR SALES.csv")
df = data.frame(data)
print(head(df,12))
```

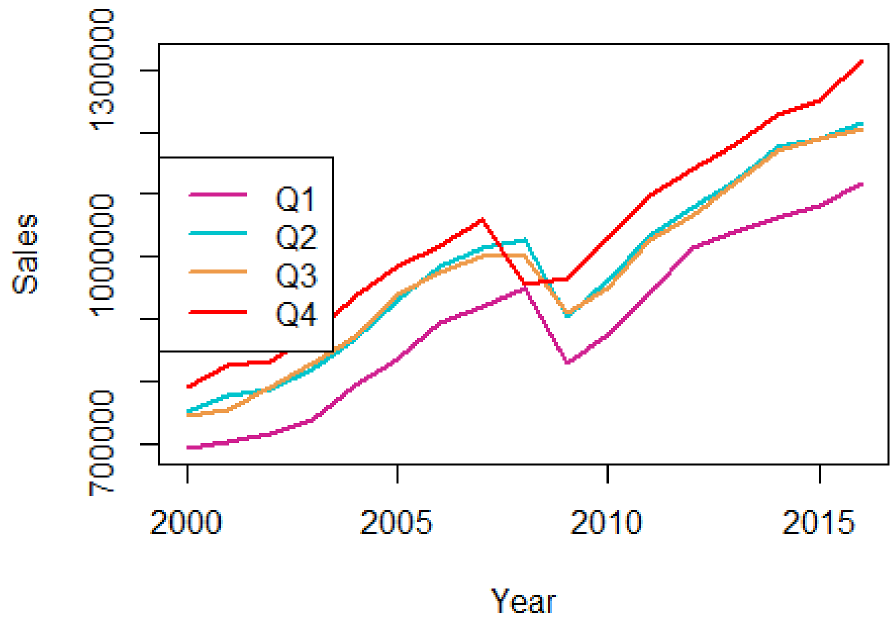
```
##    Quarter Year  Sales
## 1       Q1 2000 694513
## 2       Q2 2000 751698
## 3       Q3 2000 745556
## 4       Q4 2000 791509
## 5       Q1 2001 704201
## 6       Q2 2001 778035
## 7       Q3 2001 754174
## 8       Q4 2001 825858
## 9       Q1 2002 715763
## 10      Q2 2002 789424
## 11      Q3 2002 790861
## 12      Q4 2002 832504
```

The chart shows that sales increased significantly between the first(Q1) and second quarters(Q2), but decreased between the second and third(Q3) quarters. It has also increased again from the third quarter to the fourth(Q4) quarter. As we can see, during the first six years between 2000 and 2006, sales increased in the same way, but during 2007 and 2008, they had a significant decrease in all quarters, and after that, they increased again as in the previous trend.

```
## plot data

plot(1:length(data$Quarter), y=data$Sales, type='l', col='violetred',
     lwd = 2, ylab = 'Sales', xlab = 'Year', main = 'Quarterly car sales in
Quebec', xaxt="n")
axis(1, at = seq(4, length(data$Quarter), by = 4), labels = seq(2000,2016), )
```




```
quarters = unique(df$Quarter)
cols =c('violetred','turquoise3','tan2','red')
plot(x=2000:2016, xlim=c(2000,2016), ylim=c(min(df$Sales),max(df$Sales)),
     ylab = 'Sales', xlab = 'Year')
c = 1
for(i in quarters){
  lines(x=2000:2016, df$Sales[df$Quarter == i], col=cols[c], lwd=2)
  c = c + 1}
legend('left',legend = quarters, col = cols, lwd = 2)
```



Preprocessed Data

In this section, data preprocessing is done before use in models. To do this, the dummy_cols function is used to convert quarters to one-hot code. One-hot encoding is the process of converting a categorical variable with multiple categories into multiple variables, each with a value of 1 or 0.

