R_code.R

Alice Li

Thu Jul 27 10:28:44 2017

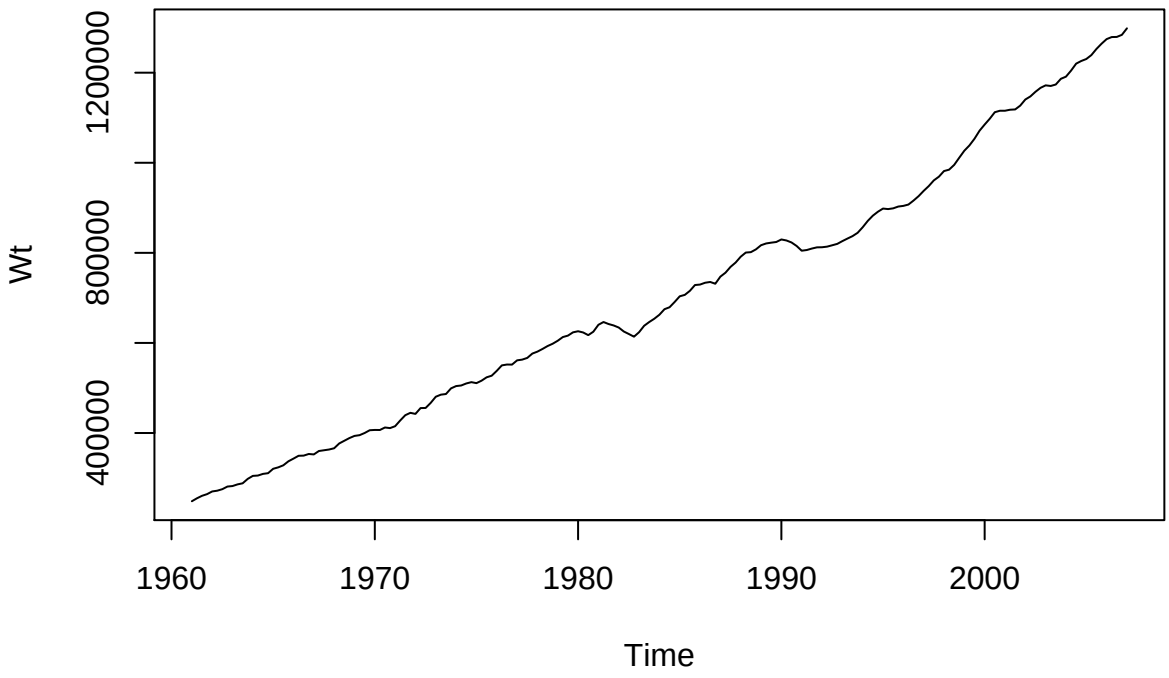
```
rm(list=ls())
require(fGarch)
```

```
## Loading required package: fGarch
## Warning: package 'fGarch' was built under R version 3.4.1
## Loading required package: timeDate
## Loading required package: timeSeries
## Loading required package: fBasics
## Warning: package 'fBasics' was built under R version 3.4.1
##
## Rmetrics Package fBasics
## Analysing Markets and calculating Basic Statistics
## Copyright (C) 2005-2014 Rmetrics Association Zurich
## Educational Software for Financial Engineering and Computational Science
## Rmetrics is free software and comes with ABSOLUTELY NO WARRANTY.
## https://www.rmetrics.org --- Mail to: info@rmetrics.org
```

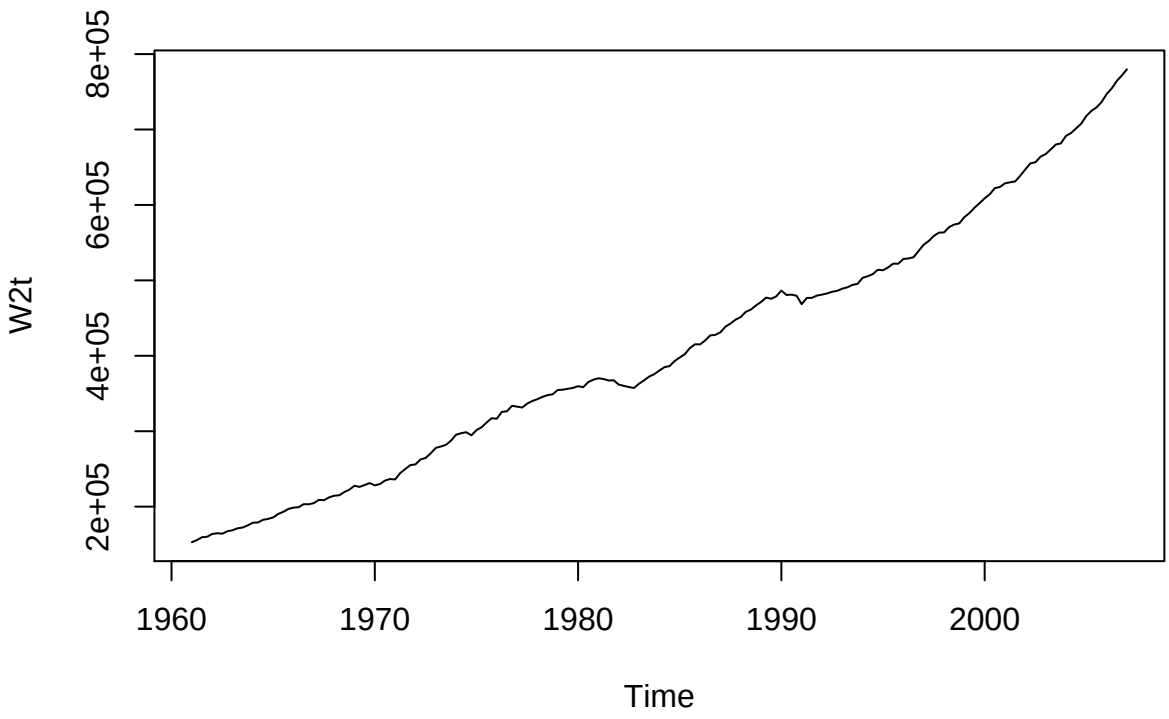
```
#1)
library(readxl)
```

```
## Warning: package 'readxl' was built under R version 3.4.1
```

```
CPE.data <- read_excel("C:/alice/Waterloo Stuff/STUDYMATERIALS/05. STAT443/Project/GDP_CONS_Canada.xls")
Wt <- as.numeric(as.character(CPE.data$GDP))
Wt2 <- as.numeric(as.character(CPE.data$CONS))
xlab_years <- seq(from=1961, to=2007, by= 0.25)
plot(xlab_years, Wt, xlab = "Time", ylab = "Wt", type = "l")
```




```
plot(xlab_years, Wt2, xlab = "Time", ylab = "W2t", type = "l")
```



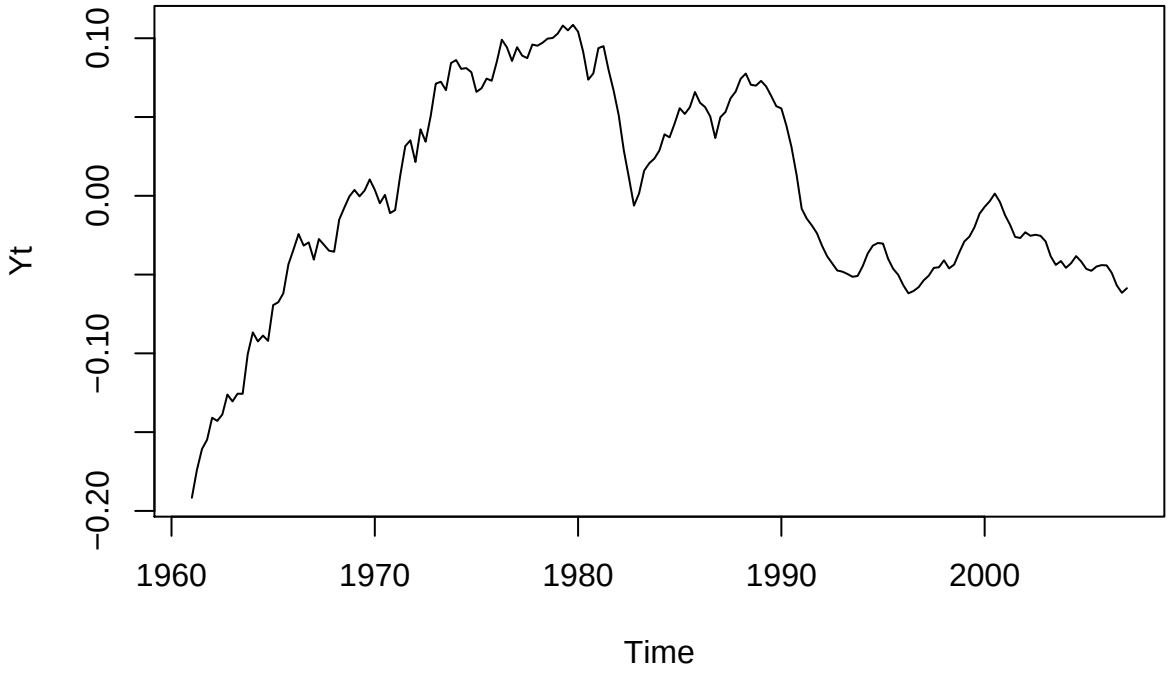

```
#2)
#Convert into log form and plot
Xt <- log(Wt)

#TS
#linear regression of Xt
tSeq <- 1:length(Xt)
model1 <- lm(Xt ~ tSeq)
summary(model1)
```

```
##
## Call:
## lm(formula = Xt ~ tSeq)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.191639 -0.043878 -0.008175  0.059031  0.108496
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.261e+01  9.711e-03 1298.21   <2e-16 ***
## tSeq        8.263e-03  9.055e-05   91.26   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06577 on 183 degrees of freedom
```

```
## Multiple R-squared:  0.9785, Adjusted R-squared:  0.9784
## F-statistic:  8328 on 1 and 183 DF,  p-value: < 2.2e-16
```

```
#Yt is the residuals of regression
Yt <- model1$residuals
plot(xlab_years, Yt, xlab = "Time", ylab = "Yt", type = "l")
```

