Bayesian Structural Time Series

1. Introduction

1.1 Time Series Analysis

Time series records the development of random events based on time changing. Time series analysis comprises methods for analyzing time series data in order to extract meaningful statistics and other characteristics of the data [1]. Time series forecasting is the use of a model to predict future values based on previously observed values [1].

Time series analysis is often used in macro-control of national economy, enterprise management, market potential forecast, environmental pollution control, ecological balance, astronomy and oceanography. It is mainly analysed from four aspects, which are system description, system analysis, prediction of the future and decision-making control [1].

There are two analysis methods, which are frequency domain and time domain analysis [1]. Time domain analysis is mainly used here, which reveals the development rule of time series respect to time and include autocorrelation and cross correlation. Autocorrelation indicates the correlation between the time series and its delayed copy, the coefficient could be shown mathematically as,

$$\rho(t,t-k)=\frac{E(Y_t-\mu_t)(Y_{t-k}-\mu_{t-k})}{\sqrt{DY_t\times DY_{t-k}}}$$

Y_t is the observation in time series for the time point t. The earlier stages of Y_t are called lags for example, Y_{t-1} is the lag 1 of Y_t, Y_{t-k} is the lag k of Y_t. μ_t is the mean of the time series.

The other important parameter of time domain series is partial autocorrelation coefficient. The partial autocorrelation (k order) coefficient is the correlation between Y_{t-k} and Y_t under the condition of given $Y_{t-1}\ldots Y_{t-k+1}$, which could be shown mathematically as,

$$\rho_{Y_t,Y_{t-k}|Y_{t-1},\ldots,Y_{t-k+1}}=\frac{E[(Y_t-E[Y_t|Y_{t-1},\ldots,Y_{t-k+1}])(Y_{t-k}-E[Y_{t-k}|Y_{t-1},\ldots,Y_{t-k+1}])]}{E[(Y_{t-k}-E[Y_{t-k}|Y_{t-1},\ldots,Y_{t-k+1}])]^2}$$

1.2 Curve Fitting Model with Error Autoregression

In time series analysis, the stationarity of time series needs to be figured out firstly. A time series is stationary if the mean has no systematic change (no trend), the variance has no systematic change, and the periodic change is avoided. However, many time series are not stationary, with obvious upward or downward trend. In this case, the trend component can be fitted using a regression model, taking time as the independent variable and the corresponding observations as the dependent variable [2]. The trend fitting method include linear fitting and curve fitting [2]. The linear fitting model is,

$$Y_t=\alpha+\beta t+\varepsilon$$

The curve fitting method could be divided by several types, for example, quadratic, exponential or logistic. In this study, exponential fitting is mainly used, which could be transformed as,

$$ln(Y_t) = \alpha + \beta t + \varepsilon$$

1.3 Autoregression Moving Average (ARMA) Model

Autoregressive moving average is the combination of autoregression model and moving average model. Autoregression means that the dependent variable has a regression relationship with the dependent variable value of the previous time point, that is, its lag. The moving average considers that the difference between dependent variables, mainly due to current and past random term. Therefore, the ARMA model could be described as,

$$Y_t = \alpha + \beta_1 Y_{t-1} + \cdots + \beta_p Y_{t-p} + \varepsilon_t - \varphi_1 \varepsilon_{t-1} - \cdots - \varphi_q \varepsilon_{t-q}$$

Y_{t-p} is the p lag of Y_t, $\varepsilon_t \ldots \varepsilon_{t-q}$ is the white noise series.

1.4 Bayesian Method Analysis

In real life, the data size of many time series is quite small, which might contain just several months or years. At the same time, there are often data quality problems in the limited size sample. What is more, there is sometimes error autocorrelation in the data which could not be described directly using common models.

The Bayesian method is introduced into the computation of time series analysis in 80s. When using Bayesian method, a reasonable prior distribution is set for the parameters which needs to be estimated (a prior distribution with little effect on the calculation process is selected when there is no prior information), and then the posterior distribution is obtained by adding the weight of the sample data. Markov Chain Monte Carlo is a main Bayesian computation method. It introduces the Markov process of stochastic process into Monte Carlo simulation to realize dynamic simulation and generates posterior distribution samples based on Bayesian principle [3].

Using Bayesian method to analyse time series, the estimation will be not only a prediction value, but a complete probability distribution. At the same time, taking priori distribution into account can better handle uncertain factors. Winbugs is an object-oriented interactive version of BUGS, which is a software that uses Bayesian analysis and MCMC method to solve complex statistical models.

2. Case Study

2.1 Data Description

China's agriculture has experienced several major changes. Around 1953, as China was in a planned economy system, the government adopted a policy of unified management, pricing and acquisition towards agriculture. In 1978, China's economic system changed, and the development of agriculture gradually appeared in various forms. By 1988, the agricultural development under the market economy system had stabilized. Therefore, it is meaningful to study the development trend of agriculture between 1952 and 1988.

The dataset of this study is the agricultural gross national income (GNI) index. The data cover a total of 37 years from 1952 to 1988. The main objective of the study is to use Bayesian method to build a more comprehensive and accurate model to describe how agricultural GNI change with time, and the model could be used to predict the agricultural GNI in a near future.

In order to get better understanding of the data, a plot of how GNI changes over time is drawn, as shown in Figure 1. As could be seen in Figure 1, the value of GNI from 1952 to 1988 presents a clear upward trend, but there is a significant decline in 1959 and 1960.

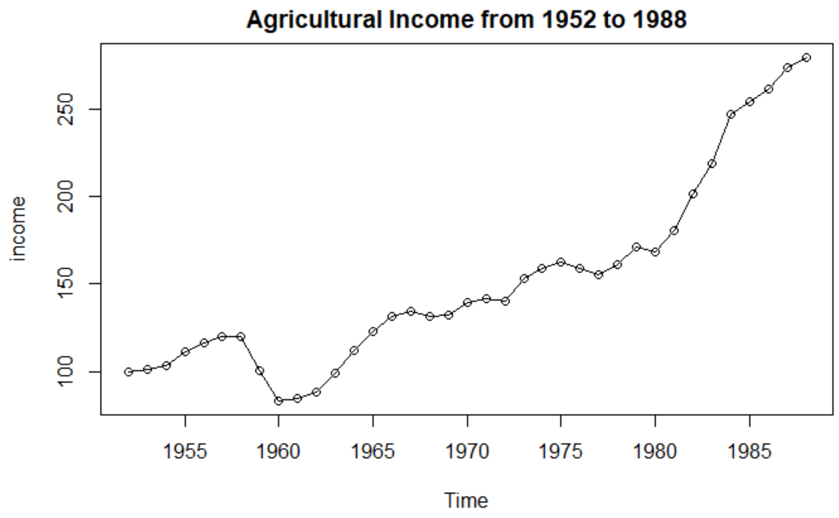


Figure 1 Agricultural Income from 1952 to 1988

Figure 2 is the autocorrelation coefficient plot of GNI. As can be seen from Figure 2, the autocorrelation coefficient decreases very slowly and shows sinusoidal fluctuation, which is the typical feature of the time series with trending.