```
# create dummy data for model
quarters = unique(df$Quarter)
data$Year = data$Year - 1999
data$Sales = data$Sales/10000
df = cbind(data$Year, data$Sales, dummy_cols(data$Quarter))
colnames(df) = c('Year','Sales','Quarter',quarters)
processed_df =
list('Sales'=df$Sales,'Year'=df$Year,'Q1'=df$`Q1`,'Q2'=df$`Q2`,
             'Q3'=df$`Q3`,'Q4'=df$`Q4`,N = length(df$Sales))
```

First Model

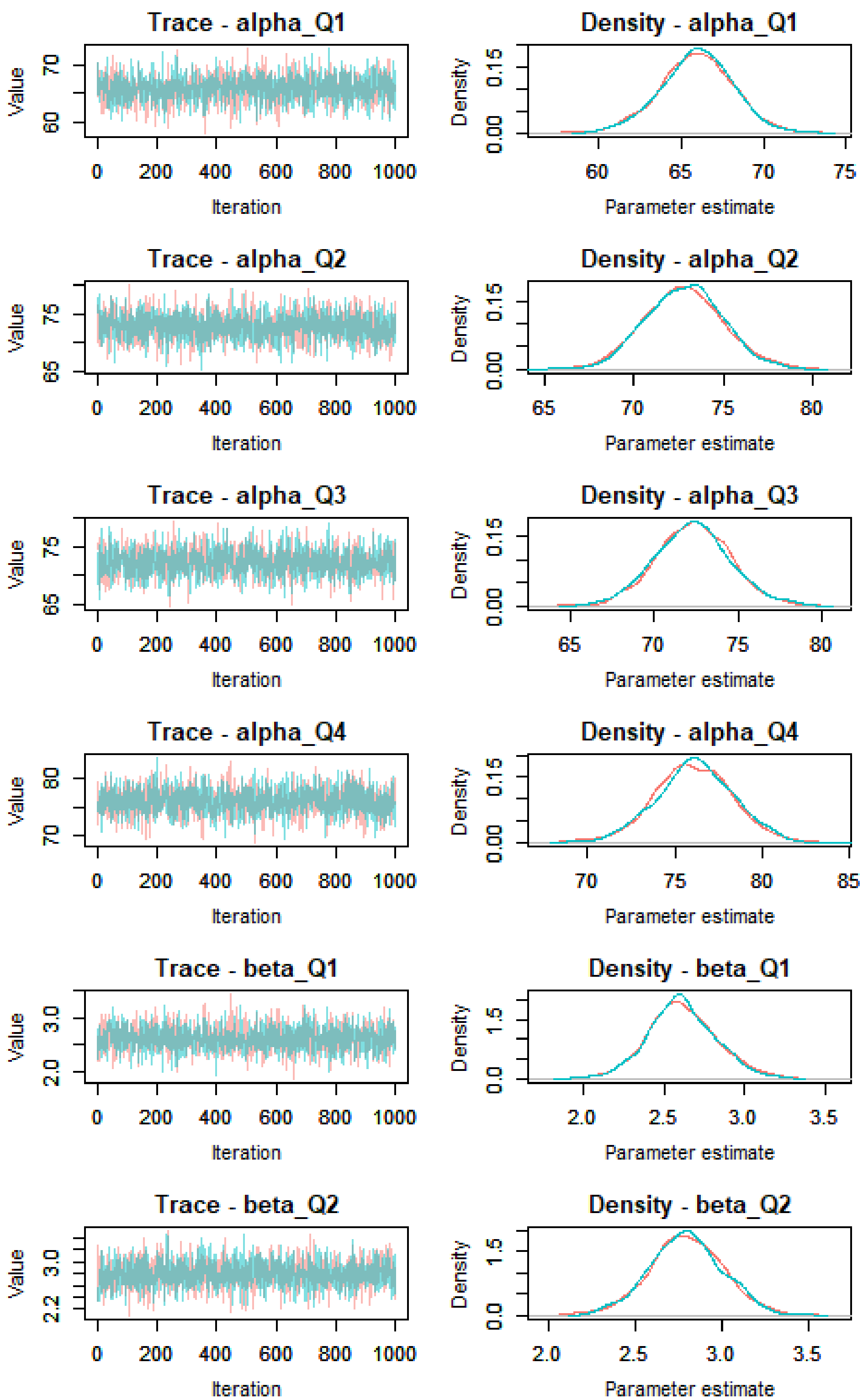
Since the data has collected based on quarters of the year, so it can be concluded that we will have four linear regression models, which by combining these four models, a single model can be introduced as follows. α refers to intercept of the year and β refers to the slope of the year.