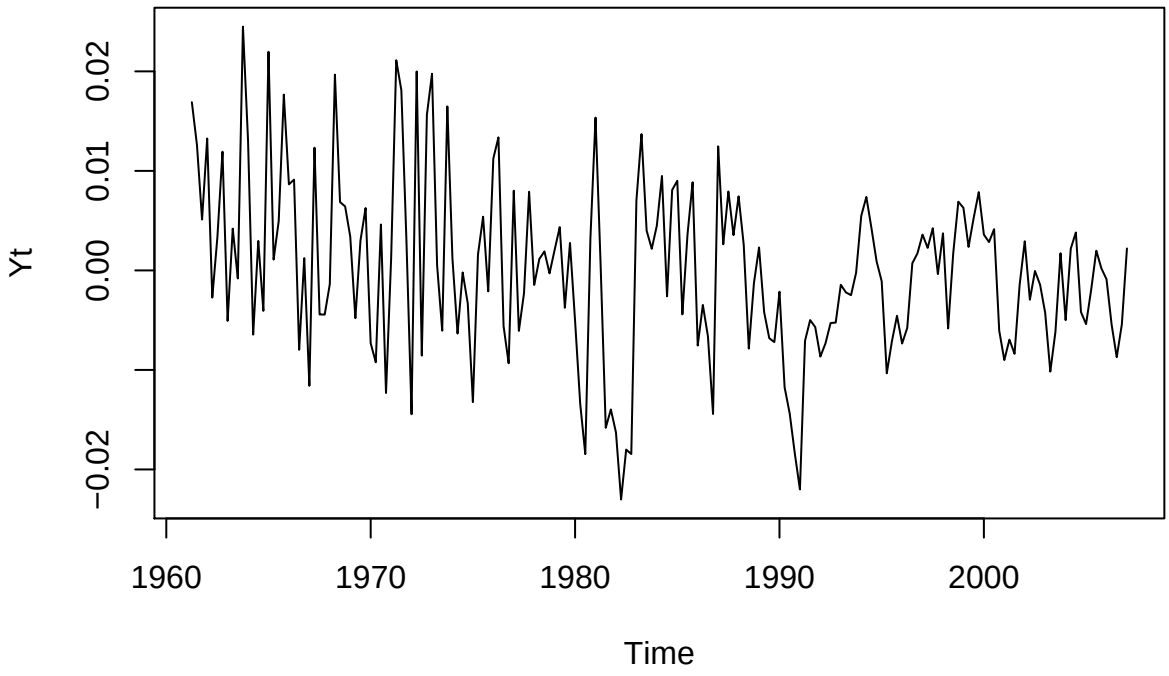



```
#DS
dX <- diff(Xt)
model2 <- lm(dX~1)
Yt2 <- model2$residuals
summary(model2)
```

```
##
## Call:
## lm(formula = dX ~ 1)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0230228 -0.0057192  0.0000702  0.0043923  0.0244942
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.0089863  0.0006494   13.84   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.00881 on 183 degrees of freedom
```

```
xlab_years2 <- xlab_years[-1]
plot(xlab_years2, Yt2, xlab = "Time", ylab = "Yt",type = "l")
```




```
#3)
#function for BIC
BIC <- function(res, k, N){
  bic <- log(sum(res ^2) / N)
  bic <- bic + log(N) * k / N
}

#TS
#obtaining the bic
bic.array <- rep(NA, 9)
N <- length(Yt)
for (ii in 0:9){
  model.arima <- arima(Yt, order = c(ii, 0, 0))
  res.arima <- model.arima$residuals
  bic.array[ii + 1] <- BIC(res.arima, ii, N)
}

bic.array
```

```
##  [1] -5.453970 -9.437384 -9.527164 -9.503044 -9.492188 -9.464028 -9.436557
##  [8] -9.413465 -9.391608 -9.371411
```

```r
which(bic.array ==  min(bic.array))
```

```
## [1] 3
```

```r
#the above result gives p = 2 is the best model for TS


#DS
bic.array2 <- rep(NA, 9)
N2 <- length(Yt2)
for (ii in 0:9){
  model.arima <- arima(Yt2, order = c(ii, 0, 0))
  res.arima <- model.arima$residuals
  bic.array2[ii + 1] <- BIC(res.arima, ii, N2)
}
bic.array2
```

```
##  [1] -9.469291 -9.554756 -9.529794 -9.517196 -9.488917 -9.461730 -9.437722
##  [8] -9.414835 -9.395702 -9.413598
```

```r
which(bic.array2 ==  min(bic.array2))
```

```
## [1] 2
```

```r
#the above result gives p = 1 is the best model for DS


#Let's try AR(2) for TS
# Yt = Yt-1 +  Yt-2
TSmodel.ar2 <- lm(Yt[-(1:2)] ~ Yt[-c(1, N)] + Yt[-c(N-1,N)] - 1)
summary(TSmodel.ar2)
```

```
##
## Call:
## lm(formula = Yt[-(1:2)] ~ Yt[-c(1, N)] + Yt[-c(N - 1, N)] - 1)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0162392 -0.0048806  0.0001241  0.0045117  0.0255097
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## Yt[-c(1, N)]       1.28986    0.07000   18.43  < 2e-16 ***
## Yt[-c(N - 1, N)] -0.31314    0.06852   -4.57 8.99e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.008157 on 181 degrees of freedom
## Multiple R-squared:  0.9834, Adjusted R-squared:  0.9832
## F-statistic:  5355 on 2 and 181 DF,  p-value: < 2.2e-16
```

```r
TSsigma2.hat <- sum(TSmodel.ar2$residuals^2)/N
TSsigma2.hat
```

```
## [1] 6.509913e-05
```

```r
sd <- sqrt(TSsigma2.hat)
sd
```

```
## [1] 0.008068403
```

```r
#psi
TSphi1 <- TSmodel.ar2$coefficients[1]
TSphi2 <- TSmodel.ar2$coefficients[2]

psi_array <- rep(NA, 185)
psi_array[1] <- 1
psi_array[2] <- TSphi1
for (ii in 3:185){
  psi_array[ii] <- TSphi1 * psi_array[ii-1] + TSphi2 * psi_array[ii-2]
}
psi_array[1:9]
```

```
## [1] 1.000000 1.289855 1.350590 1.338165 1.303121 1.261809 1.219497 1.177856
## [9] 1.137395
```

```r
#pho
pho_array <- rep(NA, 185)
pho_array[1] <- 1
pho_array[2] <- TSphi1/(1-TSphi2)
for (ii in 3:185){
  pho_array[ii] <- TSphi1 * pho_array[ii-1] + TSphi2 * pho_array[ii-2]
}
pho_array[1:9]
```

```
## [1] 1.0000000 0.9822707 0.9538508 0.9227449 0.8915221 0.8609896 0.8313841
## [8] 0.8027581 0.7751053
```

```r
#sd
TSgamma <- TSsigma2.hat/(1-pho_array[2]*TSphi1 - pho_array[3]*TSphi2)
TSgamma
```

```
## Yt[-c(1, N)]
##  0.002053715
```

```r
sqrt(TSgamma)
```

```
## Yt[-c(1, N)]
##   0.04531793
```

```r
#4)
#DS
DSgrowthrate <- coefficients(model2)[["(Intercept)"]]
DSmodel.ar2 <- lm(Yt2[-(1:2)] ~ Yt2[-c(1, N2)] + Yt2[-c(N2-1,N2)] - 1)
summary(DSmodel.ar2)
```

```
##
## Call:
## lm(formula = Yt2[-(1:2)] ~ Yt2[-c(1, N2)] + Yt2[-c(N2 - 1, N2)] -
##     1)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0173860 -0.0056604 -0.0001698  0.0041378  0.0244944
```

```
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## Yt2[-c(1, N2)]       0.29695    0.07430   3.997 9.36e-05 ***
## Yt2[-c(N2 - 1, N2)]  0.05743    0.07363   0.780    0.436
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.008275 on 180 degrees of freedom
## Multiple R-squared:  0.104,  Adjusted R-squared:  0.09407
## F-statistic: 10.45 on 2 and 180 DF,  p-value: 5.088e-05
```

```
#psi
DSphi1 <- DSmodel.ar2$coefficients[1]
DSphi2 <- DSmodel.ar2$coefficients[2]
#Forecast of Yt array
DSForecast_Yt <- rep(NA, N2)
DSForecast_Yt[1] <- Yt2[N2]
DSForecast_Yt[2] <- DSphi1*Yt2[N2] + DSphi2 * Yt2[N2-1]
for (ii in 3:N2){
  DSForecast_Yt[ii] <- DSphi1 * DSForecast_Yt[ii-1] + DSphi2 * DSForecast_Yt[ii-2]
}
DSForecast_Yt[1:9]
```

```
## [1] 2.206601e-03 3.438549e-04 2.288366e-04 8.770186e-05 3.918574e-05
## [6] 1.667317e-05 7.201641e-06 3.096113e-06 1.333001e-06
```

```
DSForecast_DeltaXt<- DSgrowthrate + DSForecast_Yt
DSForecast_DeltaXt[1:9]
```

```
## [1] 0.011192893 0.009330146 0.009215128 0.009073993 0.009025477 0.009002965
## [7] 0.008993493 0.008989387 0.008987624
```

```
#Find psi for DS and then find Var of Forecast
DSsigma2.hat <- sum(DSmodel.ar2$residuals^2)/N2
DSsigma2.hat
```

```
## [1] 6.699001e-05
```

```
DSpsi_array <- rep(NA, N2)
DSpsi_array[1] <- 1
DSpsi_array[2] <- DSphi1
for (ii in 3:N2){
  DSpsi_array[ii] <- DSphi1 * DSpsi_array[ii-1] + DSphi2 * DSpsi_array[ii-2]
}
DSpsi_array[1:9]
```

```
## [1] 1.0000000000 0.2969538375 0.1456126741 0.0602946256 0.0262674154
## [6] 0.0112629960 0.0048531563 0.0020880095 0.0008987645
```

```
#pho
DSpho_array <- rep(NA, 185)
DSpho_array[1] <- 1
DSpho_array[2] <- DSphi1/(1-DSphi2)
for (ii in 3:185){
  DSpho_array[ii] <- DSphi1 * DSpho_array[ii-1] + DSphi2 * DSpho_array[ii-2]
}
DSpho_array[1:9]
```

```
## [1] 1.0000000000 0.3150473511 0.1509856123 0.0629292705 0.0273583570
## [6] 0.0117382659 0.0050569434 0.0021758202 0.0009365439
```

```
DSForecast_VarXt <- rep(NA, N2)
DSForecast_VarXt[1] <- 0
DSForecast_VarXt[2] <- DSsigma2.hat
for (ii in 3:N2){
  DSForecast_VarXt[ii] <-
    DSsigma2.hat * sum((DSpsi_array[1:ii-1])^2)
}
DSForecast_VarXt[1:9]
```

```
## [1] 0.000000e+00 6.699001e-05 7.289730e-05 7.431769e-05 7.456123e-05
## [6] 7.460745e-05 7.461595e-05 7.461752e-05 7.461782e-05
```

```
#plotting
lwd_DSForecast <- DSForecast_DeltaXt-1.96*sqrt(DSForecast_VarXt)
upr_DSForecast <- DSForecast_DeltaXt+1.96*sqrt(DSForecast_VarXt)
lwd_DSForecast[1:9]
```

```
## [1]  0.011192893 -0.006711949 -0.007519335 -0.007822717 -0.007898896
## [6] -0.007926654 -0.007937089 -0.007941374 -0.007943170
```

```
upr_DSForecast[1:9]
```