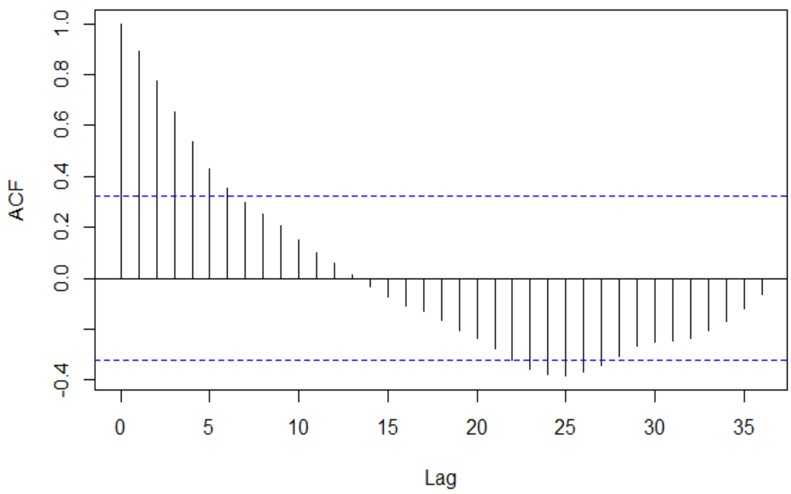


Figure 2 Agricultural income Autocorrelation Plot

2.2 Curve Fitting Model with Error Autoregression

As can be seen from Figure 1, the data shows a curve trend over time, so the exponential type model is used to fit the trend term, and the error autoregression is taken into account as well.

$$\begin{cases} \log(Y_t) \sim N(\alpha + \beta t + \varepsilon_t, \tau_1) \\ \varepsilon_t \sim N(\rho\varepsilon_{t-1}, \tau_2) \\ \tau_1 = {}^{1}/_{\sigma_1^2} \\ \tau_2 = {}^{1}/_{\sigma_2^2} \end{cases} \qquad \text{(Model 1)}$$

Y_t is the GNI value for time t, τ is the precision of the model and σ is the standard deviation. When setting prior distribution to the parameters of the model, the Normal-Gamma distribution is used. That is, if the corresponding variables has a normal distribution, the conjugate prior distribution of its precision τ is gamma distribution. Because the correlation coefficient ranges between -1 and 1, it is assumed that it is a beta distribution.

$$\begin{cases} \rho \sim Beta(1,3) \\ \alpha \sim N(0, 10^{-5}) \\ \beta \sim N(0, 10^{-5}) \\ \tau_1 \sim Gamma(0.01,0.01) \\ \tau_2 \sim Gamma(0.01,0.01) \end{cases}$$

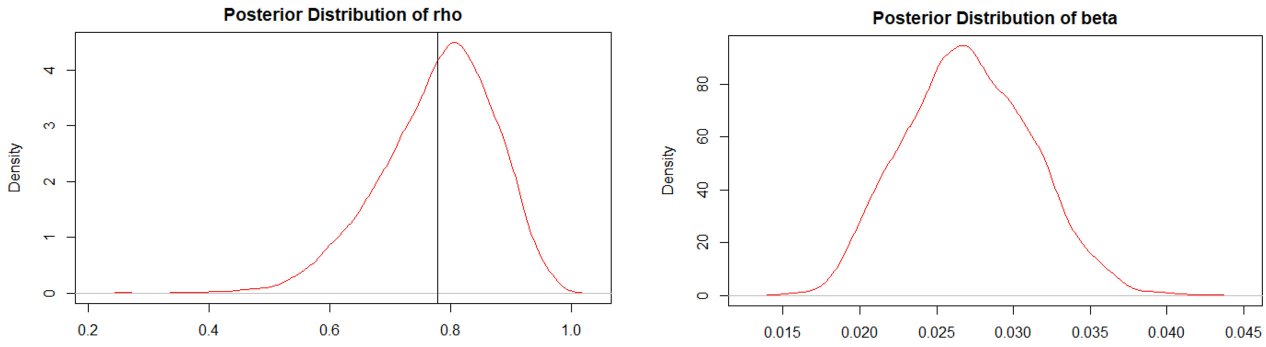
In the process of model operation, 5000 Gibbs pre-iterations are carried out to ensure the convergence of parameters. Then we discard (burn-in) the pre-iteration and do 10000 iterations. The results are shown in Table1. It could be figured out that in average, when t increases by 1 year, the value of GNI would grow 0.027 unit. At the same time, based on the value of ρ, the error has autocorrelation and the coefficient is 0.778.

Table 1 Running Result of Model 1

	mean	sd	2.5%	25%	50%	75%	97.5%
alpha	4.489	0.090	4.316	4.427	4.493	4.553	4.657
beta	0.027	0.004	0.020	0.024	0.027	0.030	0.035
tau2	171.718	52.501	91.165	133.975	165.400	201.800	292.702
rho	0.778	0.095	0.569	0.720	0.790	0.846	0.933
deviance	-123.781	13.791	-149.800	-133.200	-123.900	-114.600	-96.040

Figure 3 shows the posterior distribution ρ and β. As can be seen from Figure (a), the correlation coefficient reaches the maximum density around 0.8 and has the mean larger than 0.7 (black line) after adding the weight of the data. Because the Beta distribution (which has the mean at 1/(1+3)=0.25) used in setting prior distribution does not take the negative value for correlation coefficient into account, if the trend of the Beta posterior distribution is similar to or larger than the prior distribution, the prior distribution is reasonable.

At the same time. It can be seen from Figure (b) that the posterior distribution of β still presents a well normal distribution, so the prior distribution for β is reasonable.



(a) **(b)**

Figure 3 Posterior Distribution of Parameters in Model 1

The GNI of t ranging from 1 to 36, that is year 1953 to year 1988, is estimated by model 1. The 95% credible interval of the posterior probability distribution is obtained and shown in the following figure. As can be seen from the figure, the model can fit the increasing trend of GNI, but the turning points and vibration in the rising trend are not well estimated.

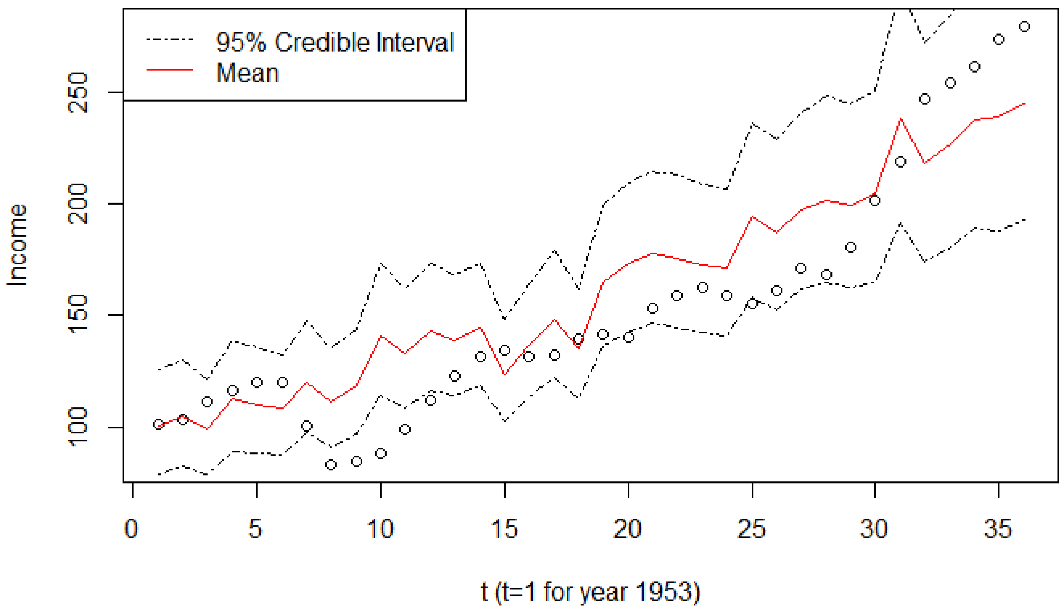


Figure 4 Posterior Estimation of Sample based on Model 1

The GNI of year 1989 (t=37) is predicted using the model. The probability density distribution of the GNI of 1989 is shown as Figure 5. The 95% credible interval of the distribution is (255.1115, 321.8345), with the mean of 268.8094.

