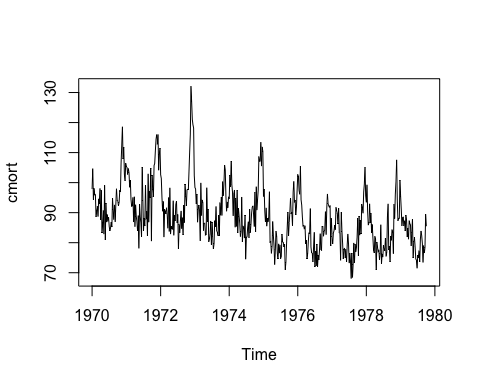
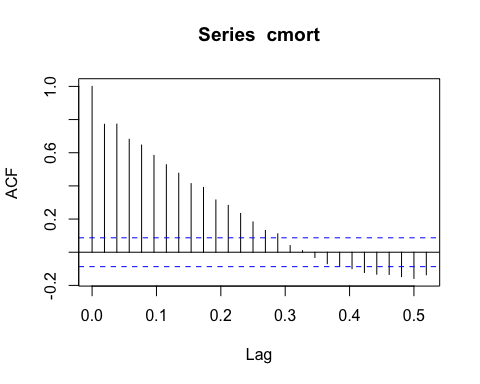
STAT4181 hw4 Min Yang

### A

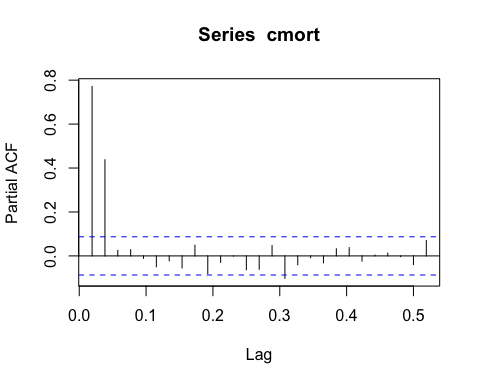
library(astsa)  
library(base)  
ts.plot(cmort)



acf(cmort)



pacf(cmort)



#### 3.

x <- cmort  
AIC\_arma\_seq <- function(x,  
 max\_p=5,  
 max\_q=5  
 ){  
   
MA\_AIC<-rep(0,max\_q)  
 for(i in 1:max\_q){  
 temp <- arima(x,order=c(0,0,i))  
 MA\_AIC[i]<-AIC(temp)  
 }  
   
AR\_AIC <- rep(0,max\_p)  
 for(i in 1:max\_p){  
 temp <- arima(x,order=c(i,0,0))  
 AR\_AIC[i]<-AIC(temp)  
  
 }  
   
 return(list(MA\_AIC,AR\_AIC))  
   
}  
  
AIC\_arma\_seq(x,max\_p=5,max\_q=5)

## [[1]]  
## [1] 3579.353 3415.498 3364.501 3319.428 3294.545  
##   
## [[2]]  
## [1] 3325.885 3217.429 3219.101 3220.675 3222.624

#### 4.

AIC\_arma\_seq <- function(x,  
 max\_p=10,  
 max\_q=10  
 ){  
   
MA\_AIC<-rep(0,max\_q)  
 for(i in 1:max\_q){  
 temp <- arima(x,order=c(0,0,i))  
 MA\_AIC[i]<-AIC(temp)  
 }  
   
AR\_AIC <- rep(0,max\_p)  
 for(i in 1:max\_p){  
 temp <- arima(x,order=c(i,0,0))  
 AR\_AIC[i]<-AIC(temp)  
  
 }  
   
 return(list(AR\_AIC, MA\_AIC))  
   
}  
  
AIC\_arma\_seq(cmort,max\_p=10,max\_q=10)

## [[1]]  
## [1] 3325.885 3217.429 3219.101 3220.675 3222.624 3223.370 3225.000  
## [8] 3225.266 3226.018 3224.640  
##   
## [[2]]  
## [1] 3579.353 3415.498 3364.501 3319.428 3294.545 3271.247 3261.969  
## [8] 3259.239 3240.434 3241.110