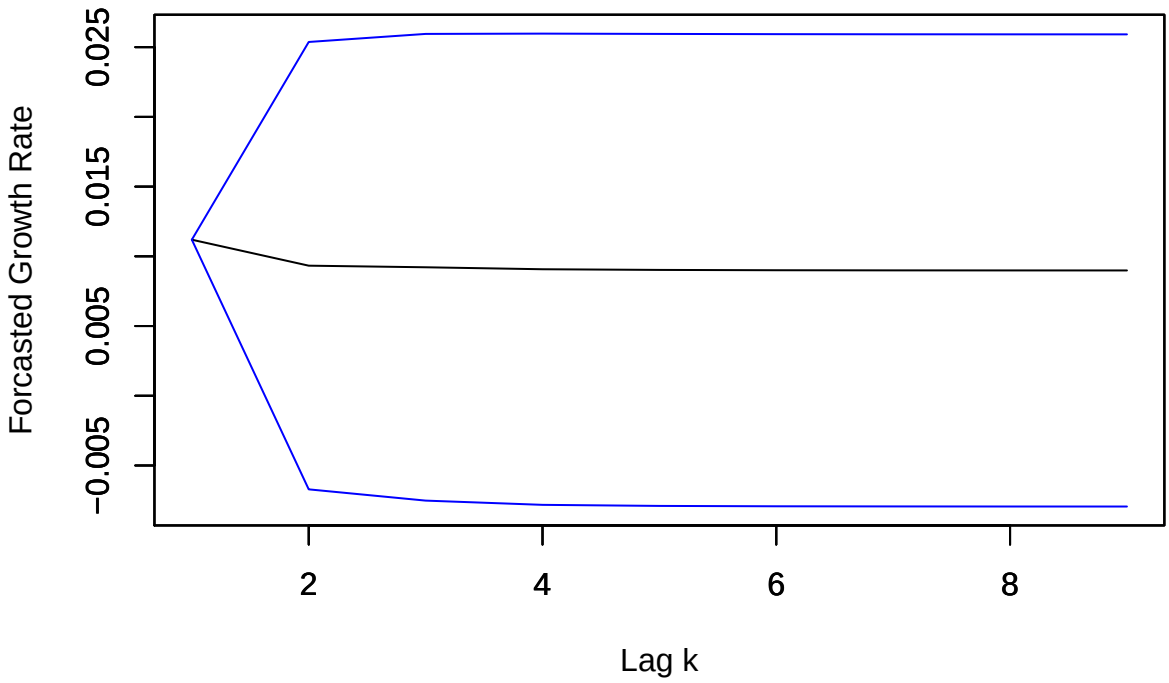
```
## [1] 0.01119289 0.02537224 0.02594959 0.02597070 0.02594985 0.02593258
## [7] 0.02592408 0.02592015 0.02591842
```

```
DSmatrix <- cbind(DSForecast_Yt[1:9], DSForecast_DeltaXt[1:9], DSForecast_VarXt[1:9], DSForecast_VarXt[
                  lwd_DSForecast[1:9], upr_DSForecast[1:9])
DSmatrix
```

