Homework #3

Vlado Vukovic

Your Last+First Name ___VUKOVIC___VLADO___ Your Matricola 1772953

Abstract

Bayesian analysis of historical log change of monthly unemployment rate with Markov chain Monte Carlo (MCMC) simulation method. The following models utilized AR(1) and ARMA(1,1).

1. Intro

2. Models

2.1 Model AR(1)

2.2 Model ARMA(1,1)

3. Data

4 Implementation

4.1 AR(1)

4.1.1 MCMC Plots

4.1.2 Frequentist Comparison

4.2 AR(1,1)

4.2.1 MCMC Plots

4.2.2 Frequentist Comparison

4.3 DIC

5. Predictions

6. Conclusion

```
## This homework will be graded and it will be part of your final evaluation
##
##
```

```
## Last update by LT: Thu Jan 11 12:09:50 2018
```

Abstract

Bayesian analysis of historical log change of monthly unemployment rate with Markov chain Monte Carlo (MCMC) simulation method. The following models utilized AR(1) and ARMA(1,1).

1. Intro

Bayesian inference revolves around making use of Bayes theorem (Bayes, 1763) (Laplace, 1812) (Jeffrys, 1939) to update prior belief

$$P(\theta|x) = \frac{P(x|\theta)P(\theta)}{P(x)}$$

and although denominator looks naive it is impossible to integrate for non-trivial problems hence through the years Monte Carlo Markov Chain methods were developed.

Markov Chain is a stochastic process that satisfies Markov property (Markov, 1954) aka memorylessness. In oversimplification, Markov properties are satisfied if the future process is predictable solely based on current state.

Monte Carlo method (Metropolis&Ulam, 1949) is class of algorithms that obtains results via random sampling and solves three broad problem classes: optimization, numerical integration and generating draws from posterior distribution.

Markov chain Monte Carlo (Metropolis et al.,1953; Hastings, 1970) is a general method for simulation of stochastic processes having probability densities known up to a constant of proportionality. (Geyer, 1992)

In particular, ability to sample from posterior distribution is crucial for Bayesian analysis although direct sampling is not usually possible following algorithms have been developed: Metropolis-Hastings(MH), Gibbs, Hamiltonian, No-U-Turn Sampler.

Current state of the art tools for implementing probabilistic programming include: winBUGS, jags, pymc3, Stan and Edward. Differences come from underlying platform, syntax and implemented samplers.

Models presented below are implemented in R programming language via rjags library hence models are developed for jags which uses both Gibbs and MH sampler depending on the problem.

2. Models

ARMA(p,q) is a auto-regressive (AR) moving average (MA) statistical model for time series data that describes the expected level of the current term as a function of the actual sizes of the previous terms and is specified by two order parameters (p,q).

Autoregressive (AR(p)) component is a random process thats used to describe time-series. Its output variable depends linearly on its own previous values and on a stochastic term. The parameter p specifies the number of lags used in the regression equation of a model. AR(1) or, equivalently, ARMA(1,0), is represented as

$$y_t = c + \phi y_{t-1} + \epsilon_t$$

Moving average (MA(q)) component represents the error of the model as a combination of previous error terms ϵ_t. The order q determines the number of terms to include in the model. MA(1) or, equivalently, ARMA(0,1), is represented as

$$y_t = c + \theta\epsilon_{t-1} + \epsilon_t$$

Combining the two componenets together in first order **ARMA(1,1)**, is represented as

$$y_t = c + \phi y_{t-1} + \theta \epsilon_{t-1} + \epsilon_t$$

Therefore, this type of models is a form of stochastic difference equation. Considering the fact that literature treats returns of exchange and stock prices as random walk: ARMA(1,1), ARMA(2,2) and autoregressive family of algorithams in general may be appropriate for the problem at hand. The disadvantage of these models is that it directly rely on past values, and therefore work best on long and stable series. Fitting an ARIMA model requires the series to be stationary. A series is said to be stationary when its mean, variance, and autocovariance are time invariant.

Parameter ϕ represents represents order of the lag used for autoregressive component. Parameter θ represents order of the lag used for moving averages component.

Following the seminal works by (Fama 1965) (Mandelbrot 1963) where is was initially shows that logarhitmic returns of cotton and stock prices have fatter tails than normal distribution, we will use student t distribution.

2.1 Model AR(1)

$$y_i \sim \mathcal{N}(\mu_i + \phi_i * Y_{i-1}, \sigma^2)$$

Whose parameters share same distribution hence

$$\mu_i \sim \mathcal{N}(0, 0.00001)$$

$$\phi_i \sim \mathcal{N}(0, 0.00001)$$

2.2 Model ARMA(1,1)

while proposed model may be represented as following

$$y_i \sim \mathcal{N}(\mu_i + \phi_i * Y_{i-1} + \theta_i * e_{i-1}, \sigma^2)$$

Whose parameters share same distribution hence

$$\mu_i \sim \mathcal{N}(0, 0.001)$$

$$\phi_i \sim \mathcal{N}(0, 0.001)$$

$$\theta_i \sim \mathcal{N}(0.001, 0.0001)$$

3. Data

Data is collected from FRED and represents monthly unemployment rates in US since 1949.