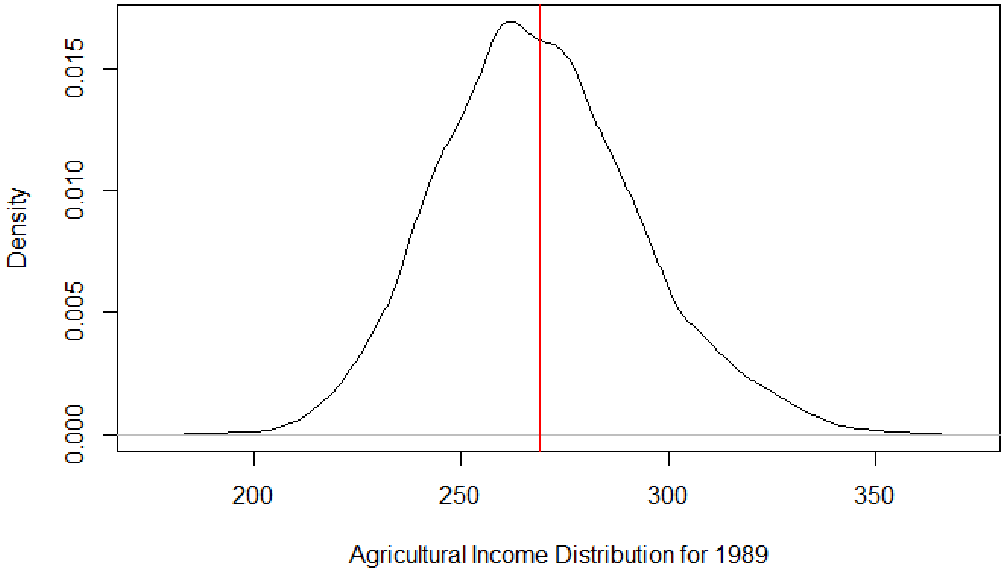


Figure 5 Prediction for t=37

2.3 Autoregression Moving Average Model

Since the data used has increasing trend, which is not stationary, so the partial correlation coefficient is plotted (Figure 6) after the data are differentiated. Figure 7 shows an obvious relationship between the response variable and its lag 1. Therefore, lag 1 should be involved in ARMA model.

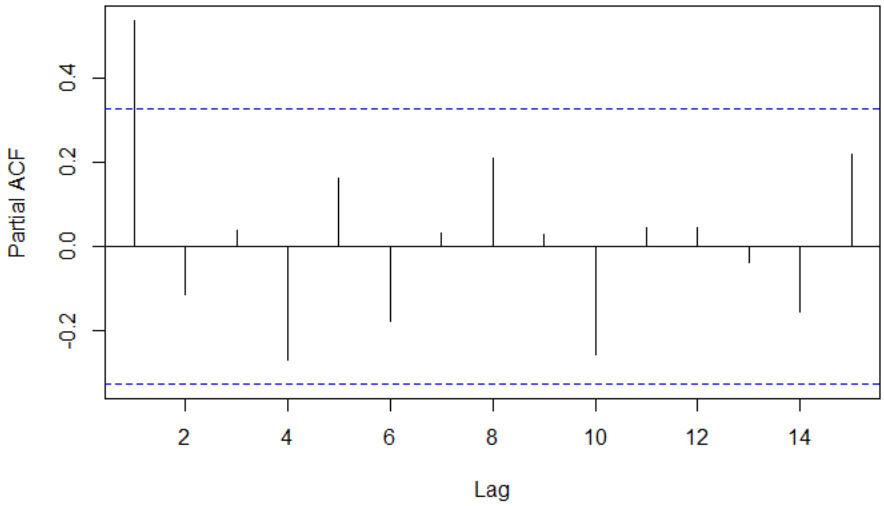


Figure 6 Partial Correlation Coefficient Plot

The ARMA model is built, the model involves the lag 1 of Y_t and error series.

$$\begin{cases} Y_t \sim N(\alpha + \beta Y_{t-1} + \varepsilon_t + \varphi \quad \varepsilon_{t-1}, \tau_1) \\ \varepsilon_t \sim N(0, \tau_2) \\ \tau_1 = {}^{1}/_{\sigma_1^2} \\ \tau_2 = {}^{1}/_{\sigma_2^2} \end{cases} \qquad \text{(Model 2)}$$

Y_t is the GNI value for time t, τ is the precision of the model and σ is the standard deviation. When the prior distribution is set, the gamma distribution is still used as the prior distribution of the precision of variable having normal distribution, and the beta distribution is used as the prior distribution of φ . Therefore, the priori distribution is then set as follows,

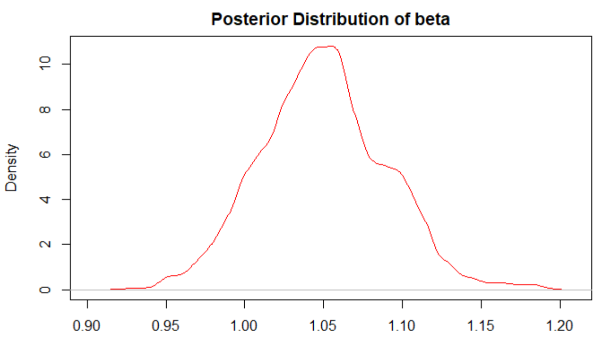
$$\begin{cases} \alpha \sim N(0{,}0.0001) \\ \tau_1 \sim Gamma(0.01{,}0.01) \\ \tau_2 \sim Gamma(0.01{,}0.01) \\ \beta \sim N(0{,}0.0001) \\ \varphi \sim Beta(1{,}3) \end{cases}$$

10000 Gibbs pre-iterations are carried out to ensure the convergence of parameters. Then we discard (burn-in) the pre-iteration and do 30000 iterations. The results are as shown in Table 2. The result shows that in average, when Y_{t-1} increased 1 unit, Y_t will increase 1.050 unit.

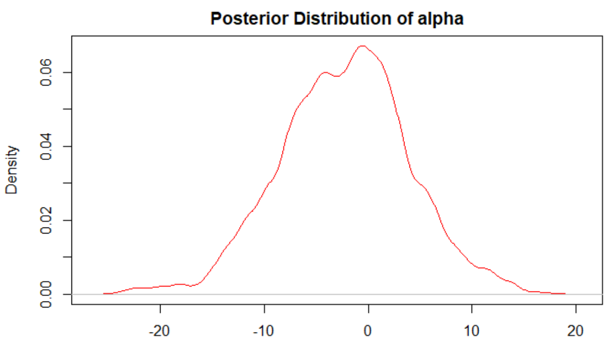
Table 2 Running Result of Model 2

	mean	sd	2.5%	25%	50%	75%	97.5%
beta	1.050	0.040	0.973	1.023	1.049	1.075	1.129
alpha	-2.259	6.222	-14.430	-6.365	-2.027	1.821	10.070
tau1	14.242	34.805	0.030	0.400	2.256	11.322	112.300
tau2	0.030	0.278	0.010	0.015	0.018	0.022	0.039
phi	0.513	0.164	0.164	0.408	0.519	0.629	0.803
deviance	75.186	79.714	-68.631	14.680	72.485	135.125	227.103

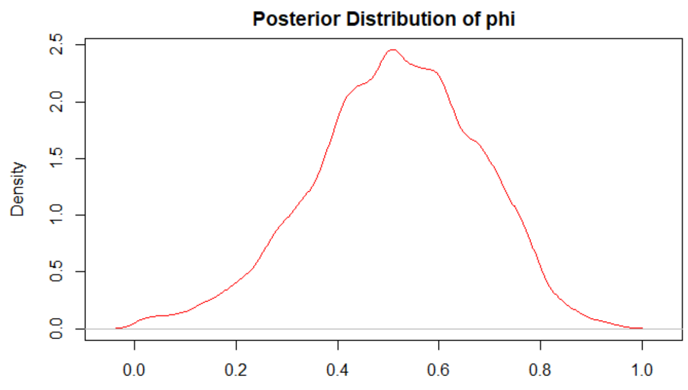
Figure 7 is the posterior distributions of the parameters of model 2. As can be seen from the (a) and (b), α and β are approximately normal distributions, and the highest probability densities of φ which are shown in (c) is around 0.5. Therefore, the prior distributions of this model are reasonable.



(a)



(b)



(c)

Figure 7 Posterior Distribution of Parameters of Model 2

The GNI of t ranging from 1 to 36, that is year 1952 to year 1988, is estimated by model 2. The 95% credible interval of the posterior probability distribution is shown in the following Figure 8. As can be seen from the figure, except the GNI value of t being 7 and 8, the rest true values all fall into 95% credible interval, the model estimates the trend and turning point very well.