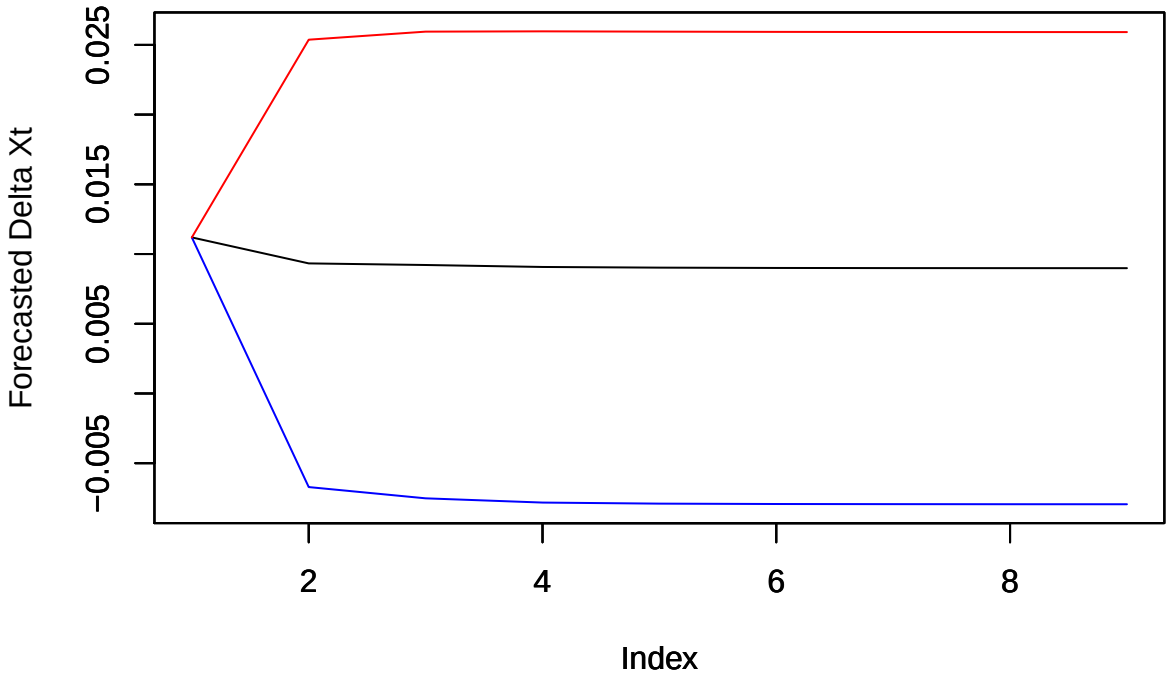
```
##               [,1]        [,2]         [,3]         [,4]         [,5]
##  [1,] 2.206601e-03 0.011192893 0.000000e+00 0.000000e+00  0.011192893
##  [2,] 3.438549e-04 0.009330146 6.699001e-05 6.699001e-05 -0.006711949
##  [3,] 2.288366e-04 0.009215128 7.289730e-05 7.289730e-05 -0.007519335
##  [4,] 8.770186e-05 0.009073993 7.431769e-05 7.431769e-05 -0.007822717
##  [5,] 3.918574e-05 0.009025477 7.456123e-05 7.456123e-05 -0.007898896
##  [6,] 1.667317e-05 0.009002965 7.460745e-05 7.460745e-05 -0.007926654
##  [7,] 7.201641e-06 0.008993493 7.461595e-05 7.461595e-05 -0.007937089
##  [8,] 3.096113e-06 0.008989387 7.461752e-05 7.461752e-05 -0.007941374
##  [9,] 1.333001e-06 0.008987624 7.461782e-05 7.461782e-05 -0.007943170
##             [,6]
##  [1,] 0.01119289
##  [2,] 0.02537224
##  [3,] 0.02594959
##  [4,] 0.02597070
##  [5,] 0.02594985
##  [6,] 0.02593258
##  [7,] 0.02592408
##  [8,] 0.02592015
##  [9,] 0.02591842
```

```
yLim <- range(lwd_DSForecast[1:9],upr_DSForecast[1:9])
plot(DSForecast_DeltaXt[1:9], type="l", ylim = yLim, ylab ="Forcasted Growth Rate", xlab = "Lag k")
par(new=TRUE)
```

```
plot(lwd_DSForecast[1:9], type="l", ylab ="", ylim = yLim, xlab = "", col = "blue")
par(new=TRUE)
plot(upr_DSForecast[1:9], type="l", ylab="",  ylim = yLim, xlab = "", col = "blue")
```




```
#sub
yLim <- range(lwd_DSForecast[1:9],upr_DSForecast[1:9])
plot(DSForecast_DeltaXt[1:9], type="l", ylim = yLim, ylab ="Forecasted Delta Xt")
par(new=TRUE)
plot(lwd_DSForecast[1:9], type="l", ylab ="", ylim = yLim, col = "blue")
par(new=TRUE)
plot(upr_DSForecast[1:9], type="l", ylab="",  ylim = yLim, col = "red")
```




```
#TS
TSgrowthrate <- coefficients(model1)[2]
TSForecast_Yt <- rep(NA, 185)
TSForecast_Yt[1] <- Yt[N]
TSForecast_Yt[2] <- TSphi1*Yt[N] + TSphi2 * Yt[N-1]
for (ii in 3:185){
  TSForecast_Yt[ii] <- TSphi1 * TSForecast_Yt[ii-1] + TSphi2 * TSForecast_Yt[ii-2]
}
Yt[N-1]
```

```
##         184
## -0.06157347
```

```
Yt[N]
```

```
##         185
## -0.05864407
```

```
TSForecast_Yt[1:9]
```

```
## [1] -0.05864407 -0.05636147 -0.05433456 -0.05243490 -0.05061931 -0.04887232
## [7] -0.04718748 -0.04556132 -0.04399140
```

```
TSForecast_DeltaXt <- rep(NA, 185)
TSForecast_DeltaXt[1] <- TSgrowthrate + TSForecast_Yt[1] - Yt[N-1]
for (ii in 2:185){
  TSForecast_DeltaXt[ii] <- TSgrowthrate +  TSForecast_Yt[ii] - TSForecast_Yt[ii-1]
}
TSForecast_DeltaXt[1:9]
```

```
## [1] 0.011192893 0.010546086 0.010290406 0.010163155 0.010079082 0.010010487
## [7] 0.009948336 0.009889650 0.009833414
```

```
TSForecast_VarXt <- rep(NA, 185)
TSForecast_VarXt[1] <- 0  #X0
TSForecast_VarXt[2] <- TSsigma2.hat #X1

for (k in 3:185){
  sum<-0
  for  (ii in 2:(k-1)) {
    sum <- sum + (psi_array[ii] - psi_array[ii-1])^2
  }
  TSForecast_VarXt[k] <- TSsigma2.hat * (1+sum)
}

TSForecast_VarXt[1:9]
```

```
## [1] 0.000000e+00 6.509913e-05 7.056850e-05 7.080863e-05 7.081868e-05
## [6] 7.089863e-05 7.100973e-05 7.112628e-05 7.123916e-05
```

```
sqrt(TSForecast_VarXt[1:9])
```

```
## [1] 0.000000000 0.008068403 0.008400506 0.008414786 0.008415383 0.008420132
## [7] 0.008426727 0.008433640 0.008440329
```

```
TSForecast_DeltaXt[1:9]-1.96*sqrt(TSForecast_VarXt[1:9])
```

```
## [1]  0.011192893 -0.005267985 -0.006174585 -0.006329826 -0.006415069
## [6] -0.006492972 -0.006568049 -0.006640284 -0.006709631
```

```
TSForecast_DeltaXt[1:9]+1.96*sqrt(TSForecast_VarXt[1:9])
```

```
## [1] 0.01119289 0.02636016 0.02675540 0.02665614 0.02657323 0.02651395
## [7] 0.02646472 0.02641958 0.02637646
```

```
TSForecast_VarYt <- rep(NA, N)
TSForecast_VarYt[1] <- 0
TSForecast_VarYt[2] <- TSsigma2.hat
for (ii in 3:N){
  TSForecast_VarYt[ii] <- TSsigma2.hat * sum((psi_array[1:ii-1])^2)
}

TSForecast_VarXt[1:9]
```

```
## [1] 0.000000e+00 6.509913e-05 7.056850e-05 7.080863e-05 7.081868e-05
## [6] 7.089863e-05 7.100973e-05 7.112628e-05 7.123916e-05
```

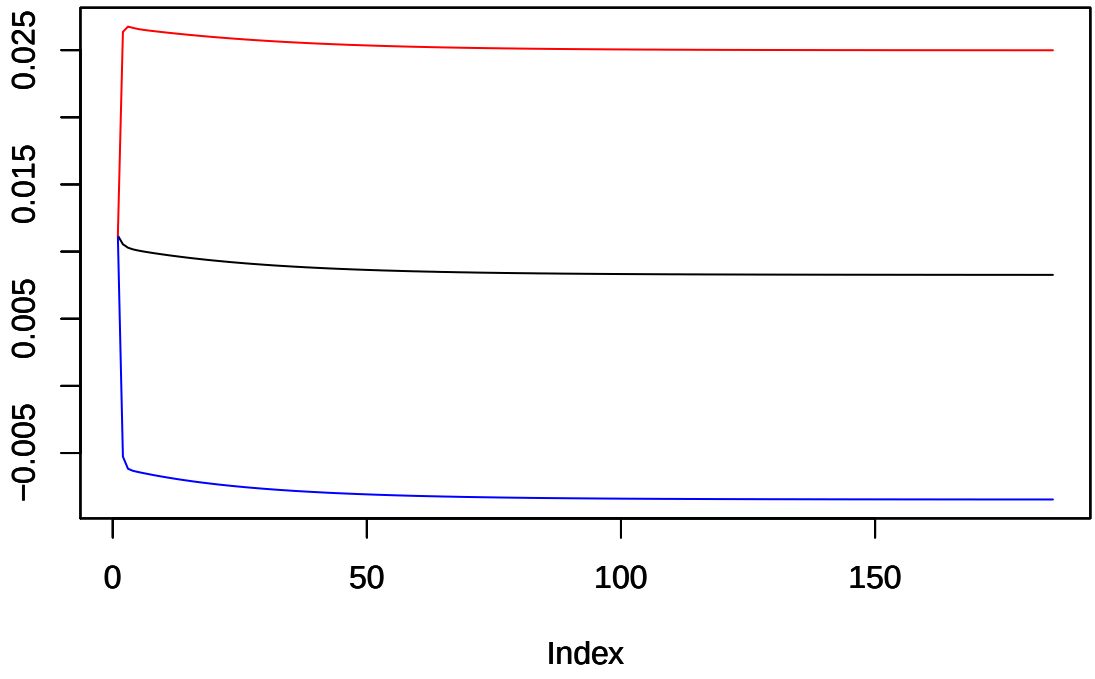
```
lwd_TSForecast <- TSForecast_DeltaXt-1.96*sqrt(TSForecast_VarXt)
upr_TSForecast <- TSForecast_DeltaXt+1.96*sqrt(TSForecast_VarXt)
TSmatrix <- cbind(TSForecast_Yt[1:9], TSForecast_DeltaXt[1:9], TSForecast_VarYt[1:9], TSForecast_VarXt[
                  lwd_TSForecast[1:9], upr_TSForecast[1:9])
TSmatrix
```

```
##              [,1]        [,2]         [,3]         [,4]         [,5]
##  [1,] -0.05864407 0.011192893 0.000000e+00 0.000000e+00  0.011192893
##  [2,] -0.05636147 0.010546086 6.509913e-05 6.509913e-05 -0.005267985
##  [3,] -0.05433456 0.010290406 1.734063e-04 7.056850e-05 -0.006174585
```

```
##  [4,] -0.05243490 0.010163155 2.921531e-04 7.080863e-05 -0.006329826
##  [5,] -0.05061931 0.010079082 4.087253e-04 7.081868e-05 -0.006415069
##  [6,] -0.04887232 0.010010487 5.192717e-04 7.089863e-05 -0.006492972
##  [7,] -0.04718748 0.009948336 6.229201e-04 7.100973e-05 -0.006568049
##  [8,] -0.04556132 0.009889650 7.197337e-04 7.112628e-05 -0.006640284
##  [9,] -0.04399140 0.009833414 8.100487e-04 7.123916e-05 -0.006709631
##             [,6]
##  [1,] 0.01119289
##  [2,] 0.02636016
##  [3,] 0.02675540
##  [4,] 0.02665614
##  [5,] 0.02657323
##  [6,] 0.02651395
##  [7,] 0.02646472
##  [8,] 0.02641958
##  [9,] 0.02637646
```

```r
#plotting

yLim <- range(lwd_TSForecast,upr_TSForecast)
plot(TSForecast_DeltaXt, type="l", ylim = yLim, ylab ="")
par(new=TRUE)
plot(lwd_TSForecast, type="l", ylab ="", ylim = yLim, col = "blue")
par(new=TRUE)
plot(upr_TSForecast, type="l", ylab="",  ylim = yLim, col = "red")
```