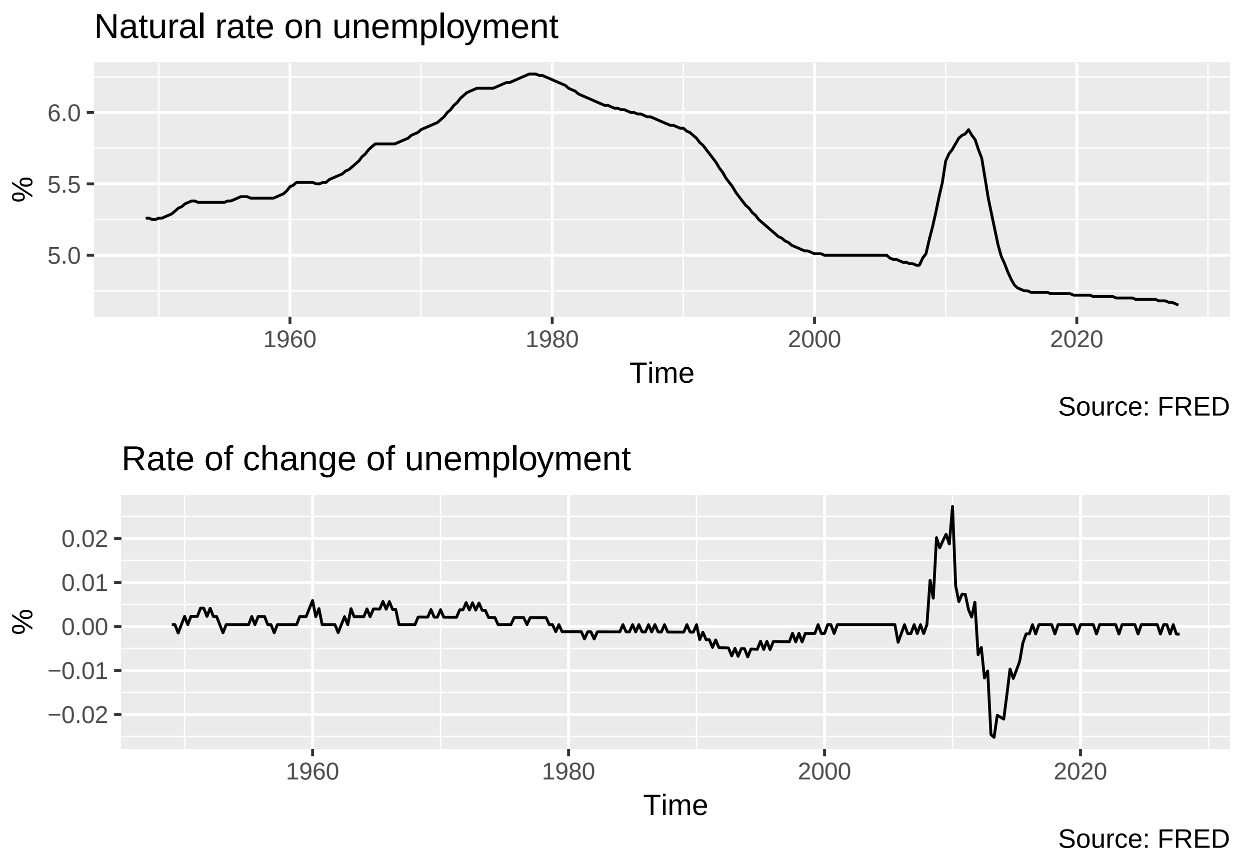
In particular, log change is used because similarly returns which are defined as

$$r_i = \frac{p_i - p_j}{p_j}$$

introduce normalization to data while log has several benefits both algorithmic and theoretical some of which are:

a) log-normality $1 + r_i = \frac{p_i}{p_j} = \exp^{\log(\frac{p_i}{p_j})}$
b) approximate raw-log equality $\log(1+r) \approx r, r << 1$
c) time-additivity $(1+r_1)(1+r_2)...(1+r_n) = \prod_i (1+r_i)$ which using following logarithmic identities $\log(1+r_i) = \log(\frac{p_i}{p_j}) = \log(p_i) - log(p_j)$ simplifies calculating compounded returns $\sum_i \log(1+r_i) = \log(1+r_1) + \log(1+r_2) + ... + \log(1+r_n) = \log(p_n) - log(p_0)$
d) mathematical ease $\exp^x = \int \exp^x dx = \frac{d}{dx}\exp^x = \exp^x$
e) numerical stability

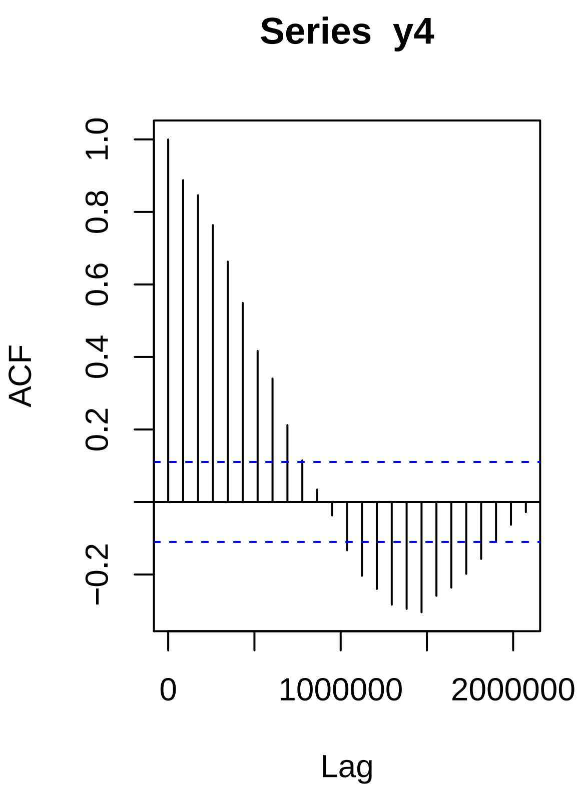
```
y <- Quandl("FRED/NROUST")
y2 <- xts(x = y$Value, order.by = as.POSIXct(y$Date))
y3 <- diff(log(y2))
y3[1] <- 0
colnames(y3) = "Change"
y4 <- y3 - mean(y3) #subtract mean from each element

p01 <- ggplot(y2, aes(x=index(y2))) + geom_line(aes(y=y2[,1])) +
  labs(title="Natural rate on unemployment", caption="Source: FRED", y="%",x = "Time")
p02 <- ggplot(y4, aes(x=index(y4))) + geom_line(aes(y=Change)) +
  labs(title="Rate of change of unemployment", caption="Source: FRED", y="%",x = "Time")
ggarrange(p01, p02, ncol = 1, nrow = 2)
```