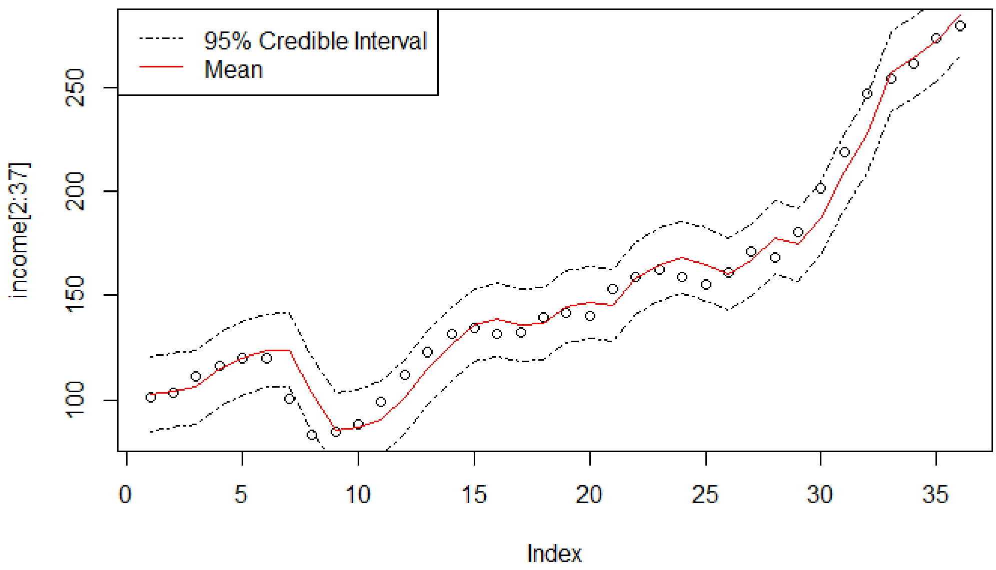


Figure 8 Posterior Estimation of Sample based on Model 2

The GNI of year 1989 (t=37) is predicted by the model. The probability density distribution of the GNI of 1989 is shown as Figure 9. The 95% credible interval of the distribution is [275.1725, 306.2537], with the mean of 291.0663.

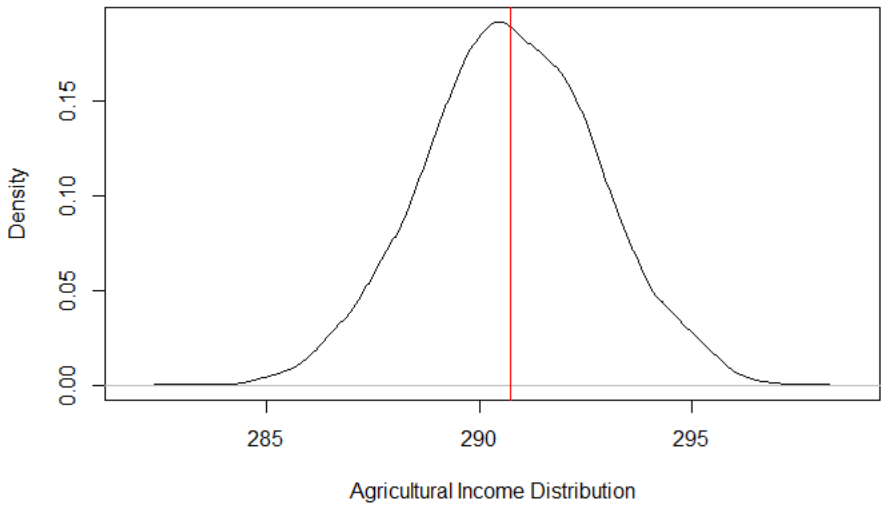


Figure 9 Prediction for t=37

Figure 10 shows the autocorrelation coefficient distribution (boxplot and density distribution) of the residual obtained from model 2. The 95% credible interval of the distribution is (-1.407, 0.344) and contains 0. Therefore, it can be considered that the residual of the model is not autocorrelated which could meet the assumption of ARMA model.

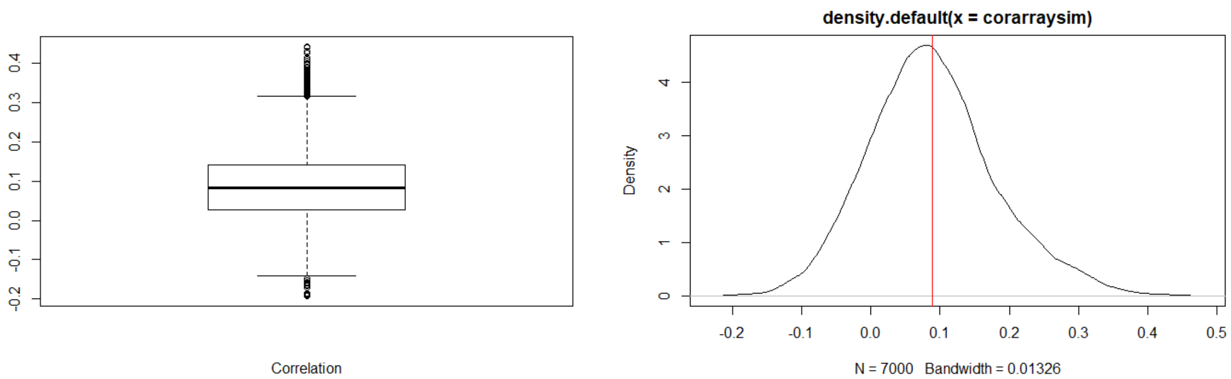


Figure 10 Autocorrelation Coefficient Distribution of Residual in Model 2

2.4 Model Comparison

Although model 2 can better fit the shape of sample data, model 1 is more meaningful in real life. Since model 1 contains trend item which contains t, the trend of GNI over time can be expressed by it. However, model 2 only estimates the current value on the basis of its lag 1 and error, which relies more on the sample data and involves more random effects, which does not represent the long-term trend of GNI over time.

3. Conclusion

In this research, time series models using Bayesian method are built based on a case study about the agricultural GNI Index. The two models are built: curve fitting model and autoregression moving average model. Curve fitting Model based on Bayesian method describes how the GNI value changes with time and could offer a complete posterior probability distribution, which could be considered when forecasting. When the research is focus on the time series in the time period of the sample, the ARMA model may work better.

Reference

[1] https://en.wikipedia.org/wiki/Time_series

[2]https://baike.baidu.com/item/%E6%97%B6%E9%97%B4%E5%BA%8F%E5%88%97%E5%88%86%E6%9E%90/8724605?fr=aladdin

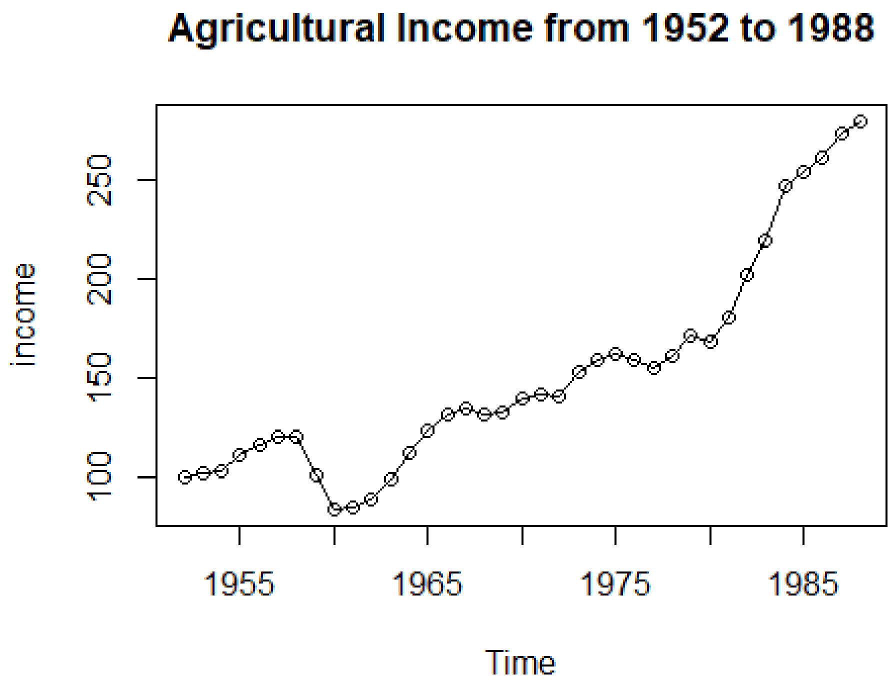
[2] Yan Wang, Time Series Analysis, 2015.03