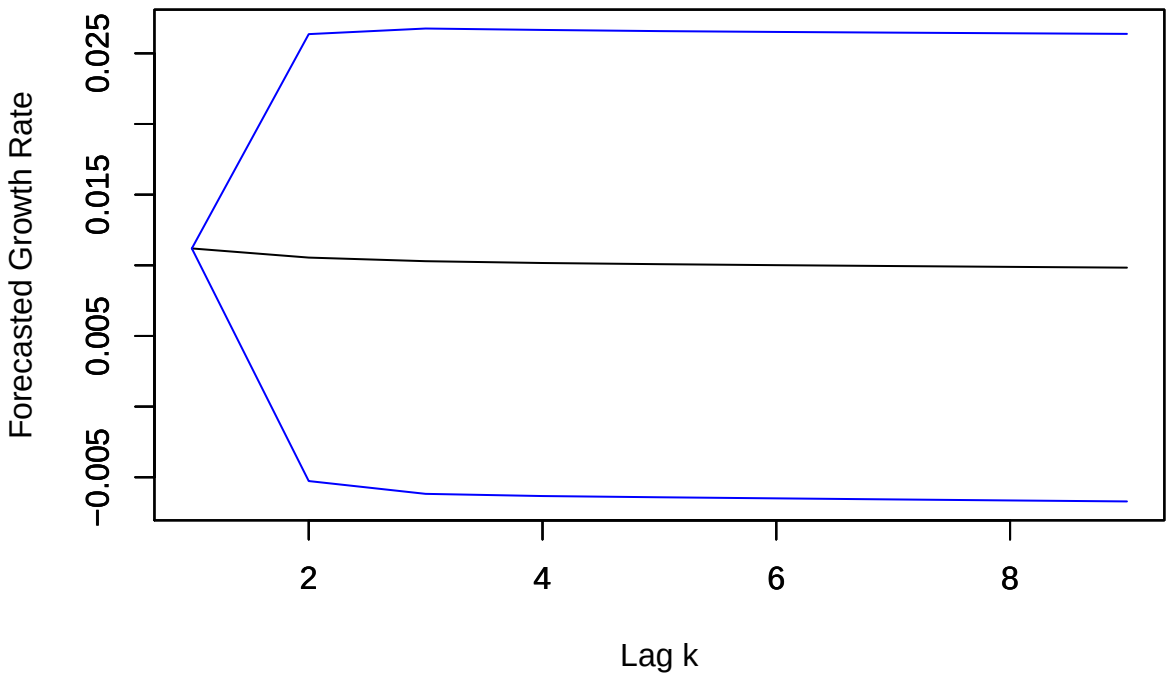


```r
#sub
yLim <- range(lwd_TSForecast[1:9],upr_TSForecast[1:9])
plot(TSForecast_DeltaXt[1:9], type="l", ylim = yLim, ylab ="Forecasted Growth Rate", xlab = "Lag k")
par(new=TRUE)
plot(lwd_TSForecast[1:9], type="l", ylab ="", ylim = yLim,xlab = "",  col = "blue")
par(new=TRUE)
plot(upr_TSForecast[1:9], type="l", ylab="",  ylim = yLim, xlab = "", col = "blue")

###################
#5)
require(tseries)
```

```
## Loading required package: tseries
```

```
## Warning: package 'tseries' was built under R version 3.4.1
```




```r
adf.test(Xt,k=5)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  Xt
## Dickey-Fuller = -2.6929, Lag order = 5, p-value = 0.2865
## alternative hypothesis: stationary
```

```r
#6)
######TS##########
#Box-Jenkins
```

```r
acf(Yt)$acf
```

```
## , , 1
##
##             [,1]
##  [1,] 1.0000000
##  [2,] 0.9656024
##  [3,] 0.9293277
##  [4,] 0.8937737
##  [5,] 0.8570070
##  [6,] 0.8224138
##  [7,] 0.7869730
##  [8,] 0.7510593
##  [9,] 0.7160919
## [10,] 0.6805314
## [11,] 0.6434719
## [12,] 0.6038760
## [13,] 0.5713205
## [14,] 0.5418860
## [15,] 0.5092971
## [16,] 0.4789705
## [17,] 0.4487886
## [18,] 0.4225141
## [19,] 0.3965441
## [20,] 0.3709504
## [21,] 0.3470672
## [22,] 0.3243639
## [23,] 0.3053780
```

```r
2/sqrt(length(Yt))
```

```
## [1] 0.1470429
```

```r
#all rho(K) sig, therefore AR(q)
TSmodel.ar1 <- lm(Yt[-1] ~ Yt[-length(Yt)] - 1)
phi_11 <- as.numeric(TSmodel.ar1$coefficients)[1]
phi_11
```

```
## [1] 0.9698154
```

```r
phi_22 <- as.numeric(TSmodel.ar2$coefficients)[2]
phi_22
```

```
## [1] -0.313136
```

```r
TSmodel.ar3 <- lm(Yt[-(1:3)] ~ Yt[-c(1:2, N)] + Yt[-c(1, (N-1):N)] + Yt[-c((N-2):N)] - 1)
phi_33 <- as.numeric(TSmodel.ar3$coefficients)[3]
phi_33
```

```
## [1] -0.05518801
```

```r
TSmodel.ar4 <- lm(Yt[-(1:4)] ~ Yt[-c(1:3, N)] + Yt[-c(1:2, N-1,N)]
                + Yt[-c(1, (N-2):N)] + Yt[-c((N-3):N)] - 1)
phi_44 <- as.numeric(TSmodel.ar4$coefficients)[4]
phi_44
```

```
## [1] -0.1222834
```

```
TSmodel.ar5 <- lm(Yt[-(1:5)] ~ Yt[-c(1:4, N)] + Yt[-c(1:3, (N-1): N)]
                  + Yt[-c(1:2, (N-2):N)] + + Yt[-c(1, (N-3):N)] + Yt[-c((N-4):N)] - 1)
phi_55 <- as.numeric(TSmodel.ar5$coefficients)[5]
phi_55
```

```
## [1] 0.006423765
```

```
TSmodel.ar6 <- lm(Yt[-(1:6)] ~ Yt[-c(1:5, N)] + Yt[-c(1:4, N-1,N)]
                  + Yt[-c(1:3, (N-2):N)] + Yt[-c(1:2, (N-3):N)]
                  + Yt[-c(1, (N-4):N)] + Yt[-c((N-5):N)] - 1)
phi_66 <- as.numeric(TSmodel.ar6$coefficients)[6]
phi_66
```

```
## [1] 0.03168757
```

```
TSmodel.ar7 <- lm(Yt[-(1:7)] ~ Yt[-c(1:6, N)] + Yt[-c(1:5, N-1,N)]
                  + Yt[-c(1:4, (N-2):N)] + Yt[-c(1:3, (N-3):N)]
                  + Yt[-c(1:2, (N-4):N)] + Yt[-c(1, (N-5):N)] + Yt[-c((N-6):N)]- 1)
phi_77 <- as.numeric(TSmodel.ar7$coefficients)[7]
phi_77
```

```
## [1] -0.06549344
```

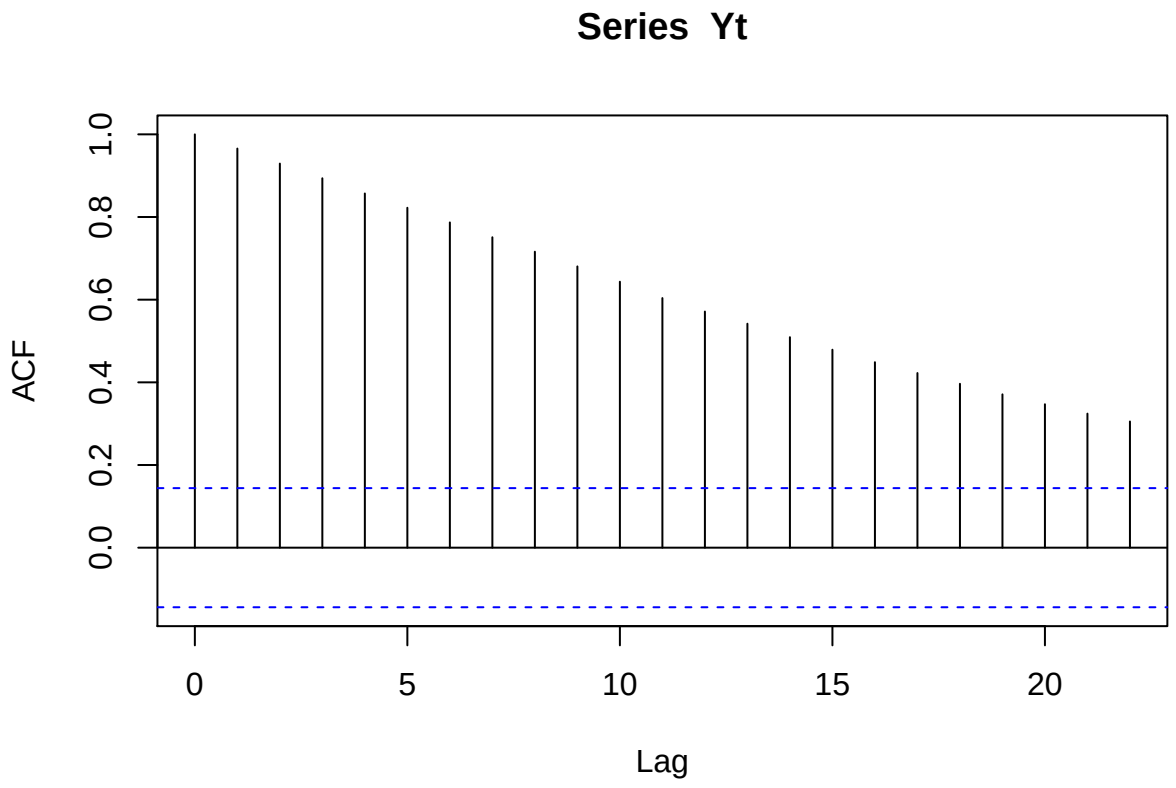
```
TSmodel.ar8 <- lm(Yt[-(1:8)] ~ Yt[-c(1:7, N)] + Yt[-c(1:6, N-1,N)]
                  + Yt[-c(1:5, (N-2):N)] + Yt[-c(1:4, (N-3):N)]
                  + Yt[-c(1:3, (N-4):N)] + Yt[-c(1:2, (N-5):N)] +  Yt[-c(1, (N-6):N)] +  Yt[-c((N-7):N)]
phi_88 <- as.numeric(TSmodel.ar8$coefficients)[8]
phi_88
```

```
## [1] -0.0746004
```

```
TSmodel.ar9 <- lm(Yt[-(1:9)] ~ Yt[-c(1:8, N)] + Yt[-c(1:7, N-1,N)]
                  + Yt[-c(1:6, (N-2):N)] + Yt[-c(1:5, (N-3):N)]
                  + Yt[-c(1:4, (N-4):N)] + Yt[-c(1:3, (N-5):N)]
                  + Yt[-c(1:2, (N-6):N)] + Yt[-c(1, (N-7):N)] +  Yt[-c((N-8):N)]- 1)
phi_99 <- as.numeric(TSmodel.ar9$coefficients)[9]
phi_99
```

```
## [1] 0.08906843
```

```
acf(Yt)$acf[1:10]
```