```
first_model = function(){

      # likelihood
    for( i in 1 : N) {
        Sales[i] ~ dnorm(mu[i],tau2)
        mu[i] <-
          (alpha_Q1 + beta_Q1 * Year[i])*Q1[i] +(alpha_Q2 + beta_Q2 *
Year[i])*Q2[i] +
          (alpha_Q3 + beta_Q3 * Year[i])*Q3[i] +(alpha_Q4 + beta_Q4 *
Year[i])*Q4[i]
    }
      # priors
  alpha_Q1 ~ dunif(0,300);alpha_Q2 ~ dunif(0,300)
  alpha_Q3 ~ dunif(0,300);alpha_Q4 ~ dunif(0,300)
  beta_Q1 ~ dunif(-30,30);beta_Q2 ~ dunif(-30,30)
  beta_Q3 ~ dunif(-30,30);beta_Q4 ~ dunif(-30,30)
  tau2 ~ dgamma(1,0.5)
}
saved_modelI <- write.jags.model(first_model, filename = 'first_model.txt')
first_params = c( "alpha_Q1",  "alpha_Q2",  "alpha_Q3", "alpha_Q4",
"beta_Q1",  "beta_Q2",
   "beta_Q3",  "beta_Q4","tau2")
fit_first_model <- jags(data = processed_df, parameters.to.save =
first_params,
                        model.file =saved_modelI, n.chains = 2, n.iter =
20000,
                        n.burnin = 5000)

## module glm loaded

## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 68
##    Unobserved stochastic nodes: 9
##    Total graph size: 764
##
## Initializing model