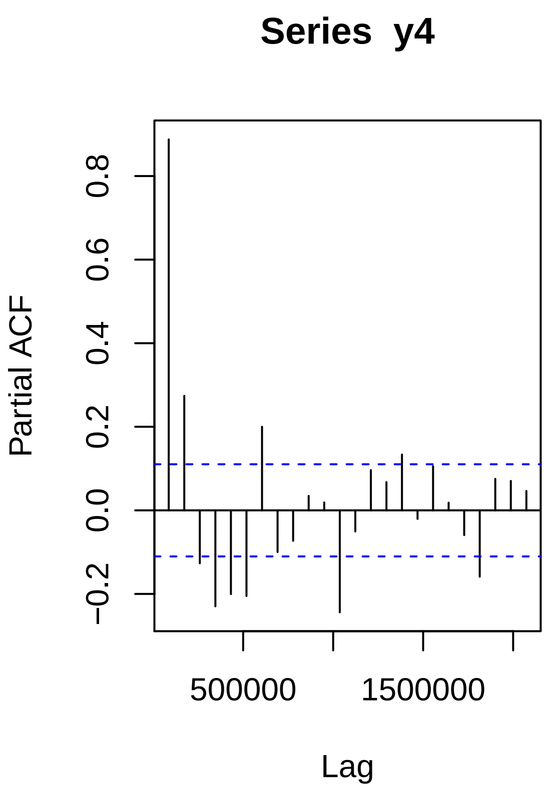


Stationality is confirmed via Augumented Dickey-Fuller (ADF) test which gives p-value=0.01 indicating strong confirmation. While ACF and PACF plots confirm positive AR and MA order of 1.

```
library(tseries)
stat = adf.test(y4)
stat
```

```r
par(mfrow=c(1,2))
acf(y4)
pacf(y4)
```





```r
pred <- y4[300:316]
y4[300:316] <- NA

yL <- y4[[1]]
for(i in 2:length(y4)) {
  yL <- c(yL,y4[[i]])
}
# create a list useful for rjags model, that includes number of occurances
unempData <- list(N = length(yL), Ynote = yL)
```

4 Implementation

Probabilistic library rjags is one of the libraries in R programming language for performing Bayesian analysis using Markov chain Monte Carlo (MCMC) methods. The approach for all models is to iterate 15,000 times for each variable whereby initial 5,000 values are discarded as burn-in period values. In order to perform DIC model checking 2 chains will be run. Each model will be described by parameter summary values, traceplots, running means, density plots, and autocorrelation plots. Models and approach in line with explanations in Models are given vague/non-informative priors in order to exercise least effect on posterior distribution. Models draw from Student T distribution due to the evidence of returns having fatter tails than in normal distribution (Mandelbrot 1963, Fama 1965).

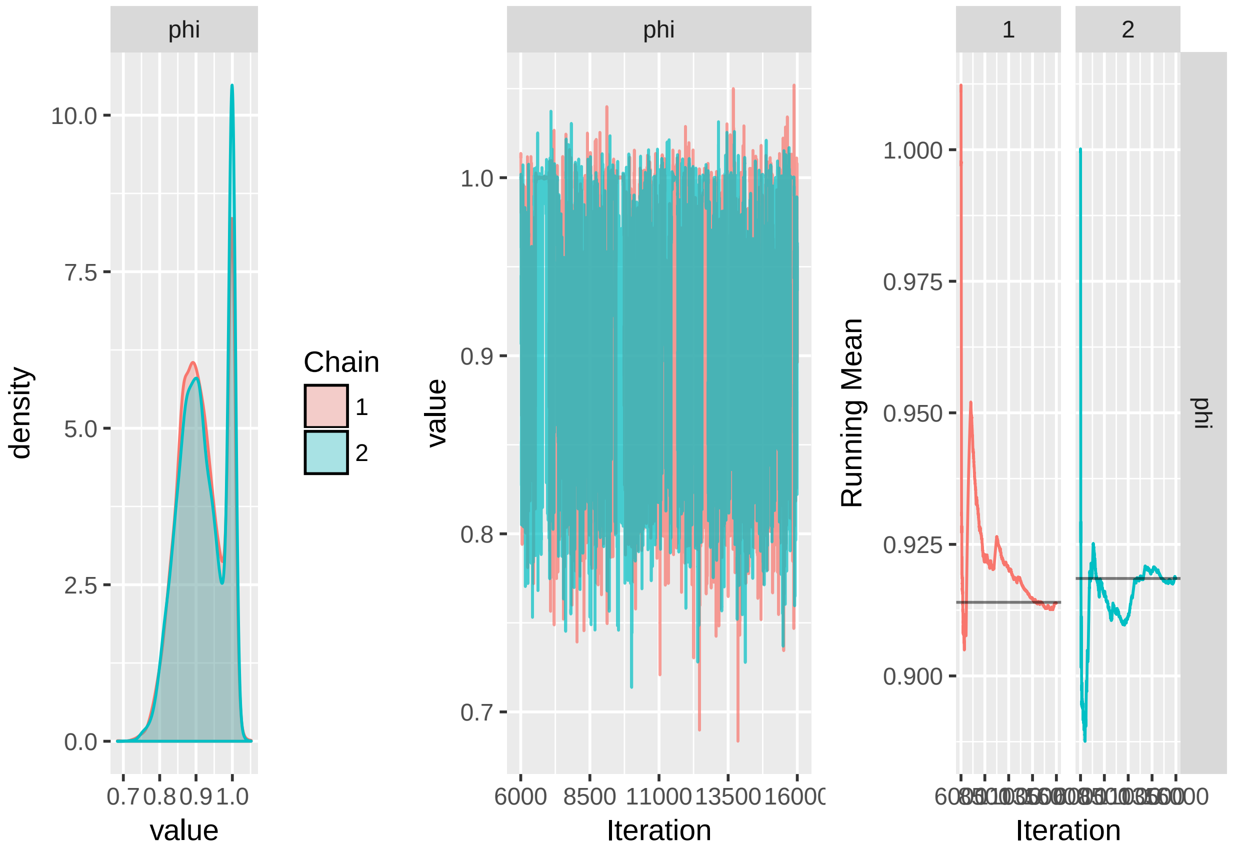
4.1 AR(1)

```r
ar <-
"model{
predmu[1] <- 1
for(i in 2:N){
  predmu[i] <- mu + phi * (Ynote[i-1] - mu)
  Ynote[i] ~ dt(predmu[i], tau,1)
}
mu ~ dnorm(0, 0.00001)
phi ~ dnorm(0, 0.00001)
tau ~ dgamma(0.1 ,0.001)
sigma <- 1/sqrt(tau)
}"

m2 <- jags.model(textConnection(ar), data = unempData, n.chains = 2)

update(m2, 5000)
m2.samples <- coda.samples(m2, c('Ynote','mu','phi','predmu'), 10000)

m2.sum <- summary(m2.samples)
m2.allSamples <- ggs(m2.samples)
m2.relSamples <- m2.allSamples %>% filter(Parameter %in% c("phi","predmu"))
m2.predY <- m2.allSamples %>% filter(Parameter %in% unique(grep("Ynote*",m2.allSamples$Parameter,value=
m2.predYMean <- m2.predY %>% group_by(Parameter) %>% summarise(mean(value))
m2.predYMeanL <- list(m2.predYMean$`mean(value)`)
m2.Values <- data.frame("Time"=seq(1:lengths(m2.predYMeanL)),m2.predYMeanL)
colnames(m2.Values) <- c("Time","Yar")
```