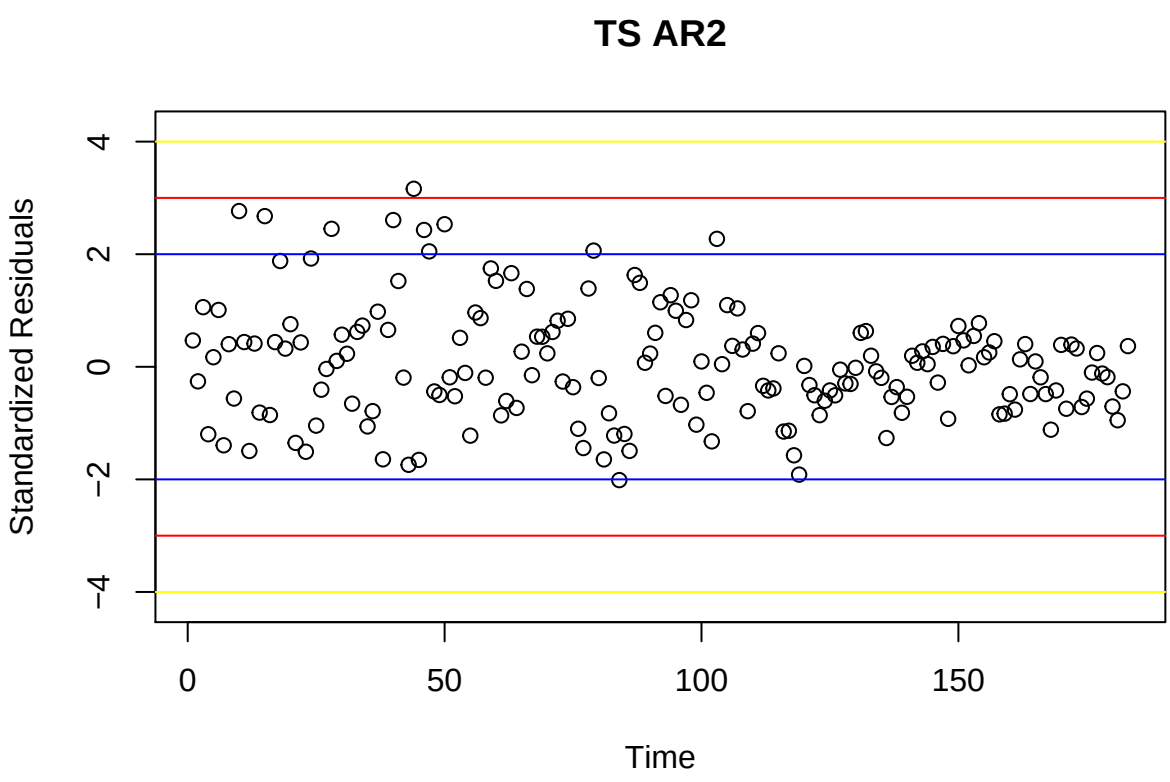



```
##  [1] 1.0000000 0.9656024 0.9293277 0.8937737 0.8570070 0.8224138 0.7869730
##  [8] 0.7510593 0.7160919 0.6805314
```

```
c(phi_11, phi_22, phi_33, phi_44, phi_55, phi_66, phi_77, phi_88, phi_99)
```

```
## [1]  0.969815398 -0.313136044 -0.055188010 -0.122283448  0.006423765
## [6]  0.031687565 -0.065493445 -0.074600400  0.089068429
```

```
#choose AR(2)
TSstdres <- TSmodel.ar2$residuals/(sqrt(TSsigma2.hat))
plot(TSstdres, ylab="Standardized Residuals",
     xlab="Time",
     main="TS AR2", ylim = c(-4.2,4.2))
abline(h=-2, col="blue")
abline(h=2, col="blue")
abline(h=-3, col="red")
abline(h=3, col="red")
abline(h=-4, col="yellow")
abline(h=4, col="yellow")
```




```
Box.test(x = TSmodel.ar2$residuals, type = "Box-Pierce",lag=10)
```

```
##
##  Box-Pierce test
##
## data:  TSmodel.ar2$residuals
## X-squared = 17.517, df = 10, p-value = 0.06369
```

```
#the model is normal but didnt pass JB b/c of the existence of significant outliers or structure breaks
#Note: if an ARMA modle passes all disagnostics except for JB due to outliers, it is still a good mode

#overfit
TSsigma2.hat.overfit <- sum(TSmodel.ar6$residuals^2)/N
TSDev <- 179*log(TSsigma2.hat/TSsigma2.hat.overfit)
TSDev
```

```
## [1] 6.402781
```

```
pval <- 1-pchisq(TSDev, 4)
pval
```

```
## [1] 0.17102
```

```
#do not reject

#Jarque Bera Test
TSskew<-sum(TSstdres^3)/N
TSkur<-sum(TSstdres^4)/N
```

```
TSJBstat<-N*(TSskew^2/6+(TSkur-3)^2/24)
TSt3 <- sqrt(N/6) * TSskew
TSt4<- sqrt(N/24)* (TSkur-3)
TSt3
```

```
## [1] 4.169857
```

```
TSt4
```

```
## [1] 1.43734
```

```
TSt3^2 + TSt4^2
```

```
## [1] 19.45366
```

```
TSJBstat
```

```
## [1] 19.45366
```

```
#jarque.bera.test(Yt)
#JBstat > 6 so reject normality hypothesis

#ARCH(6)
TSres <- TSmodel.ar2$residuals
TSres2 <- TSres^2
N4 <- length(TSres2)
ARCH6<-lm(TSres2[-(1:6)]~TSres2[-c(1:5,N4)]^2+TSres2[-c(1:4,(N4-1):N4)]^2
          +TSres2[-c(1,2,3,(N4-2):N4)]^2+TSres2[-c(1,2,(N4-3):N4)]^2
          +TSres2[-c(1,(N4-4):N4)]^2+TSres2[-((N4-5):N4)]^2)

R<-summary(ARCH6)$r.squared
LMstat<-N4*R
LMstat
```

```
## [1] 22.26204
```

```
1-pchisq(LMstat,6)
```

```
## [1] 0.00108534
```

```
#LMstat is less than 0.872, 99% quantile of chi square 6 distribution, therefore p<0.01. So we reject t
#hypothesis that there is no non-linear


#########DS##########
acf(Yt2)$acf
```

```
## , , 1
##
##               [,1]
##  [1,]  1.000000000
##  [2,]  0.324588402
##  [3,]  0.153930453
##  [4,]  0.173827410
##  [5,]  0.081885800
##  [6,]  0.015592531
##  [7,]  0.072451188
##  [8,]  0.102534042
##  [9,] -0.025286516
```

```
## [10,]  0.164644969
## [11,]  0.149639177
## [12,] -0.074235883
## [13,] -0.002539763
## [14,]  0.113061757
## [15,] -0.089277553
## [16,] -0.033980408
## [17,]  0.124691191
## [18,]  0.019317130
## [19,]  0.056054157
## [20,]  0.148329993
## [21,]  0.068531957
## [22,] -0.090154535
## [23,]  0.086642120
```

```
2/sqrt(length(Yt2))
```

```
## [1] 0.147442
```

```
DSmodel.ar1 <- lm(Yt2[-1] ~ Yt2[-length(Yt2)] - 1)
DSphi_11 <- as.numeric(DSmodel.ar1$coefficients)[1]
DSphi_11
```

```
## [1] 0.3246997
```

```
DSphi_22 <- as.numeric(DSmodel.ar2$coefficients)[2]
DSphi_22
```

```
## [1] 0.05743109
```

```
DSmodel.ar3 <- lm(Yt2[-(1:3)] ~ Yt2[-c(1:2, N2)] + Yt2[-c(1, (N2-1):N2)] + Yt2[-c((N2-2):N2)] - 1)
DSphi_33 <- as.numeric(DSmodel.ar3$coefficients)[3]
DSphi_33
```

```
## [1] 0.1238945
```

```
DSmodel.ar4 <- lm(Yt2[-(1:4)] ~ Yt2[-c(1:3, N2)] + Yt2[-c(1:2, N2-1,N2)]
                  + Yt2[-c(1, (N2-2):N2)] + Yt2[-c((N2-3):N2)] - 1)
DSphi_44 <- as.numeric(DSmodel.ar4$coefficients)[4]
DSphi_44
```

```
## [1] -0.008011481
```

```
DSmodel.ar5 <- lm(Yt2[-(1:5)] ~ Yt2[-c(1:4, N2)] + Yt2[-c(1:3, (N2-1): N2)]
                  + Yt2[-c(1:2, (N2-2):N2)] + + Yt2[-c(1, (N2-3):N2)] + Yt2[-c((N2-4):N2)] - 1)
DSphi_55 <- as.numeric(DSmodel.ar5$coefficients)[5]
DSphi_55
```

```
## [1] -0.03350508
```

```
DSmodel.ar6 <- lm(Yt2[-(1:6)] ~ Yt2[-c(1:5, N2)] + Yt2[-c(1:4, N2-1,N2)]
                  + Yt2[-c(1:3, (N2-2):N2)] + Yt2[-c(1:2, (N2-3):N2)]
                  + Yt2[-c(1, (N2-4):N2)] + Yt2[-c((N2-5):N2)] - 1)
DSphi_66 <- as.numeric(DSmodel.ar6$coefficients)[6]
DSphi_66
```

```
## [1] 0.06472511
```

```
DSmodel.ar7 <- lm(Yt2[-(1:7)] ~ Yt2[-c(1:6, N2)] + Yt2[-c(1:5, N2-1,N2)]
                  + Yt2[-c(1:4, (N2-2):N2)] + Yt2[-c(1:3, (N2-3):N2)]
                  + Yt2[-c(1:2, (N2-4):N2)] + Yt2[-c(1, (N2-5):N2)] + Yt2[-c((N2-6):N2)]- 1)
DSphi_77 <- as.numeric(DSmodel.ar7$coefficients)[7]
DSphi_77
```

```
## [1] 0.07229672
```