fit_first_model

## Inference for Bugs model at
"C:\Users\msh\AppData\Local\Temp\RtmpUbcgRh/first_model.txt", fit using jags,
##  2 chains, each with 20000 iterations (first 5000 discarded), n.thin = 15
##  n.sims = 2000 iterations saved
##           mu.vect sd.vect    2.5%     25%     50%     75%   97.5%  Rhat
n.eff
## alpha_Q1  65.949   2.184  61.552  64.521  65.959  67.404  70.205 1.001
2000
## alpha_Q2  72.802   2.161  68.602  71.305  72.804  74.251  77.168 1.001
```

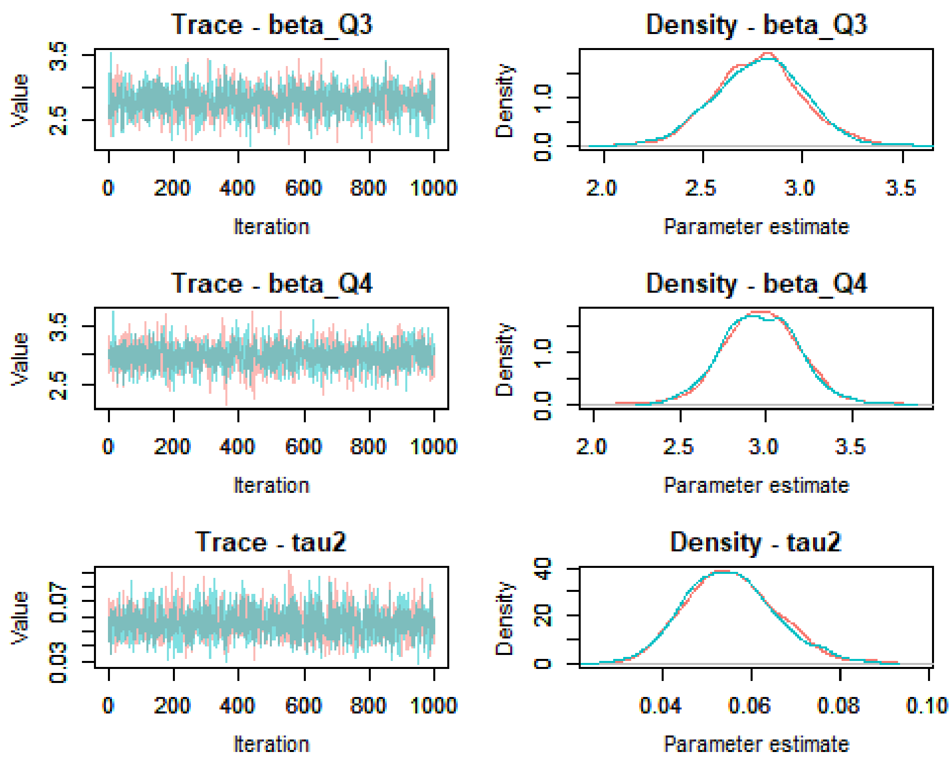
```
2000
## alpha_Q3  72.304   2.225  67.938  70.845  72.293  73.762  76.851 1.001
2000
## alpha_Q4  76.076   2.174  71.722  74.660  76.068  77.549  80.292 1.002
990
## beta_Q1    2.613   0.211   2.212   2.473   2.603   2.745   3.053 1.001
2000
## beta_Q2    2.794   0.210   2.366   2.655   2.792   2.931   3.208 1.001
2000
## beta_Q3    2.781   0.217   2.360   2.636   2.787   2.921   3.208 1.001
2000
## beta_Q4    2.973   0.215   2.551   2.827   2.968   3.119   3.388 1.001
2000
## tau2       0.055   0.010   0.037   0.048   0.055   0.062   0.076 1.001
1600
## deviance 392.789   4.518 386.235 389.519 392.112 395.227 403.483 1.001
2000
##
## For each parameter, n.eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor (at convergence, Rhat=1).
##
## DIC info (using the rule, pD = var(deviance)/2)
## pD = 10.2 and DIC = 403.0
## DIC is an estimate of expected predictive error (lower deviance is
better).
```

Convergence diagnostics

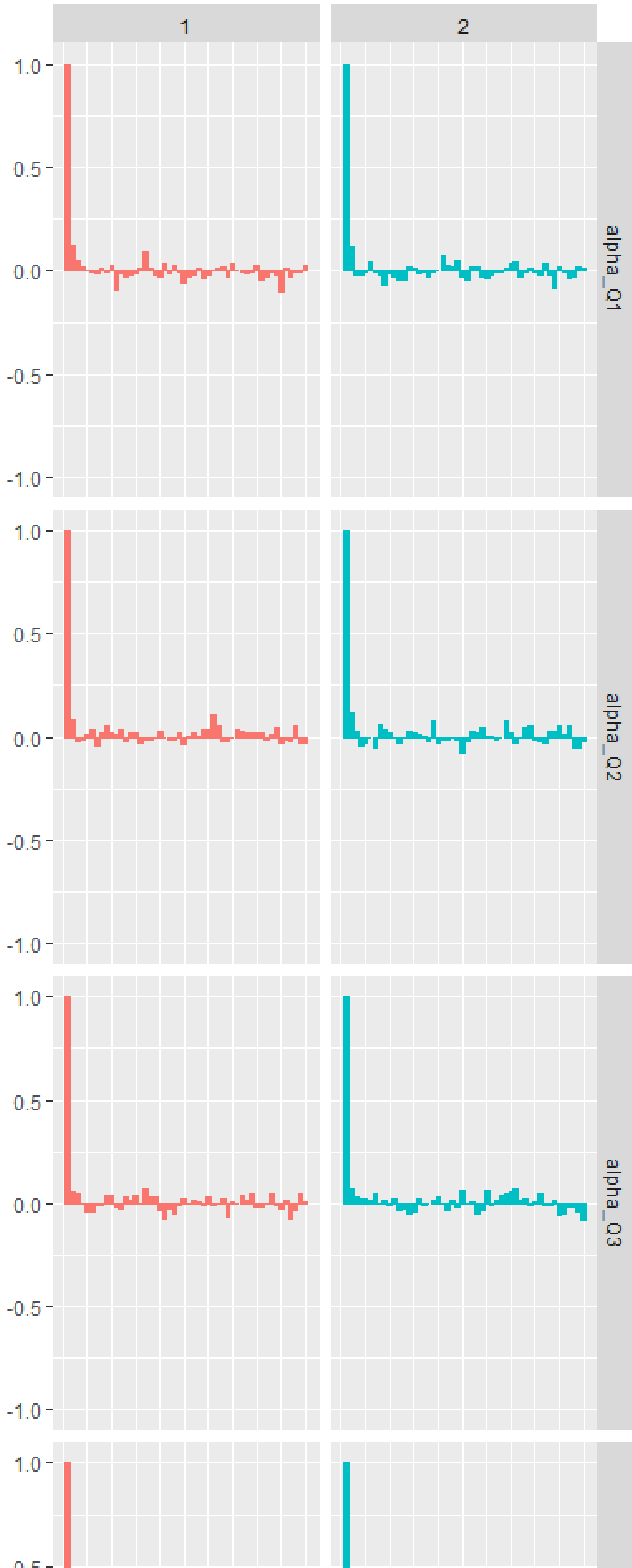
```
mcmclistI=as.mcmc(fit_first_model)
MCMCtrace(mcmclistI,first_params,ISB = FALSE,
          exact = TRUE,
          iter = 4000,
          ind = TRUE,
          pdf = FALSE)
```



Trace - alpha_Q1
Value
Iteration
Density - alpha_Q1
Density
Parameter estimate
Trace - alpha_Q2
Value
Iteration
Density - alpha_Q2
Density
Parameter estimate
Trace - alpha_Q3
Value
Iteration
Density - alpha_Q3
Density
Parameter estimate
Trace - alpha_Q4
Value
Iteration
Density - alpha_Q4
Density
Parameter estimate
Trace - beta_Q1
Value
Iteration
Density - beta_Q1
Density
Parameter estimate
Trace - beta_Q2
Value
Iteration
Density - beta_Q2
Density
Parameter estimate