[3] Brooks S. Markov Chain Monte Carlo method and its Application. The Statistician, 1998, 47(1): 69-100

Appendix: R Code

Model 1

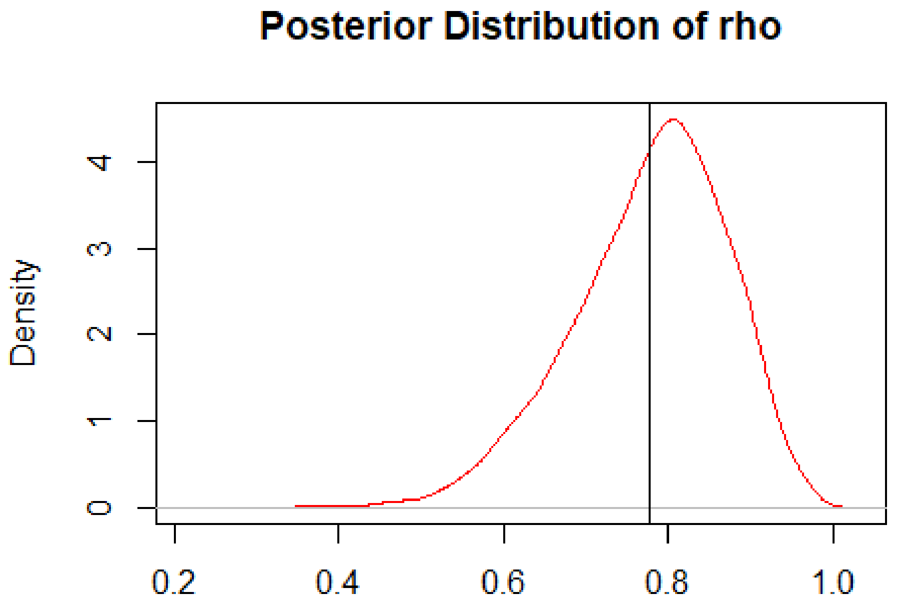
```
#import the data
agincome<-read.csv('D:/llq/2018/Bayes/project/Agriculturedata.csv',header =
TRUE)
head(agincome)
##   Year Agriculture.National..Income
## 1 1952                        100.0
## 2 1953                        101.6
## 3 1954                        103.3
## 4 1955                        111.5
## 5 1956                        116.5
## 6 1957                        120.1
summary(agincome)
##       Year      Agriculture.National..Income
##  Min.   :1952   Min.   : 83.6
##  1st Qu.:1961   1st Qu.:111.9
##  Median :1970   Median :139.8
##  Mean   :1970   Mean   :151.9
##  3rd Qu.:1979   3rd Qu.:168.4
##  Max.   :1988   Max.   :279.4
income<-ts(agincome$Agriculture.National..Income,start = 1952)
plot(income,type='o',main="Agricultural Income from 1952 to 1988")
```



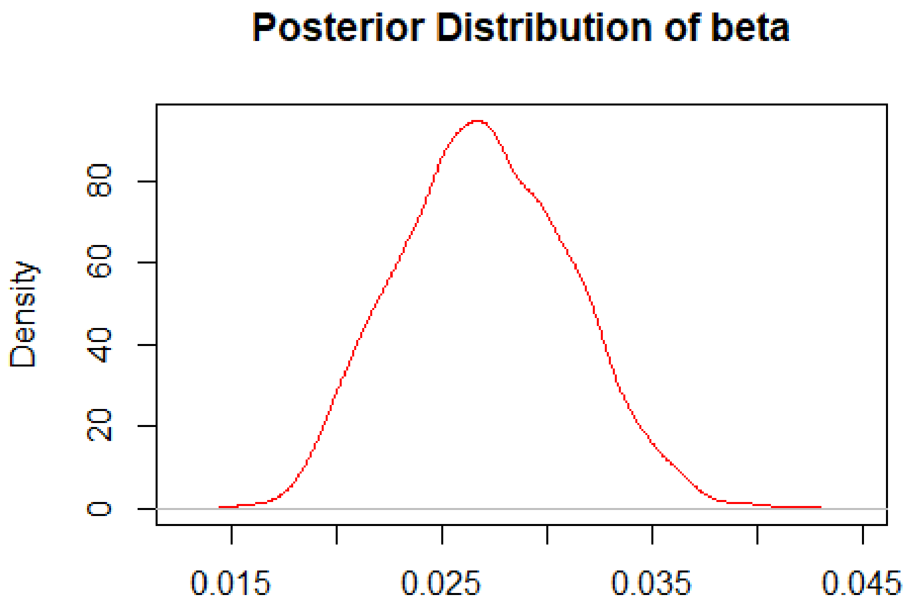

```
incomeres<-income[1:37]
###one lag
library(R2WinBUGS)
## Warning: package 'R2WinBUGS' was built under R version 3.4.4
## Loading required package: coda
## Warning: package 'coda' was built under R version 3.4.4
## Loading required package: boot
## Warning: package 'boot' was built under R version 3.4.4
```

```
tslm.model1 <- function(){

  for (t in 2:T) {

    #mu[t]<-alpha+beta*t+error[t]

    y[t] ~ dnorm(mu[t],tau1)
    mu[t]<-alpha+beta*t+error[t]
    error[t] ~ dnorm(mu2[t],tau2)
    mu2[t] <- rho*error[t-1]

  }
  error[1] ~ dnorm(0,tau2)
  rho ~ dbeta(1,3)
  alpha ~ dnorm(0,1.0E-5)
  beta ~ dnorm(0,1.0E-5)
  tau1 ~ dgamma(0.01,0.01)
  tau2 ~ dgamma(0.01,0.01)
}

# the file I will save it too
tslm.file1 <- "D://tsmodel1.txt"

## write model file:
write.model(tslm.model1,tslm.file1)
## and let's take a look:
file.show(tslm.file1)

system.time(
ts.bugs1 <- bugs(data=list(T=37,y=log(incomeres)),
      inits=list(list(tau1=1,tau2=1,alpha=4,beta=0.04)),
      parameters.to.save=c("alpha","beta","tau2","rho","error","tau1"),
      model.file=tslm.file1,
      n.chains=1,
      n.iter=10000,
      n.sim=5000,
      n.burnin=5000,
      n.thin=1,
      DIC=T,
      bugs.directory=paste0(Sys.getenv(c("USERPROFILE")),
"\\WinBUGS14"),debug=T)
)
##    user  system elapsed
##    1.09    0.07   14.55
ts.bugs1$DIC
## [1] -97.765
round(ts.bugs1$summary,3)
##                mean      sd    2.5%     25%     50%     75%   97.5%
## alpha         4.489   0.090   4.316   4.427   4.493   4.553   4.657
## beta          0.027   0.004   0.020   0.024   0.027   0.030   0.035
## tau2        171.718  52.501  91.165 133.975 165.400 201.800 292.702
## rho           0.778   0.095   0.569   0.720   0.790   0.846   0.933
## error[1]      0.031   0.075  -0.120  -0.019   0.030   0.080   0.184
```