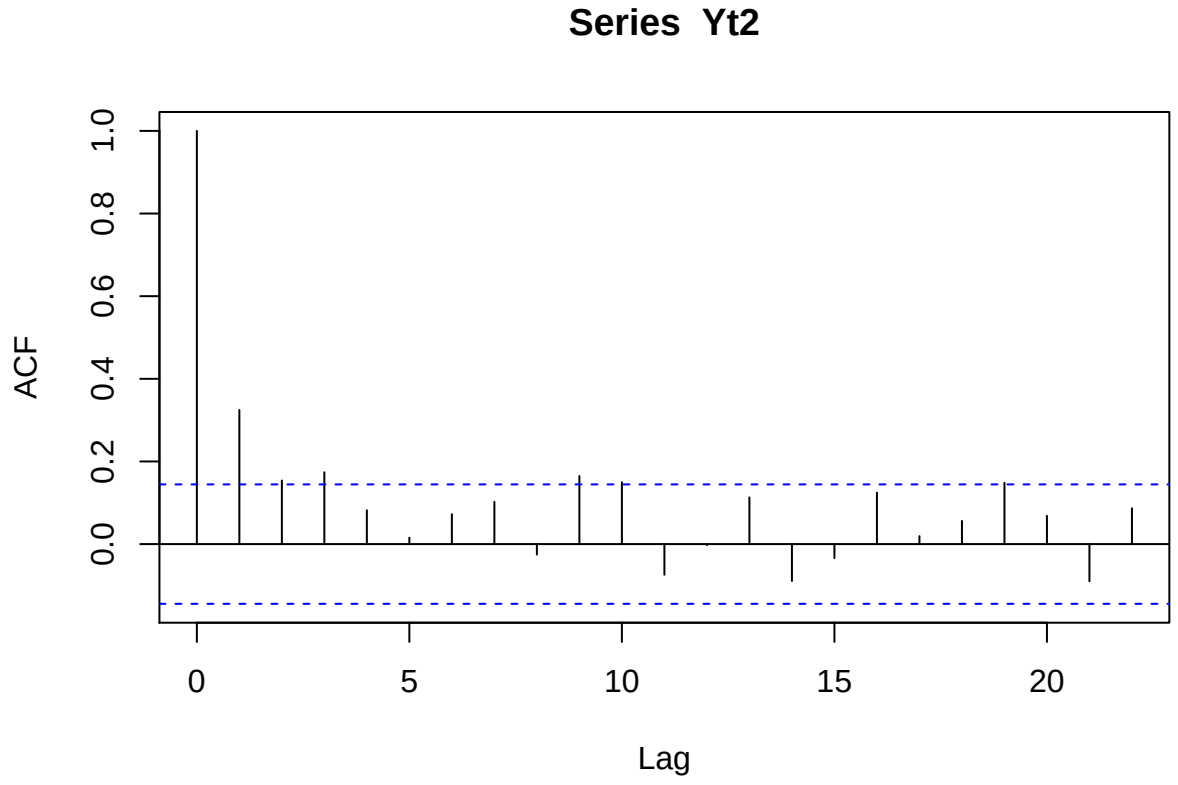
```
DSmodel.ar8 <- lm(Yt2[-(1:8)] ~ Yt2[-c(1:7, N2)] + Yt2[-c(1:6, N2-1,N2)]
                  + Yt2[-c(1:5, (N2-2):N2)] + Yt2[-c(1:4, (N2-3):N2)]
                  + Yt2[-c(1:3, (N2-4):N2)] + Yt2[-c(1:2, (N2-5):N2)] +  Yt2[-c(1, (N2-6):N2)] +  Yt2[-
DSphi_88 <- as.numeric(DSmodel.ar8$coefficients)[8]
DSphi_88
```

```
## [1] -0.0937796
```

```
DSmodel.ar9 <- lm(Yt2[-(1:9)] ~ Yt2[-c(1:8, N2)] + Yt2[-c(1:7, N2-1,N2)]
                  + Yt2[-c(1:6, (N2-2):N2)] + Yt2[-c(1:5, (N2-3):N2)]
                  + Yt2[-c(1:4, (N2-4):N2)] + Yt2[-c(1:3, (N2-5):N2)] +  Yt2[-c(1:2, (N2-6):N2)] +  Yt2
DSphi_99 <- as.numeric(DSmodel.ar9$coefficients)[8]
DSphi_99
```

```
## [1] -0.1634809
```

```
acf(Yt2)$acf[1:10]
```




```
##  [1]  1.00000000  0.32458840  0.15393045  0.17382741  0.08188580
##  [6]  0.01559253  0.07245119  0.10253404 -0.02528652  0.16464497
```