With the smalles value of 3217.429, the best model is AR[2].

predict(arima(cmort,order=c(2,0,0)),n.ahead=12)

## $pred  
## Time Series:  
## Start = c(1979, 41)   
## End = c(1979, 52)   
## Frequency = 52   
## [1] 87.66207 86.85311 87.46615 87.37190 87.60258 87.66009 87.78688  
## [8] 87.86685 87.95734 88.03163 88.10362 88.16744  
##   
## $se  
## Time Series:  
## Start = c(1979, 41)   
## End = c(1979, 52)   
## Frequency = 52   
## [1] 5.689543 6.193387 7.148343 7.612531 8.078800 8.405686 8.687550  
## [8] 8.908323 9.092083 9.241328 9.364851 9.466536

### B

#### 1.

bit<-read.csv("~/Desktop/HW2Bitcoin.csv")  
bit$Date <- as.Date(bit$Date, format="%m / %d / %y ")

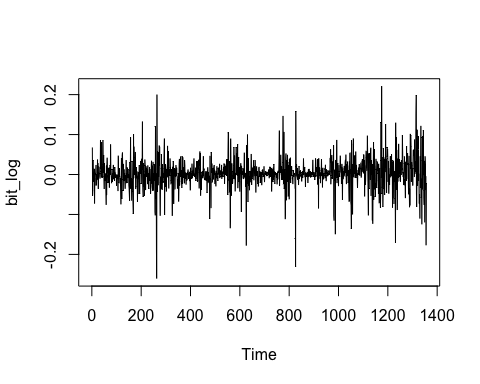
#### 2.

library(zoo)

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

library(xts)  
bit\_zoo <- zoo(bit$Last,order.by = bit$Date)  
bit\_xts <- as.xts(bit\_zoo)  
bit\_log<- diff(log(bit\_xts), lag = 1)   
plot.ts(bit\_log)



### 3.

bitpre17 <- na.omit(bit\_log["/2016"])  
str(bitpre17)#check if get rid of NA

## An 'xts' object on 2014-04-16/2016-12-31 containing:  
## Data: num [1:974, 1] 0.0678 -0.0528 -0.0393 0.0358 -0.0171 ...  
## Indexed by objects of class: [Date] TZ: UTC  
## xts Attributes:   
## List of 1  
## $ na.action:Class 'omit' atomic [1:1] 1  
## .. ..- attr(\*, "index")= num 1.4e+09

### 4.

AR\_MAX\_ORDER <- 10  
AR\_aic <- rep(0,AR\_MAX\_ORDER)  
for(i in 1:AR\_MAX\_ORDER){  
 temp <- arima(bitpre17,order = c(i,0,0))  
 AR\_aic[i] <- AIC(temp)  
}  
AR\_aic

## [1] -3875.685 -3882.865 -3884.334 -3882.393 -3882.351 -3895.490 -3893.514  
## [8] -3892.688 -3895.441 -3895.654

MA\_MAX\_ORDER <- 10  
MA\_aic <- rep(0,MA\_MAX\_ORDER)  
for(i in 1:MA\_MAX\_ORDER){  
 temp <- arima(bitpre17,order = c(0,0,i))  
 MA\_aic[i] <- AIC(temp)  
}  
MA\_aic