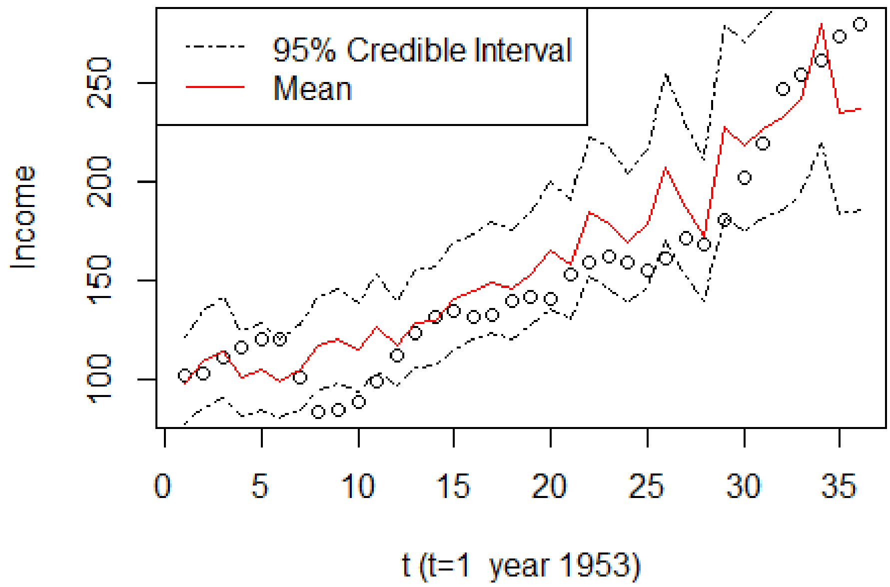
```
## error[2]      0.067   0.083   -0.093    0.011    0.065    0.122   0.231
## error[3]      0.074   0.086   -0.089    0.015    0.072    0.131   0.246
## error[4]      0.107   0.085   -0.054    0.049    0.106    0.165   0.276
## error[5]      0.123   0.083   -0.035    0.066    0.121    0.179   0.285
## error[6]      0.119   0.081   -0.031    0.063    0.118    0.173   0.275
## error[7]      0.072   0.079   -0.075    0.017    0.071    0.128   0.228
## error[8]     -0.093   0.076   -0.239   -0.146   -0.093   -0.041   0.056
## error[9]     -0.259   0.076   -0.405   -0.312   -0.260   -0.208  -0.108
## error[10]    -0.295   0.074   -0.436   -0.346   -0.296   -0.245  -0.152
## error[11]    -0.279   0.072   -0.416   -0.327   -0.278   -0.229  -0.137
## error[12]    -0.210   0.070   -0.346   -0.258   -0.210   -0.161  -0.078
## error[13]    -0.128   0.067   -0.260   -0.173   -0.127   -0.082   0.005
## error[14]    -0.064   0.066   -0.198   -0.108   -0.063   -0.020   0.059
## error[15]    -0.026   0.065   -0.161   -0.069   -0.024    0.019   0.096
## error[16]    -0.031   0.064   -0.159   -0.074   -0.028    0.013   0.092
## error[17]    -0.062   0.064   -0.193   -0.104   -0.060   -0.019   0.059
## error[18]    -0.080   0.063   -0.209   -0.122   -0.077   -0.035   0.037
## error[19]    -0.068   0.064   -0.199   -0.109   -0.066   -0.023   0.052
## error[20]    -0.075   0.064   -0.205   -0.118   -0.073   -0.031   0.044
## error[21]    -0.094   0.066   -0.229   -0.137   -0.092   -0.049   0.031
## error[22]    -0.059   0.066   -0.203   -0.101   -0.056   -0.015   0.065
## error[23]    -0.047   0.067   -0.190   -0.089   -0.045   -0.002   0.078
## error[24]    -0.056   0.068   -0.196   -0.101   -0.054   -0.010   0.068
## error[25]    -0.095   0.070   -0.240   -0.140   -0.093   -0.047   0.035
## error[26]    -0.132   0.073   -0.287   -0.179   -0.128   -0.083   0.003
## error[27]    -0.129   0.074   -0.282   -0.178   -0.125   -0.077   0.006
## error[28]    -0.111   0.075   -0.267   -0.159   -0.109   -0.059   0.032
## error[29]    -0.127   0.076   -0.288   -0.176   -0.125   -0.076   0.018
## error[30]    -0.092   0.079   -0.256   -0.142   -0.090   -0.037   0.058
## error[31]    -0.023   0.083   -0.199   -0.078   -0.020    0.036   0.127
## error[32]     0.035   0.084   -0.140   -0.021    0.039    0.095   0.189
## error[33]     0.108   0.088   -0.074    0.050    0.110    0.170   0.268
## error[34]     0.123   0.090   -0.063    0.062    0.125    0.187   0.291
## error[35]     0.128   0.093   -0.065    0.066    0.130    0.193   0.303
## error[36]     0.139   0.095   -0.052    0.074    0.142    0.208   0.319
## error[37]     0.132   0.096   -0.066    0.070    0.133    0.200   0.311
## tau1        446.595 170.198  190.990  323.100  419.100  544.800 842.617
## deviance   -123.781  13.791 -149.800 -133.200 -123.900 -114.600 -96.040
plot(density(ts.bugs1$sims.list$rho),col='red',main='Posterior Distribution of
rho',xlab='')
abline(v=mean(ts.bugs1$sims.list$rho),cex=6)
```



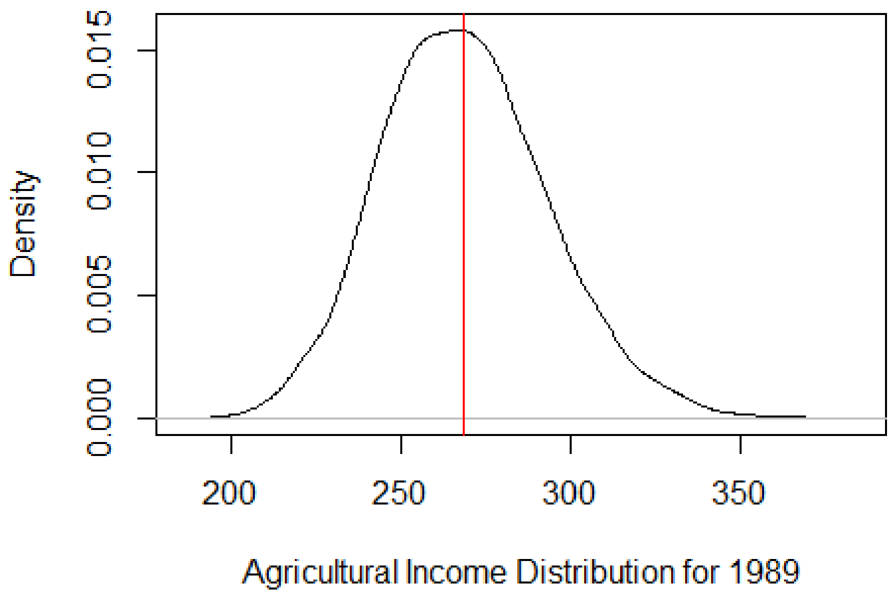

```
plot(density(ts.bugs1$sims.list$beta),col='red',main='Posterior Distribution of
beta',xlab='')
```




```
simarray1<-array(dim = c(36,5000))
for(j in 1:36){
  simarray1[j,]<-
exp(ts.bugs1$sims.list$alpha+ts.bugs1$sims.list$beta*(j+1)+rnorm(5000,ts.bugs1$s
ims.list$rho*ts.bugs1$sims.list$error[j],1/sqrt(ts.bugs1$sims.list$tau2)))
}
post25_1<-apply(simarray1,1,quantile,.025)
post975_1<-apply(simarray1,1,quantile,.975)
plot(income[2:37],ylab = "Income",xlab = "t (t=1  year 1953)")
legend('topleft',lty = c(4,1),legend=c('95% Credible
Interval','Mean'),col=c('black','red'))
points(apply(simarray1,1,mean),type = 'l',col='red')
points(post25_1,type = 'l',lty=4)
points(post975_1,type = 'l',lty=4)
```




```
##forecast
forecastdis<-
exp(rnorm(5000,(4.489+0.027*37+(rnorm(5000,0.132*0.778,(1/sqrt(171.718)))),(1/s
qrt(446.6))))
plot(density(forecastdis),xlab = 'Agricultural Income Distribution for
1989',main='')
abline(v=mean(forecastdis),col='red',cex=6)
```