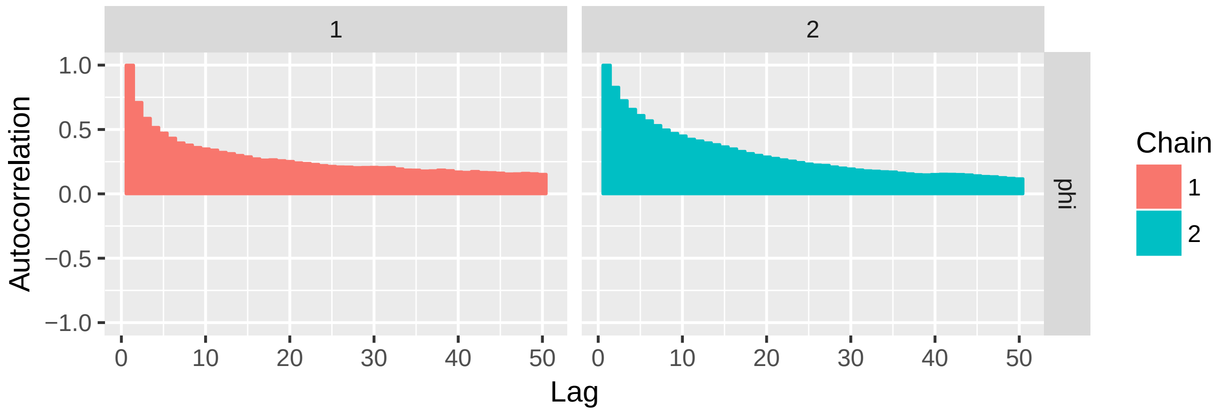
4.2.1 MCMC Plots

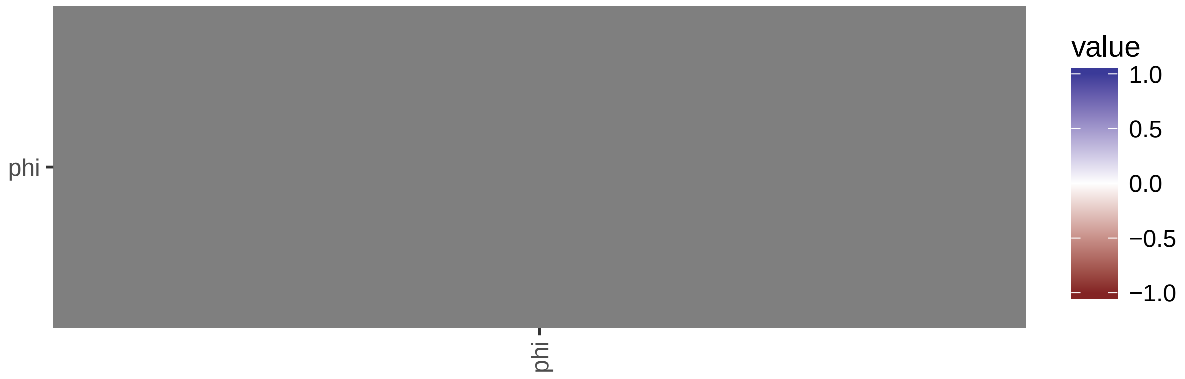
The following plots indicate stable chains and timely convergence of parameters.

```r
p12 <- ggs_density(m2.relSamples) + guides(colour=FALSE)
p22 <- ggs_traceplot(m2.relSamples) + guides(colour=FALSE)
p32 <- ggs_running(m2.relSamples) + guides(colour=FALSE)
ggarrange(p12, p22, p32, ncol = 3, nrow = 1)
```



```r
p42 <- ggs_autocorrelation(m2.relSamples)
p52 <- ggs_crosscorrelation(m2.relSamples)
ggarrange(p42, p52, ncol = 1, nrow = 2)
```