```
ggs_autocorrelation(ggs(mcmclistI))
```


1
2
1.0
0.5
0.0
-0.5
-1.0
alpha_Q1
alpha_Q2
alpha_Q3

We can do summary() of an mcmc object to get summary statistics for the posterior.The results give the posterior means, posterior standard deviations,and posterior quantiles for each variable.The “naive” standard error is the standard error of the mean,which captures simulation error of the mean rather than posterior uncertainty.

The time-series standard error adjusts the “naive” standard error for autocorrelation.

```
summary(mcmclistI)

##
## Iterations = 5001:19986
## Thinning interval = 15
## Number of chains = 2
## Sample size per chain = 1000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##               Mean      SD  Naive SE Time-series SE
## alpha_Q1  65.9492 2.18390 0.0488336      0.0550970
## alpha_Q2  72.8020 2.16103 0.0483222      0.0534972
## alpha_Q3  72.3041 2.22504 0.0497534      0.0540893
## alpha_Q4  76.0759 2.17435 0.0486199      0.0548739
## beta_Q1    2.6135 0.21127 0.0047242      0.0052833
## beta_Q2    2.7940 0.20975 0.0046902      0.0051273
## beta_Q3    2.7809 0.21748 0.0048630      0.0053058
## beta_Q4    2.9729 0.21505 0.0048086      0.0054243
## deviance 392.7890 4.51769 0.1010186      0.1010399
## tau2       0.0554 0.01001 0.0002239      0.0002239
##
## 2. Quantiles for each variable:
##
##               2.5%       25%       50%       75%     97.5%
## alpha_Q1  61.55245  64.52133  65.95904  67.40374  70.20544
## alpha_Q2  68.60158  71.30512  72.80352  74.25134  77.16780
## alpha_Q3  67.93765  