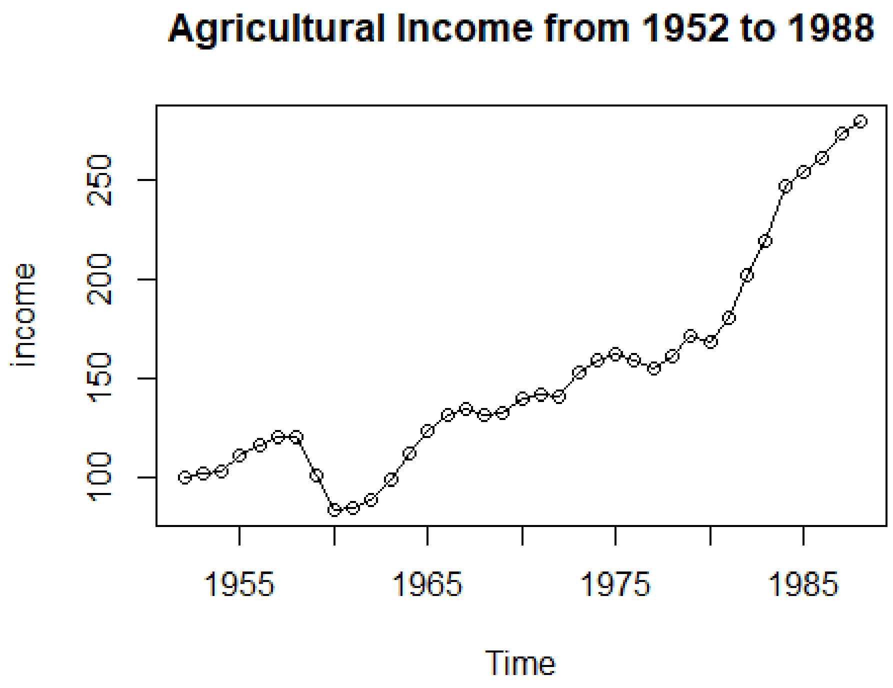



```
quantile(forecastdis,c(0.025,0.975))
##     2.5%    97.5%
## 223.879 320.623
mean(forecastdis)
```

```
## [1] 268.7188
```

Model 2

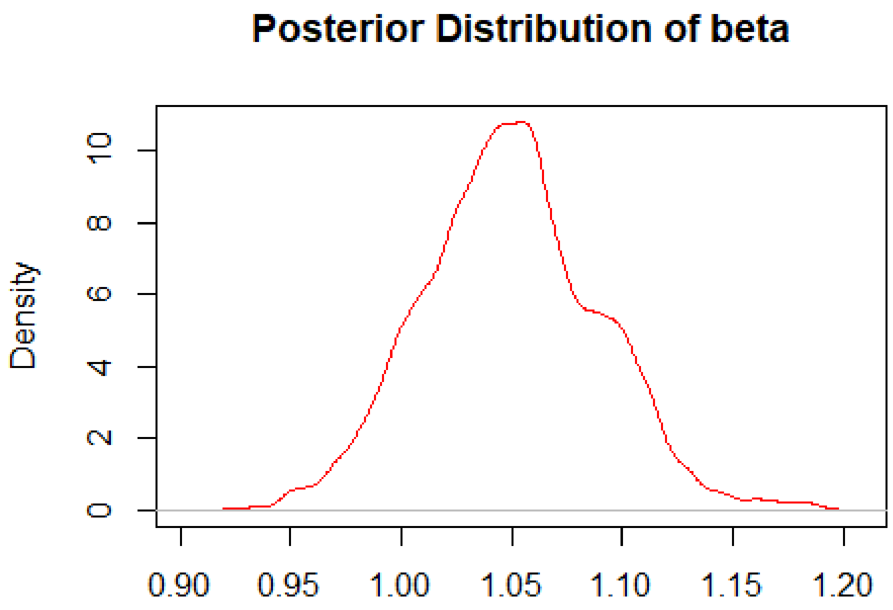
```
#import the data
agincome<-read.csv('D:/llq/2018/Bayes/project/Agriculturedata.csv',header = TRUE)
head(agincome)
##   Year Agriculture.National..Income
## 1 1952                        100.0
## 2 1953                        101.6
## 3 1954                        103.3
## 4 1955                        111.5
## 5 1956                        116.5
## 6 1957                        120.1
summary(agincome)
##       Year      Agriculture.National..Income
##  Min.   :1952   Min.   : 83.6
##  1st Qu.:1961   1st Qu.:111.9
##  Median :1970   Median :139.8
##  Mean   :1970   Mean   :151.9
##  3rd Qu.:1979   3rd Qu.:168.4
##  Max.   :1988   Max.   :279.4
income<-ts(agincome$Agriculture.National..Income,start = 1952)
plot(income,type='o',main="Agricultural Income from 1952 to 1988")
```



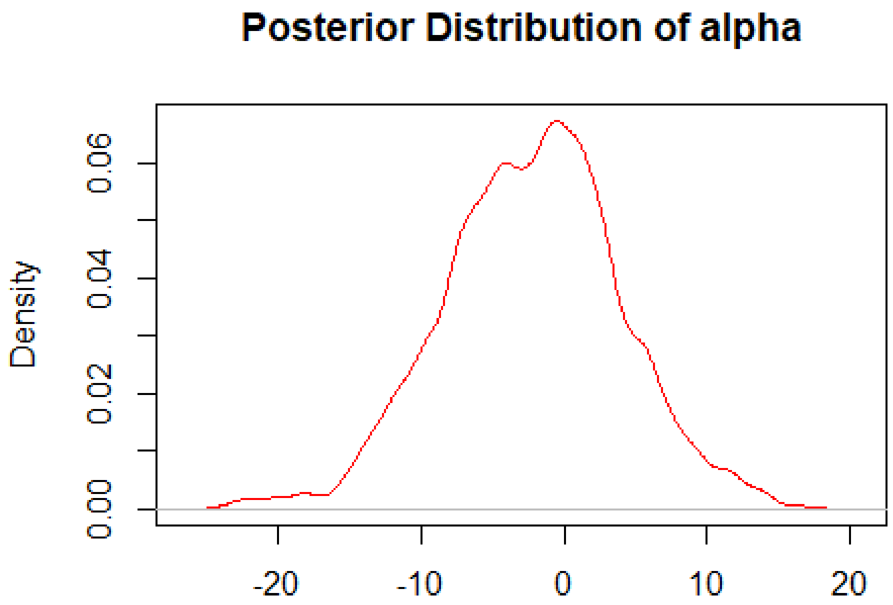

```
incomeres<-income[1:37]
###one lag
library(R2WinBUGS)
## Warning: package 'R2WinBUGS' was built under R version 3.4.4
## Loading required package: coda
## Warning: package 'coda' was built under R version 3.4.4
## Loading required package: boot
## Warning: package 'boot' was built under R version 3.4.4
```

```
tslm.model4 <- function(){
  for (t in 1:T) {

    u[t]~dnorm(0,tau2)
  }
  for (t in 2:T) {
    Y[t]~dnorm(mu[t],tau1)
    mu[t]<-u[t]+beta*Y[t-1]+phi*u[t-1]+alpha
  }

  alpha~dnorm(0,0.0001)
  tau1~dgamma(0.01,0.01)
  tau2~dgamma(0.01,0.01)
  beta~dnorm(0,0.0001)
  phi~dbeta(1,3)
}

# the file I will save it too
tslm.file4 <- "D://tsmodel4.txt"

## write model file:
write.model(tslm.model4,tslm.file4)
## and let's take a look:
file.show(tslm.file4)

system.time(
ts.bugs4 <- bugs(data=list(T=37,Y=incomeres),
      inits=list(list(tau1=1,tau2=1),list(tau1=0.5,tau2=0.5)),
      parameters.to.save=c("beta","alpha","tau1","tau2","phi","u"),
      ### PATH TO THE MODEL FILE
      model.file=tslm.file4,
      n.chains=2,
      n.iter=30000,
      n.sim=10000,
      n.burnin=10000,
      n.thin=1,
      DIC=T,
      bugs.directory=paste0(Sys.getenv(c("USERPROFILE")), "\\WinBUGS14"),debug=T)
)
##    user  system elapsed
##    7.61    0.02   35.19
ts.bugs4$DIC
## [1] 47.169
round(ts.bugs4$summary,3)
##              mean     sd    2.5%     25%     50%     75%   97.5%  Rhat
## beta        1.050  0.040   0.973   1.023   1.049   1.075   1.129 1.012
## alpha      -2.259  6.222 -14.430  -6.365  -2.027   1.821  10.070 1.007
## tau1       14.242 34.805   0.030   0.400   2.256  11.322 112.300 1.005
## tau2        0.030  0.278   0.010   0.015   0.018   0.022   0.039 1.045
## phi         0.513  0.164   0.164   0.408   0.519   0.629   0.803 1.001
## deviance   75.186 79.714 -68.631  14.680  72.485 135.125 227.103 1.005
##           n.eff
## beta        200
## alpha      1100
## tau1        360
## tau2        930
## phi       40000
## u[1]        280
```