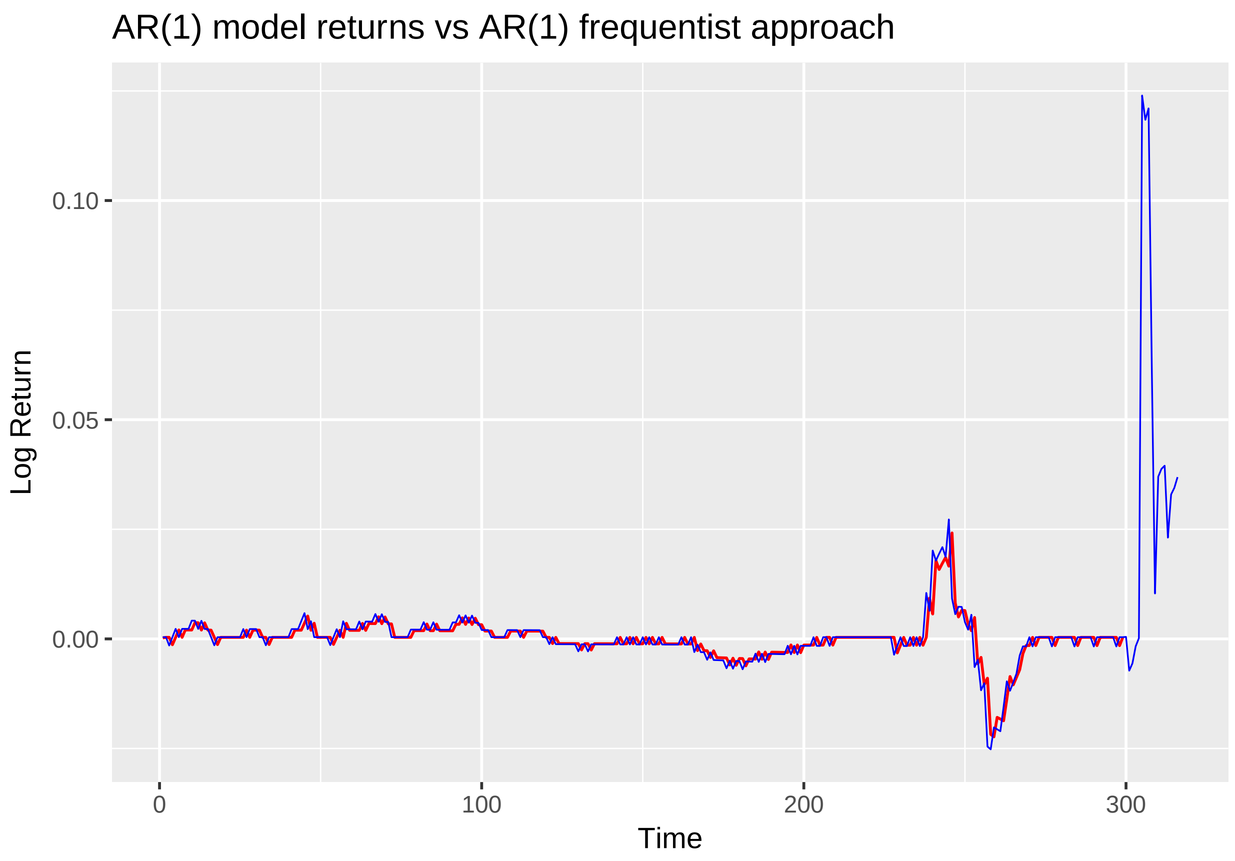


4.2.2 Frequentist Comparison

In order to test obtained results frequentist comparison is used via Arima package, and plot of fitted values indicates a strong model.

```
m2.temp1 <- Arima(ts(unempData$Y),order=c(1,0,0))
m2.temp2 <- data.frame(cbind("Time" = time(ts(unempData$Ynote)),"V"= fitted(m2.temp1), "O"=m2.temp1$x,"

f2 <- ggplot() + geom_line(data=m2.temp2, aes(x=Time,y=V), color="red", size=0.5)  + geom_line(data=m2.
f2
```

```
## Warning: Removed 17 rows containing missing values (geom_path).
```