## [1] -3875.816 -3882.069 -3882.591 -3881.988 -3881.928 -3896.478 -3894.491  
## [8] -3898.209 -3900.012 -3899.129

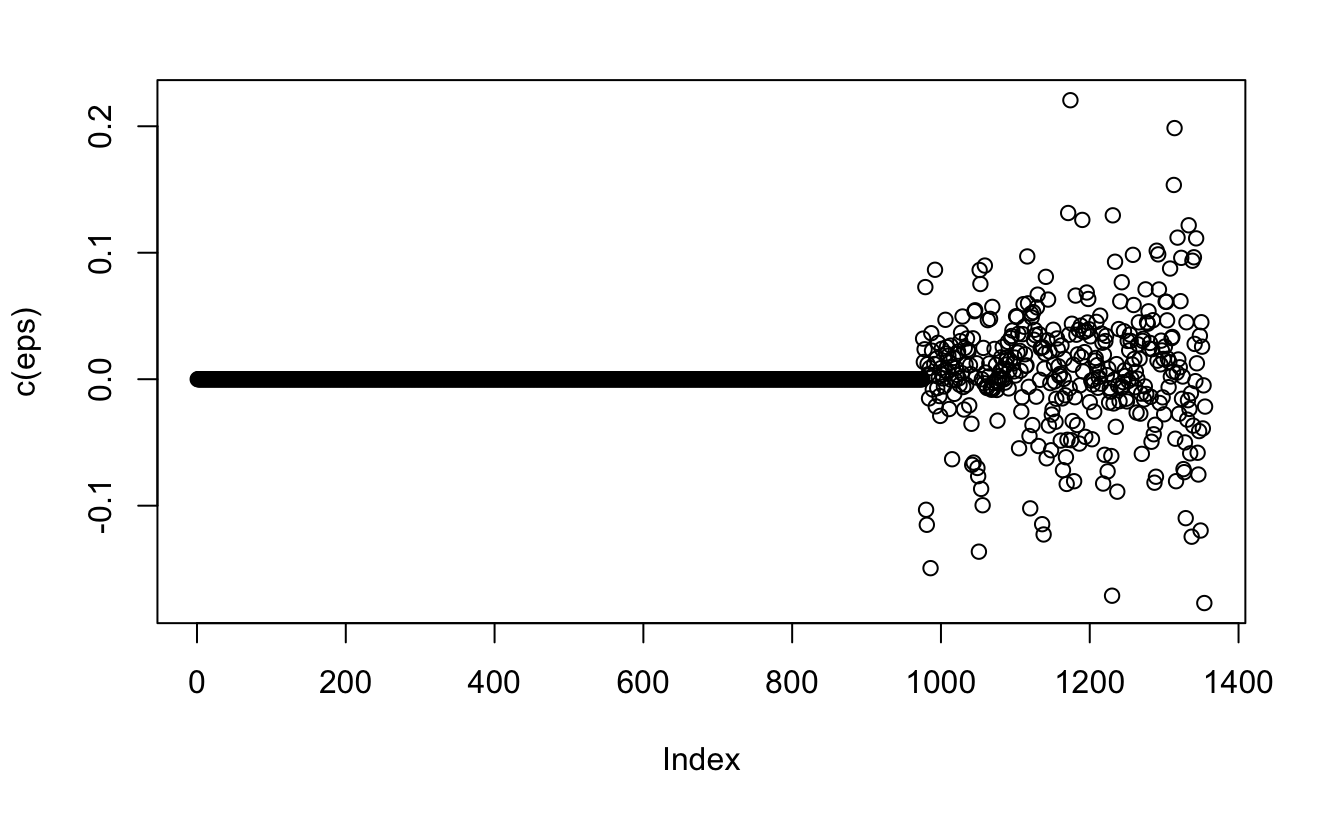
With AIC of -3899.129, the best model is MA[9].

#### 5.

bit\_logg <- na.omit(bit\_log)  
TT <- length(bit\_logg)  
T0 <- 974 #975 data before 2017 - 1 data NA

#I directly plug in the data and order in the function  
backtest\_arma <- function(bit\_logg,   
arima\_order=c(0,0,9),  
T0=975){  
eps <- coredata(bit\_logg)  
eps[1:T0] <- 0  
for(t in (T0+1):TT){ #at every time step  
  
#evaluate previous prediction  
if(t!=(T0+1)){  
eps[t] <- coredata(prev\_pred)-coredata(bit\_logg[t])  
}  
  
#fit model to past data  
temp\_model <- arima(bit\_logg[1:t],order=arima\_order)  
  
#predict from the model  
prev\_pred <- predict(temp\_model,n\_ahead=1)$pred  
}  
  
eps <- eps[(T0+1):TT]  
  
return(c(eps))

plot(c(eps))  
}



From the plot, all eps are within 0.2, so I conclude this model performs pretty well.