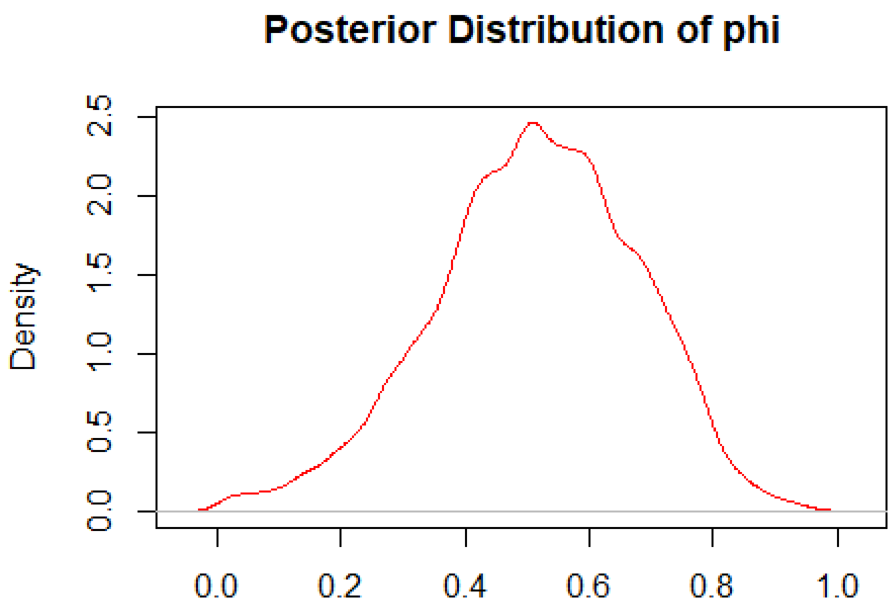
```
## u[2]      2200
## u[3]       220
## u[4]     30000
## u[5]       230
## u[6]       600
## u[7]       330
## u[8]       230
## u[9]       710
## u[10]    40000
## u[11]     1000
## u[12]     7800
## u[13]     7600
## u[14]      320
## u[15]     1000
## u[16]      170
## u[17]      170
## u[18]      220
## u[19]      400
## u[20]      170
## u[21]      140
## u[22]     1400
## u[23]      170
## u[24]      150
## u[25]      170
## u[26]      120
## u[27]      340
## u[28]      160
## u[29]      130
## u[30]      370
## u[31]     1200
## u[32]       76
## u[33]     1700
## u[34]       99
## u[35]      200
## u[36]      150
## u[37]       91
## deviance   360
plot(density(ts.bugs4$sims.list$beta),col='red',main='Posterior Distribution of
beta',xlab='')
```



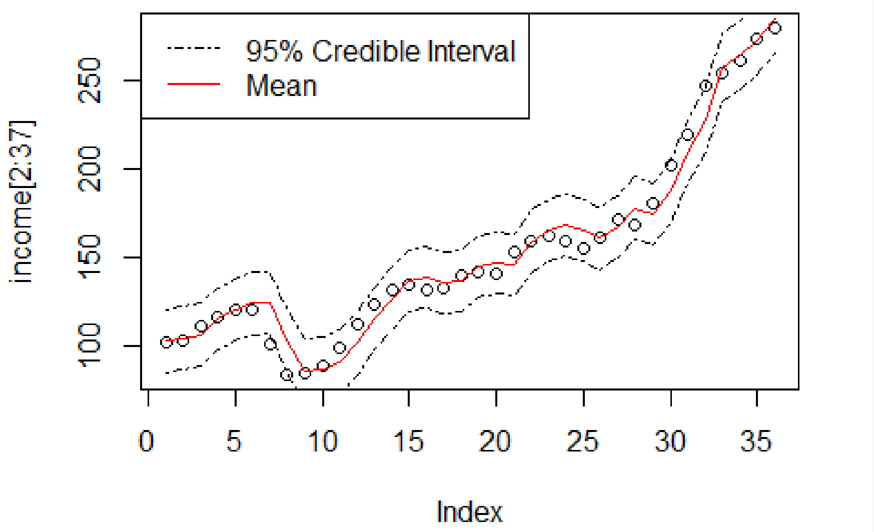

```r
plot(density(ts.bugs4$sims.list$alpha),col='red',main='Posterior Distribution of alpha',xlab='')
```




```r
plot(density(ts.bugs4$sims.list$phi),col='red',main='Posterior Distribution of phi',xlab='')
```

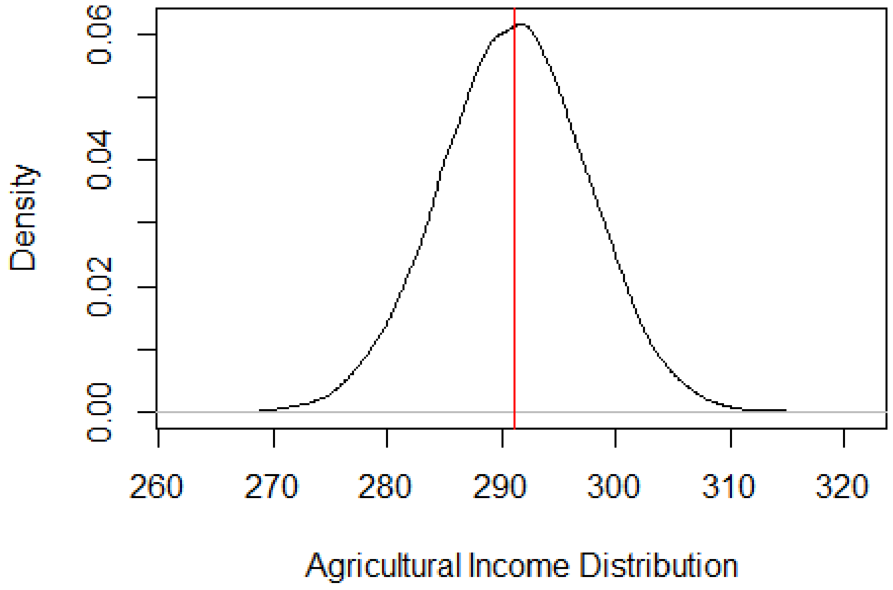



```
simarray1<-array(dim = c(36,40000))
for(j in 2:37){
  simarray1[j-1,]<-
rnorm(40000,0,1/sqrt(ts.bugs4$sims.list$tau2))+ts.bugs4$sims.list$beta*incomeres[j-
1]+ts.bugs4$sims.list$phi*rnorm(40000,0,1/sqrt(ts.bugs4$sims.list$tau2))+ts.bugs4$sims.l
ist$alpha
}
post25_1<-apply(simarray1,1,quantile,.025)
post975_1<-apply(simarray1,1,quantile,.975)
plot(income[2:37])
legend('topleft',lty = c(4,1),legend=c('95% Credible
Interval','Mean'),col=c('black','red'))
points(apply(simarray1,1,mean),type = 'l',col='red')
points(post25_1,type = 'l',lty=4)
points(post975_1,type = 'l',lty=4)
```




```
##forecast
forecastdis3<-rnorm(20000,rnorm(20000,0,(1/sqrt(0.03)))+1.05*(279.4)-
rnorm(20000,0,(1/sqrt(0.03)))*0.513-2.259,(1/sqrt(14.424)))
```

```r
plot(density(forecastdis3),xlab = 'Agricultural Income Distribution',main='')
abline(v=mean(forecastdis3),col='red',cex=6)
```




```r
quantile(forecastdis3,c(0.025,0.975))
##     2.5%    97.5%
## 278.3157 303.6690
mean(forecastdis3)
## [1] 291.1096
```