4.2 ARMA(1,1)

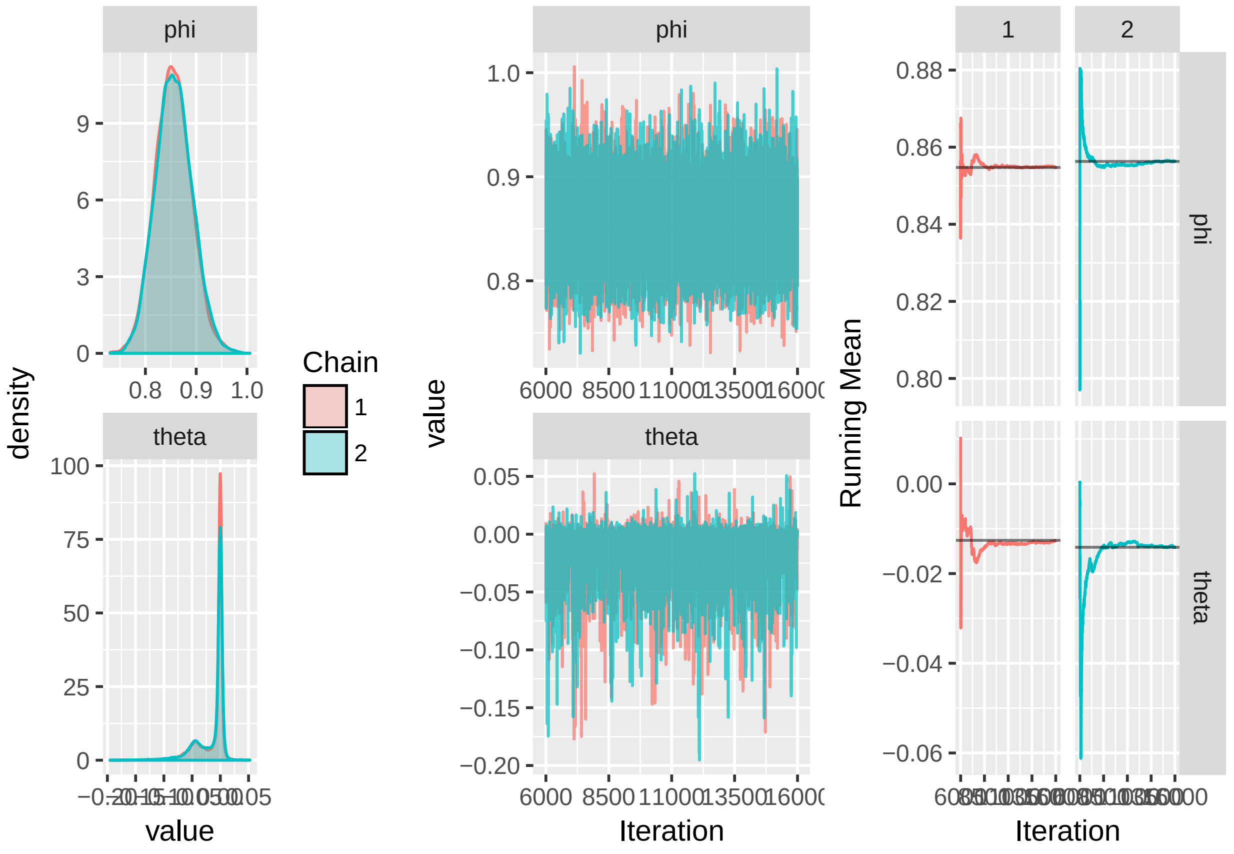
```
arma1 <- "
model
{
  predw[1] <- 1
  diffmu[1] <- 1
  for (i in 2:N){
    predw[i] <- Ynote[i] - phi * Ynote[i-1] - theta * predw[i-1]
    diffmu[i] <- phi * Ynote[i-1] + theta * predw[i-1]
    Ynote[i] ~ dt(diffmu[i], tau,1)
  }
  phi ~ dnorm(0, 0.001)
  theta ~ dnorm(0, 0.001)
  tau ~ dgamma(0.001 ,0.0001)
  sigma <- 1/sqrt(tau)
}"

m1 <- jags.model(textConnection(arma1), data = unempData, n.chains = 2)

update(m1, 5000)
m1.samples <- coda.samples(m1,c('phi','theta','predw','diffmu','Ynote'),10000)

m1.sum <- summary(m1.samples)
m1.allSamples <- ggs(m1.samples)
m1.relSamples <- m1.allSamples %>% filter(Parameter %in% c("phi","theta"))
```

```
m1.predY <- m1.allSamples %>% filter(Parameter %in% unique(grep("Ynote*",m1.allSamples$Parameter,value=
m1.predYMean <- m1.predY %>% group_by(Parameter) %>% summarise(mean(value))
m1.predYMeanL <- list(m1.predYMean$`mean(value)`)
m1.Values <- data.frame("Time"=seq(1:lengths(m1.predYMeanL)),m1.predYMeanL)
colnames(m1.Values) <- c("Time","Yarma11")
```

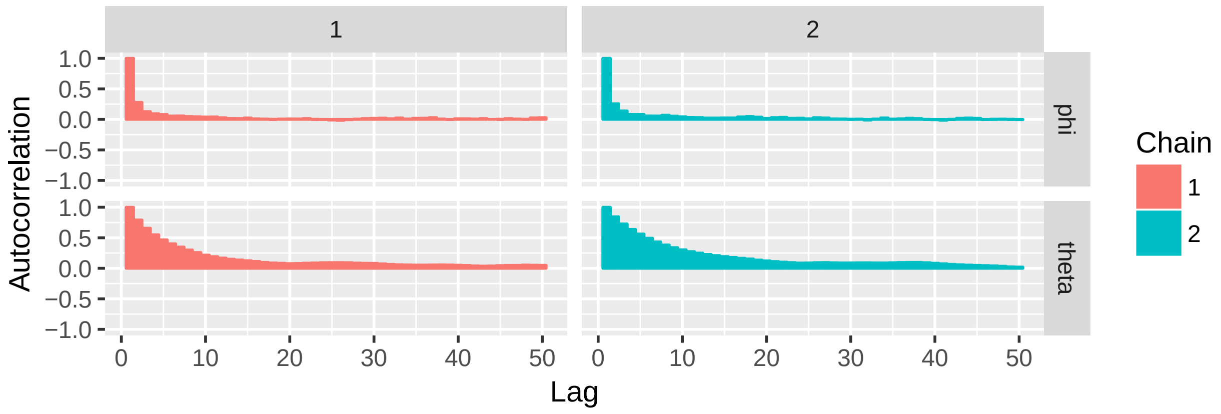
4.2.1 MCMC Plots

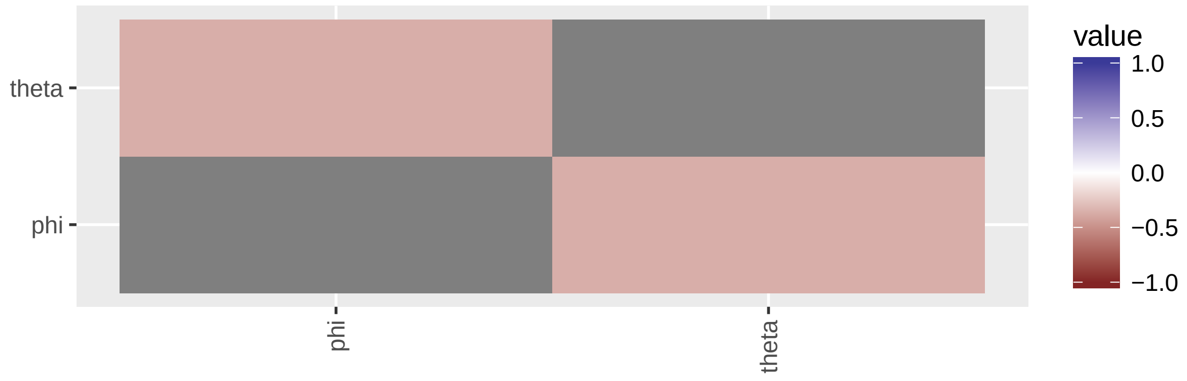
The following plots indicate stable chains and timely convergence of parameters.

```
p11 <- ggs_density(m1.relSamples) + guides(colour=FALSE)
p21 <- ggs_traceplot(m1.relSamples) + guides(colour=FALSE)
p31 <- ggs_running(m1.relSamples) + guides(colour=FALSE)
ggarrange(p11, p21, p31, ncol = 3, nrow = 1)
```



```
p41 <- ggs_autocorrelation(m1.relSamples)
p51 <- ggs_crosscorrelation(m1.relSamples)
ggarrange(p41, p51, ncol = 1, nrow = 2)
```