70.84470  72.29321  73.76221  76.85083
## alpha_Q4  71.72206  74.65970  76.06813  77.54854  80.29226
## beta_Q1    2.21234   2.47340   2.60313   2.74542   3.05283
## beta_Q2    2.36605   2.65515   2.79195   2.93099   3.20787
## beta_Q3    2.35976   2.63570   2.78730   2.92118   3.20829
## beta_Q4    2.55145   2.82664   2.96834   3.11873   3.38757
## deviance 386.23489 389.51901 392.11235 395.22731 403.48307
## tau2       0.03732   0.04817   0.05489   0.06171   0.07621
```

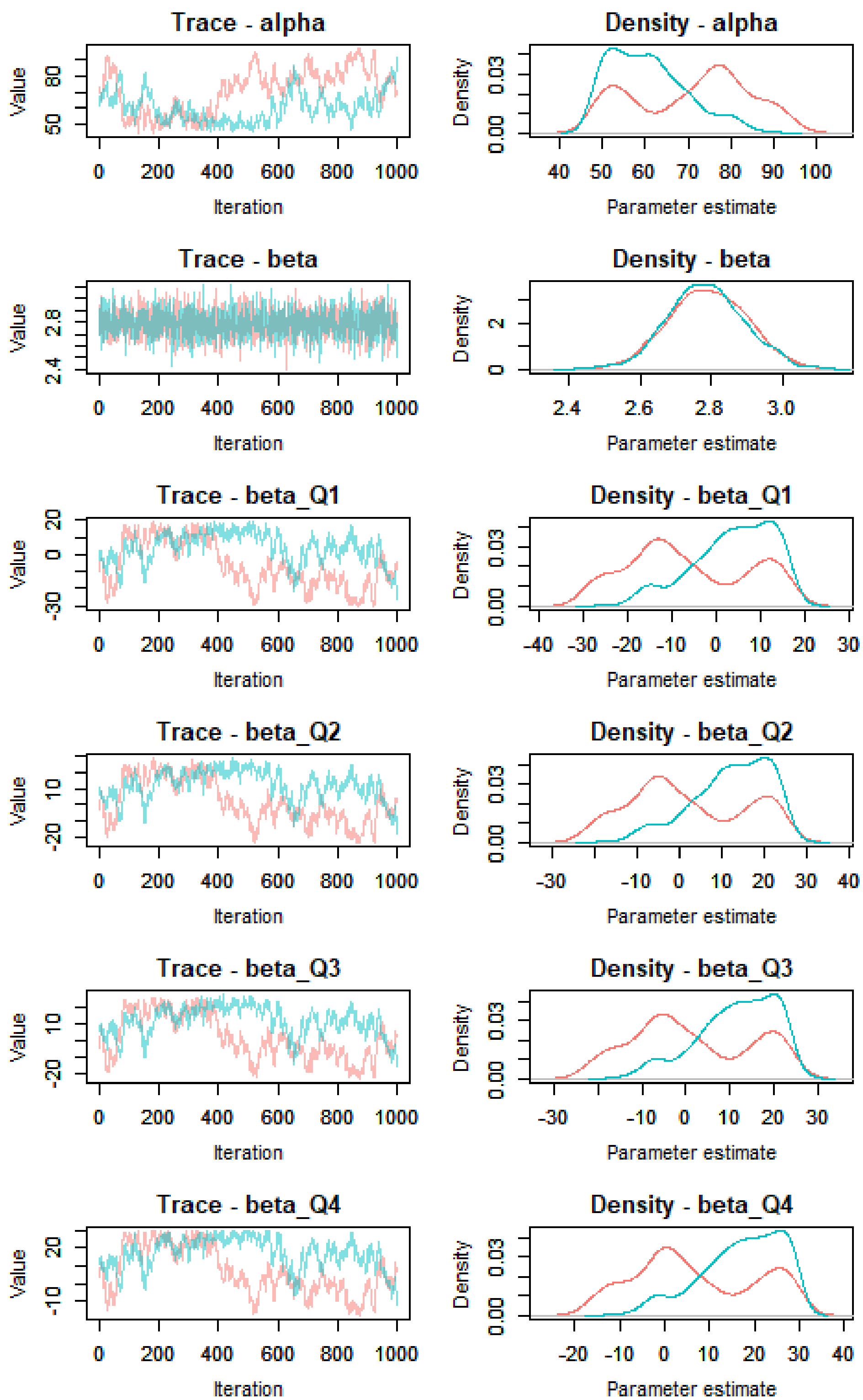
Second Model

The purpose of the second model is to reduce the DIC, the best way to do this is to reduce the number of parameters. Therefore, according to the year added to the quarterly fluctuations, a general regression model is used.

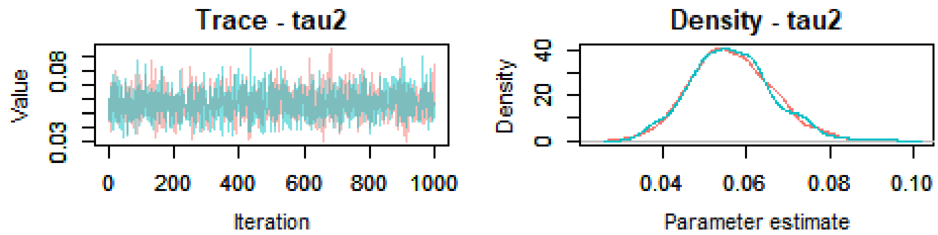
```
second_model = function(){
    for( i in 1 : N) {
        Sales[i] ~ dnorm(mu[i],tau2)
        mu[i] <- alpha + beta*Year[i] +
          beta_Q1 * Q1[i] + beta_Q2 * Q2[i] +
          beta_Q3 * Q3[i] + beta_Q4 * Q4[i]
    }
  alpha ~ dunif(0,300); beta ~ dunif(0,30)
  beta_Q1 ~ dunif(-30,30);beta_Q2 ~ dunif(-30,30)
  beta_Q3 ~ dunif(-30,30);beta_Q4 ~ dunif(-30,30)

  tau2 ~ dgamma(1,0.5)
}
saved_modelII <- write.jags.model(second_model, filename =
'second_model.txt')
second_params = c("alpha",
"beta","beta_Q1","beta_Q2","beta_Q3","beta_Q4","tau2")
fit_second_model <- jags(data = processed_df, parameters.to.save =
second_params
                         , model.file = saved_modelII,n.chains = 2, n.iter =
20000,
                         n.burnin = 5000)

## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 68
##    Unobserved stochastic nodes: 7
##    Total graph size: 515
##
## Initializing model

fit_second_model

## Inference for Bugs model at
"C:\Users\msh\AppData\Local\Temp\RtmpUbcgRh/second_model.txt", fit using
jags,
##  2 chains, each with 20000 iterations (first 5000 discarded), n.thin = 15
##  n.sims = 2000 iterations saved
##          mu.vect sd.vect    2.5%     25%     50%     75%   97.5%  Rhat
```

```
n.eff
## alpha     65.718  12.548  48.142  54.546  64.075  76.013  90.951 1.389
8
## beta       2.789   0.109   2.581   2.716   2.787   2.863   3.002 1.001
1600
## beta_Q1   -1.287  12.582 -26.913 -11.658   0.388   9.860  16.420 1.441
7
## beta_Q2    7.137  12.625 -18.219  -3.331   8.883  18.320  24.803 1.437
7
## beta_Q3    6.506  12.514 -18.690  -3.927   8.107  17.642  23.913 1.434
7
## beta_Q4   12.013  12.517 -13.363   1.612  13.778  23.204  29.282 1.452
7
## tau2       0.056   0.010   0.038   0.050   0.056   0.063   0.077 1.001
2000
## deviance 391.235   3.532 386.326 388.647 390.523 393.203 399.980 1.002
1000
##
## For each parameter, n.eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor (at convergence, Rhat=1).
##
## DIC info (using the rule, pD = var(deviance)/2)
## pD = 6.2 and DIC = 397.5
## DIC is an estimate of expected predictive error (lower deviance is
better).
```

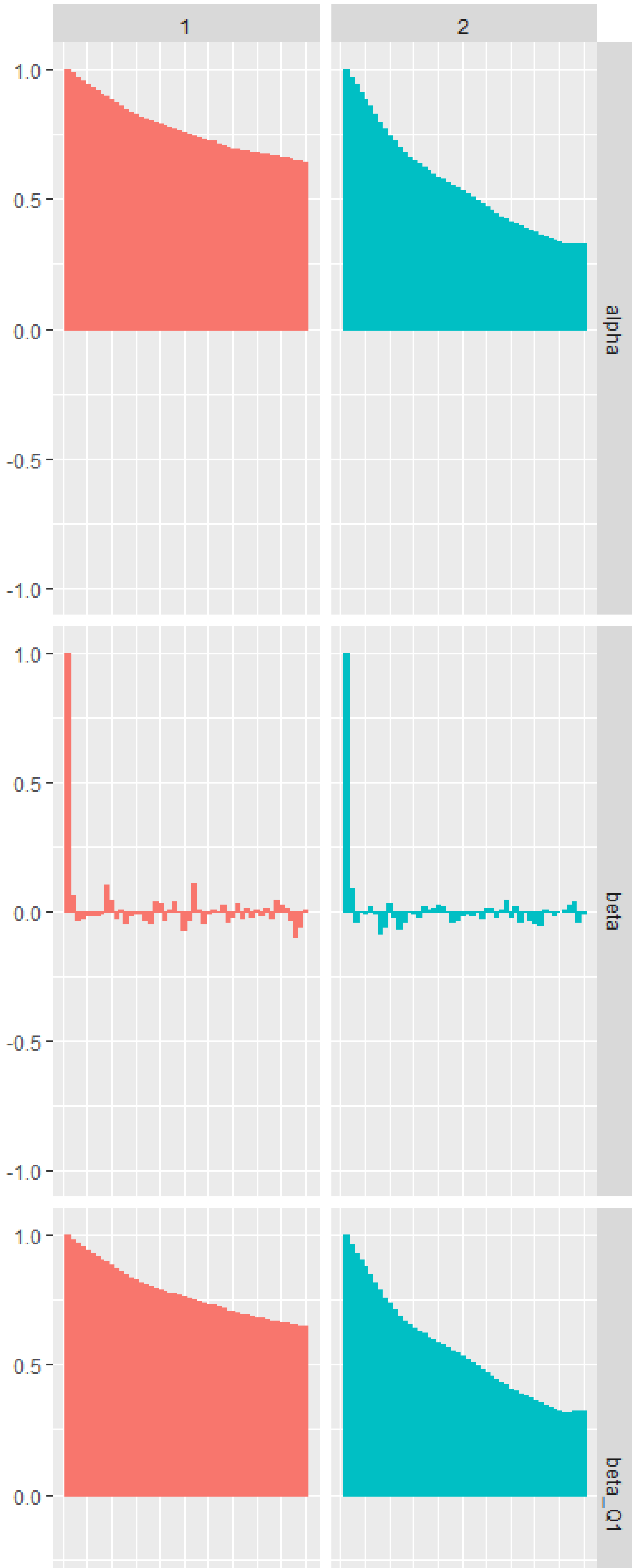
```
mcmclistII=as.mcmc(fit_second_model)
MCMCtrace(mcmclistII,second_params,ISB = FALSE,
          exact = TRUE,
          iter = 4000,
          ind = TRUE,
          pdf = FALSE)
```



Trace - alpha
Density - alpha
Trace - beta
Density - beta
Trace - beta_Q1
Density - beta_Q1
Trace - beta_Q2
Density - beta_Q2
Trace - beta_Q3
Density - beta_Q3
Trace - beta_Q4
Density - beta_Q4
Value
Density
Iteration
Parameter estimate