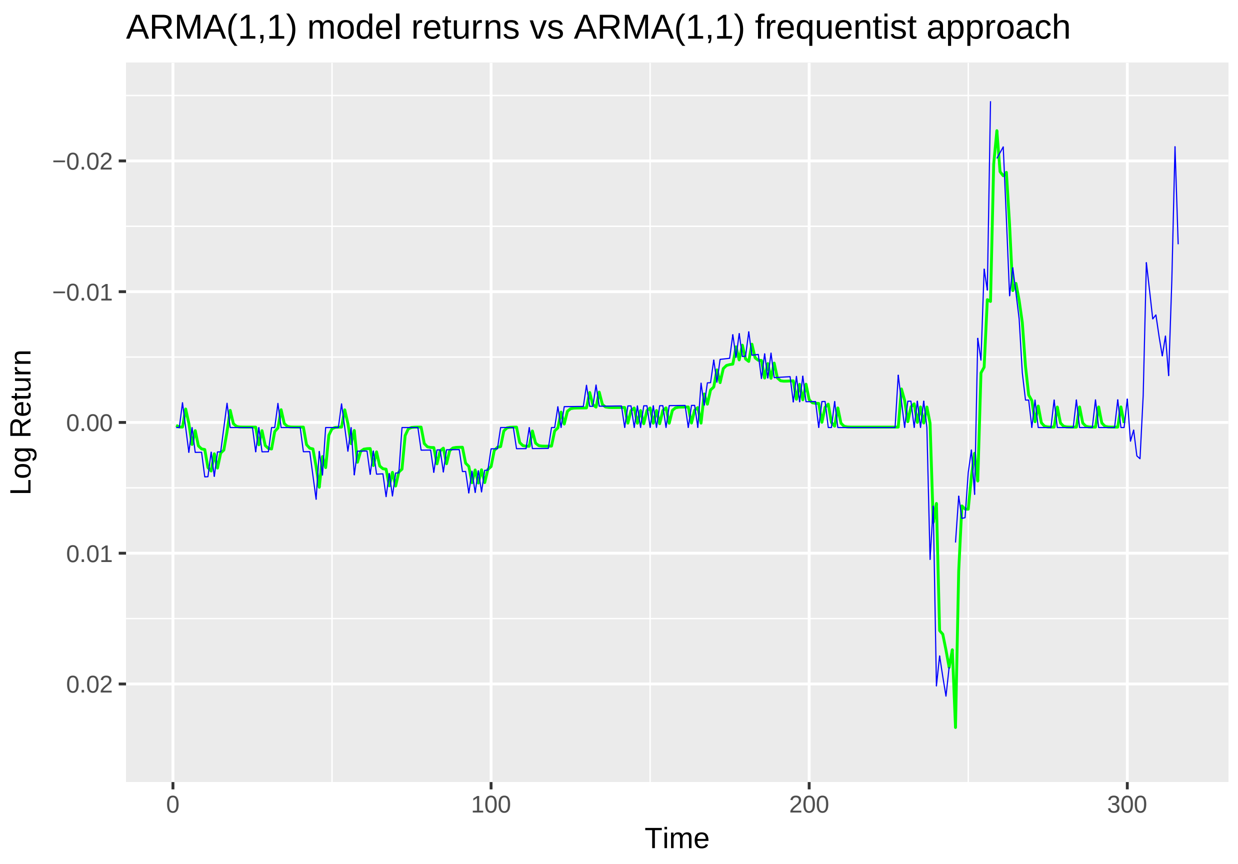


4.2.2 Frequentist Comparison

In order to test obtained results frequentist comparison is used via Arima package, and plot of fitted values indicates a strong model.

```
m1.temp1 <- Arima(ts(unempData$Y),order=c(1,0,1))
m1.temp2 <- data.frame(cbind("Time" = time(ts(unempData$Y)),"V"= fitted(m1.temp1), "O"=m1.temp1$x,"B"=m

f1 <- ggplot() + geom_line(data=m1.temp2, aes(x=Time,y=V), color="green", size=0.5)  + geom_line(data=m
f1
```

```
## Warning: Removed 17 rows containing missing values (geom_path).
```



4.3 DIC

The deviance information criterion (DIC) is a hierarchical modeling generalization of the Akaike information criterion (AIC) and the Bayesian information criterion (BIC). It is only valid when the posterior distribution is approximately multivariate normal. Given 10 rounds model 2 has lower DIC.

```
var1 <- character(1)
var2 <- character(1)
var3 <- character(1)
var4 <- character(1)
for (i in 1:10){
  a <- dic.samples(m1, 10000)
  var1[i] <- sum(a$deviance)
  var2[i] <- sum(a$penalty)

  b <- dic.samples(m2, 10000)
  var3[i] <- sum(b$deviance)
  var4[i] <- sum(b$penalty)
  rm(a)
  rm(b)
}
df <- data.frame(as.numeric(var1),as.numeric(var2),as.numeric(var3),as.numeric(var4))
df1 <- data.frame(mean(df$as.numeric.var1.),mean(df$as.numeric.var2.),mean(df$as.numeric.var3.),mean(df

colnames(df1) <- c("Model1Deviance", "Model1Penalty","Model2Deviance", "Model2Penalty")
df1
```

```
##   Model1Deviance Model1Penalty Model2Deviance Model2Penalty
## 1      -2862.556      6.004011      -2615.831      2.120836
```