```
ggs_autocorrelation(ggs(mcmclistII))
```


1
2
alpha
beta
beta_Q1
1.0
0.5
0.0
-0.5
-1.0

```
summary(mcmclistII)
```

```
## 
## Iterations = 5001:19986
## Thinning interval = 15 
## Number of chains = 2 
## Sample size per chain = 1000 
## 
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
## 
##              Mean        SD  Naive SE Time-series SE
## alpha     65.71806 12.548180 0.2805858      2.6953516
## beta       2.78901  0.108658 0.0024297      0.0025724
## beta_Q1   -1.28651 12.582326 0.2813494      2.9047426
## beta_Q2    7.13661 12.624583 0.2822943      2.7673948
## beta_Q3    6.50569 12.514003 0.2798216      2.7720333
## beta_Q4   12.01257 12.517334 0.2798961      2.8014173
## deviance 391.23534  3.532400 0.0789869      0.0827779
## tau2       0.05643  0.009743 0.0002179      0.0002067
## 
## 2. Quantiles for each variable:
## 
##              2.5%       25%       50%       75%     97.5%
## alpha     48.14183  54.54574  64.07528  76.01270  90.95138
## beta       2.58114   2.71588   2.78747   2.86276   3.00170
## beta_Q1  -26.91338 -11.65848   0.38821   9.85983  16.42020
## beta_Q2  -18.21907  -3.33144   8.88341  18.31973  24.80278
## beta_Q3  -18.68987  -3.92684   8.10660  17.64221  23.91329
## beta_Q4  -13.36269   1.61221  13.77773  23.20393  29.28196
## deviance 386.32636 388.64684 390.52275 393.20346 399.98045
## tau2       0.03812   0.04969   0.05582   0.06256   0.07682
```

Comparing models

In this project, the Monte Carlo Markov (MCMC) chain approach was implemented on the sales data quarter.Moreover, , a linear regression model and a general linear model have been used. The MCMC value is checked by monitoring tracking graphs, auto-correlations, density functions and summary of results for its convergence diagnostics. The deviance information criterion (DIC) and the number of parameters are used as criteria for comparing models. The results show that the second model with 8 parameters is better than the first model with 10 parameters.

```
cat(' First_Model' ,'\t',  'DIC=  ',fit_first_model$BUGSoutput$DIC,'\t',
     'number of params=  ',length(fit_first_model$parameters.to.save),'\n',
     'Second_Model' ,'\t',  'DIC=  ',fit_second_model$BUGSoutput$DIC,'\t',
     'number of params=  ',length(fit_second_model$parameters.to.save))
```

```
##  First_Model      DIC=   402.9981     number of params=   10 
##  Second_Model     DIC=   397.4712     number of params=   8
```