5. Predictions

In line with explanation of predictions of the Bayesian Modeling Using WinBugs book by Ntzoufras, last value of our observations was replaced with NA in order to predict it during analysis. In Bayesian analysis, predictions are based on predictive distributions. Predictive distributions are averaged data over all possible parameter values and it allows quantification of our expectation about future. The posterior predictive density of a model is usually used for checking the assumptions of a model and its goodness-of-fit. The main reason is that we can easily generate replicated values from the posterior predictive distribution by adding a single simple step within any MCMC sampler.

In our case it seems model 2 is the most appropriate.

```
orig <- coredata(pred)
m1.pred <- as.list(m1.predYMean[300:316,2])
m2.pred <- as.list(m2.predYMean[300:316,2])

m1.predF <- predict(Arima(ts(unempData$Y[1:300]),order=c(1,0,0)), n.ahead = 16, newxreg = NULL, se.fit
m2.predF <- predict(Arima(ts(unempData$Y[1:300]),order=c(1,0,1)), n.ahead = 16, newxreg = NULL, se.fit

df <- data.frame(cbind("orig"= orig,
                        "Model 1" = m2.pred[[1]], "Model 1 freq" = m2.predF[[1]][1:16],
                        "Model 2" = m1.pred[[1]], "Model 2 freq" = m1.predF[[1]][1:16]))
df1 <- df %>% mutate(resModel1=(Change-Model.1),resModel1freq=(Change-Model.1.freq),
                      resModel2=(Change-Model.2),resModel2freq=(Change-Model.2.freq))
df2 <- df1 %>% select(resModel1,resModel1freq,resModel2,resModel2freq)
df2
```

```
##        resModel1 resModel1freq     resModel2 resModel2freq
## 1  -7.486322e-05  0.0001147482  0.0021856871  0.0000792812
## 2   7.633050e-03  0.0001329902 -0.0010459975  0.0001123253
## 3   5.976161e-03  0.0001499320 -0.0001896596  0.0001416287
## 4  -3.820960e-05 -0.0019642600 -0.0042991982 -0.0019623114
## 5   2.339151e-04  0.0001802789 -0.0023849552  0.0001906593
## 6  -1.235663e-01  0.0001938500  0.0024478401  0.0002110950
## 7  -1.180233e-01  0.0002064538  0.0126146261  0.0002292174
## 8  -1.206262e-01  0.0002181591  0.0104723158  0.0002452882
## 9  -6.186073e-02  0.0002290302  0.0083054025  0.0002595398
## 10 -9.964833e-03  0.0002391263  0.0086157308  0.0002721781
## 11 -3.875125e-02 -0.0018859697  0.0048091990 -0.0018510869
## 12 -3.838011e-02  0.0002572110  0.0054770430  0.0002933245
## 13 -3.912120e-02  0.0002652985  0.0070023849  0.0003021382
## 14 -2.484693e-02 -0.0018662288  0.0018311379 -0.0018290840
## 15 -3.256674e-02  0.0002797851  0.0112715842  0.0003168854
## 16 -3.621406e-02 -0.0018573601  0.0193384126 -0.0018205915
## 17 -3.865977e-02 -0.0020334804  0.0118678156 -0.0020689473
```

6. Conclusion

Convergence plots for all models show that algorhitham reached convergence since histories are stable and there are no major irregularities. Additionally, from running means it may be concluded that burn-in period is sufficient as values are stabilized.

Inference goals seems reached as compared to frequentist approach Bayesian analysis proved appropriate for recovering parameters of interest.

Prediction values seems most appropriate from model 2.

Finally, based on DIC analysis it may be concluded that model 2 in the best way describes data among the presented models.

References

1. (Bayes, 1763) An Essay towards solving a Problem in the Doctrine of Chances
2. (Laplace, 1812) "Théorie analytique des probabilités"
3. (Jeffrys, 1939) Theory of Probability; 3rd edition, Clarendon Press, Oxford
4. (Markov, 1954) Theory of Algorithms
5. (Metropolis&Ulam, 1949) The Monte Carlo Method, Nicholas Metropolis and S. Ulam, Journal of the American Statistical Association, Vol. 44, No. 247 (Sep., 1949), pp. 335-341, https://www.jstor.org/stable/2280232
6. (Metropolis et al.,1953) Equation of State Calculations by Fast Computing Machines, NICHOLAS METROPOLIS, THE JOURNAL OF CHEMICAL PHYSICS, VOLUME 21, NUMBER 6 JUNE, 1953
7. (Hastings, 1970) Monte Carlo Sampling Methods Using Markov Chains and Their Applications. W. K. Hastings. Biometrika, Vol. 57, No. 1. (Apr., 1970), pp. 97-109. Stable URL:https://pdfs.semanticscholar.org/a430/cb602f7d47c21cc4fa94a351ec0c4a9f1fbd.pdf
8. (Geyer, 1992)
9. (Fama 1965) Eugene F. Fama. Journal of Business, Volume 38, Issue 1 (Jan., 1965), 34-105
10. (Mandelbrot 1963) Benoit Mandelbrot. Source: The Journal of Business, Vol. 36, No. 4 (Oct., 1963), pp. 394-419. Published by: The University of Chicago Press. Stable URL: http://www.jstor.org/stable/2350970
11. (Ruppert, 2009) D. Ruppert, Statistics and Data Analysis for Financial Engineering,
12. (Ntzoufras, 2009) I. Ntzoufras, Bayesian Modeling Using WinBugs