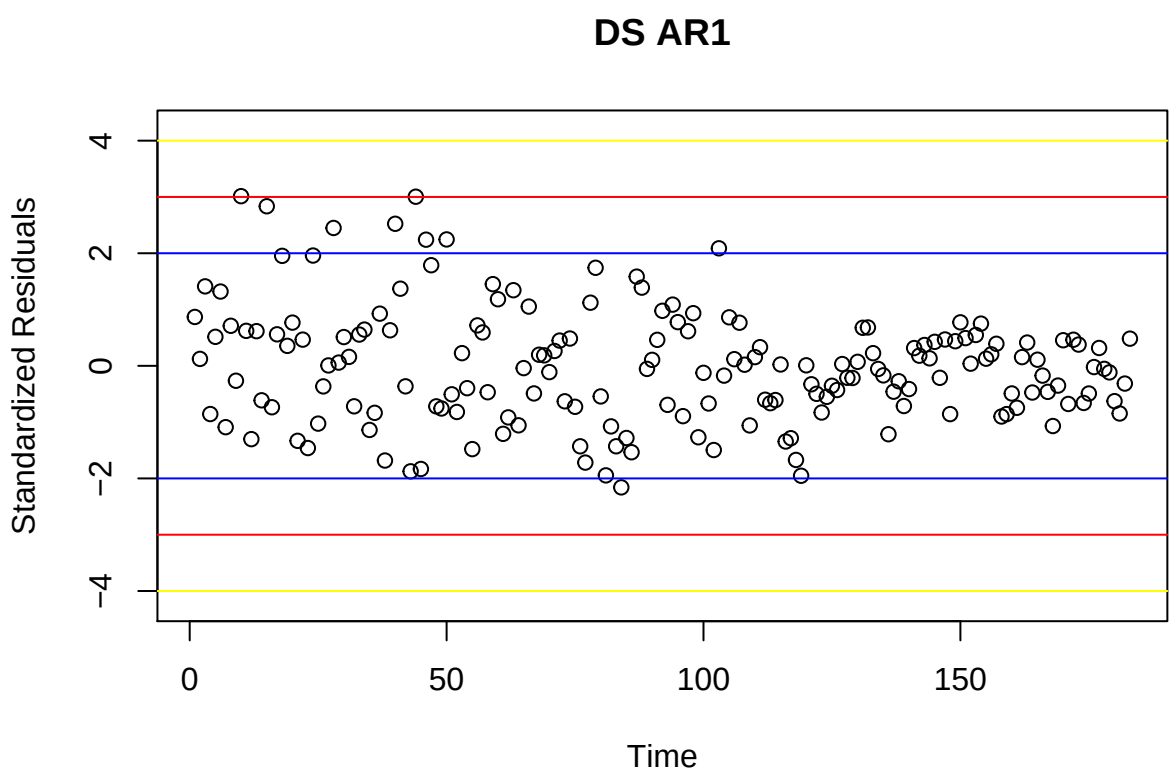
```r
2/sqrt(length(Yt2))
```

```
## [1] 0.147442
```

```r
c(DSphi_11,DSphi_22,DSphi_33,DSphi_44,DSphi_55,DSphi_66,DSphi_77,DSphi_88,DSphi_99)
```

```
## [1]  0.324699722  0.057431092  0.123894498 -0.008011481 -0.033505076
## [6]  0.064725109  0.072296723 -0.093779603 -0.163480895
```

```r
#####AR(1)#######
DSAR1sigma2.hat <- sum(DSmodel.ar1$residuals^2)/N2
DSAR1stdres <- DSmodel.ar1$residuals/(sqrt(DSAR1sigma2.hat))
plot(DSAR1stdres, ylab="Standardized Residuals",
     xlab="Time",
     main="DS AR1", ylim = c(-4.2,4.2))
abline(h=-2, col="blue")
abline(h=2, col="blue")
abline(h=-3, col="red")
abline(h=3, col="red")
abline(h=-4, col="yellow")
abline(h=4, col="yellow")
```




```r
Box.test(x = DSmodel.ar1$residuals, type = "Box-Pierce",lag=10)
```

```
##
##  Box-Pierce test
##
## data:  DSmodel.ar1$residuals
```

```
## X-squared = 17.502, df = 10, p-value = 0.06396
```

```
#the model is normal but didnt pass JB b/c of the existence of significant outliers or structure breaks

#overfit AR(5)
DSAR1sigma2.hat.overfit <- sum(DSmodel.ar5$residuals^2)/N2
DSDev1 <- 180*log(DSAR1sigma2.hat/DSAR1sigma2.hat.overfit)
DSDev1
```

```
## [1] 6.765285
```

```
pval <- 1-pchisq(DSDev1, 4)
pval
```

```
## [1] 0.148824
```

```
#do not reject

#Jarque Bera Test
DSAR1skew<-sum(DSAR1stdres^3)/N2
DSAR1kur<-sum(DSAR1stdres^4)/N2
DSAR1JBstat<-N2*(DSAR1skew^2/6+(DSAR1kur-3)^2/24)
DSt3 <- sqrt(N2/6) * DSAR1skew
DSt4<- sqrt(N2/24)* (DSAR1kur-3)
DSt3
```

```
## [1] 2.760161
```

```
DSt4
```

```
## [1] 1.233261
```

```
DSt3^2 + DSt4^2
```

```
## [1] 9.139424
```

```
DSAR1JBstat
```

```
## [1] 9.139424
```

```
#jarque.bera.test(Yt)
#JBstat > 6 so reject normality hypothesis

#ARCH(6)
# ARCH6<-lm(Yt[-(1:6)]~2~Yt[-c(1:5,N)]~2+Yt[-c(1:4,(N-1):N)]~2+Yt[-c(1,2,3,(N-2):N)]~2+Yt[-c(1,2,(N-3):
#           +Yt[-c(1,(N-4):N)]~2+Yt[-((N-5):N)]~2)

DSAR1res <- DSmodel.ar1$residuals
DSAR1res2<-DSAR1res^2
N5 <- length(DSAR1res2)
ARCH6<-lm(DSAR1res2[-(1:6)]~DSAR1res2[-c(1:5,N5)]+DSAR1res2[-c(1:4,(N5-1):N5)]
          +DSAR1res2[-c(1,2,3,(N5-2):N5)]+DSAR1res2[-c(1,2,(N5-3):N5)]
          +DSAR1res2[-c(1,(N5-4):N5)]+DSAR1res2[-((N5-5):N5)])

R<-summary(ARCH6)$r.squared
R
```

```
## [1] 0.1347175
```

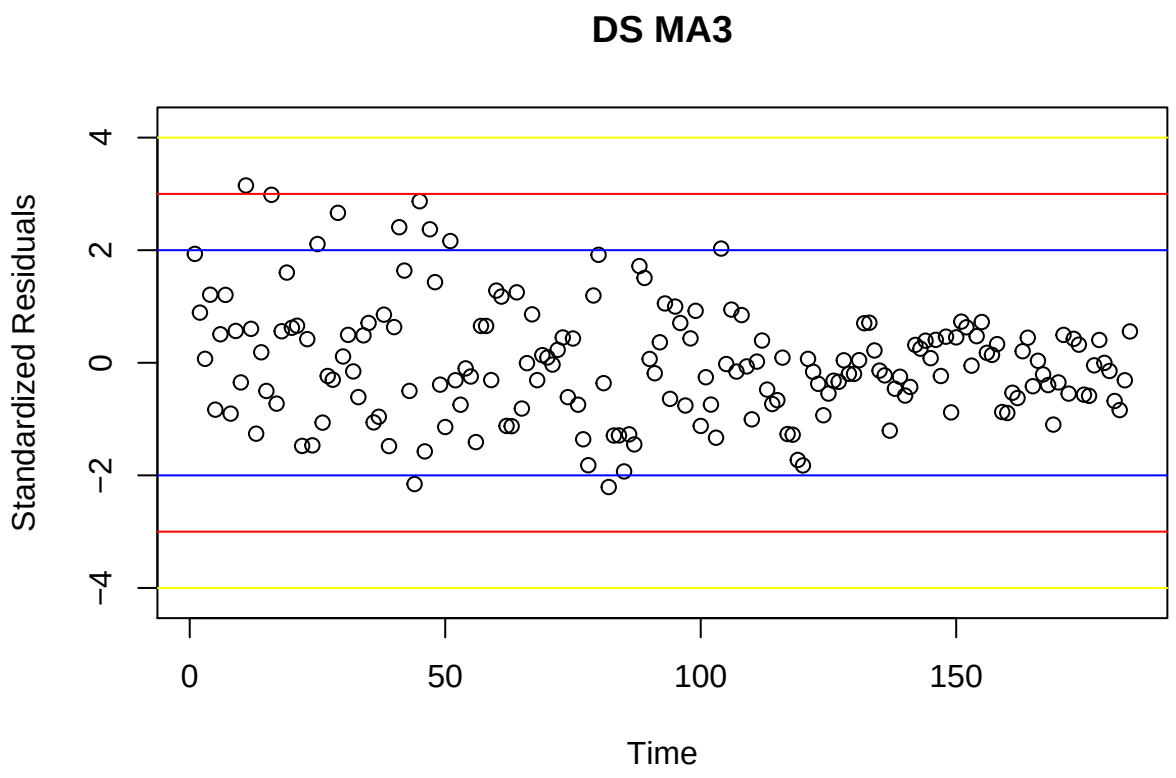
```
LMstat<-N5*R
LMstat
```

```
## [1] 24.65331
```

```
1-pchisq(LMstat,6)
```

```
## [1] 0.0003957796
```

```
#####MA(3)######
DSmodel.ma3 <- arima(Yt2, order = c(0,0,3), include.mean = F)
DSMA3stdres <- DSmodel.ma3$residuals/(sqrt(DSmodel.ma3$sigma2))
plot(DSMA3stdres, ylab="Standardized Residuals",
     xlab="Time",
     main="DS MA3", ylim = c(-4.2,4.2), type ="p")
abline(h=-2, col="blue")
abline(h=2, col="blue")
abline(h=-3, col="red")
abline(h=3, col="red")
abline(h=-4, col="yellow")
abline(h=4, col="yellow")
```




```
Box.test(x = DSmodel.ma3$residuals, type = "Box-Pierce",lag=10)
```

```
##
##  Box-Pierce test
##
## data:  DSmodel.ma3$residuals
## X-squared = 12.061, df = 10, p-value = 0.281
```

```r
#the model is normal but didnt pass JB b/c of the existence of significant outliers or structure breaks

#overfit MA(7)
DSmodel.ma7 <- arima(Yt2, order = c(0,0,7), include.mean = F)
DSDev2 <- 180*log(DSmodel.ma3$sigma2/DSmodel.ma7$sigma2)
DSDev2
```

```
## [1] 4.736769
```

```r
pval <- 1-pchisq(DSDev2, 4)
pval
```

```
## [1] 0.3153882
```

```r
#do not reject

#Jarque Bera Test
DSMA3skew<-sum(DSMA3stdres^3)/N2
DSMA3kur<-sum(DSMA3stdres^4)/N2
DSMA3JBstat<-N2*(DSMA3skew^2/6+(DSMA3kur-3)^2/24)
DSt3 <- sqrt(N2/6) * DSMA3skew
DSt4<- sqrt(N2/24)* (DSMA3kur-3)
DSt3
```

```
## [1] 3.209225
```

```r
DSt4
```

```
## [1] 1.72018
```

```r
DSt3^2 + DSt4^2
```

```
## [1] 13.25815
```

```r
DSMA3JBstat
```

```
## [1] 13.25815
```

```r
#jarque.bera.test(Yt)
#JBstat > 6 so reject normality hypothesis

#ARCH(6)
# ARCH6<-lm(Yt[-(1:6)]~2~Yt[-c(1:5,N)]~2+Yt[-c(1:4,(N-1):N)]~2+Yt[-c(1,2,3,(N-2):N)]~2+Yt[-c(1,2,(N-3):
#           +Yt[-c(1,(N-4):N)]~2+Yt[-((N-5):N)]~2)

DSMA3res <- DSmodel.ma3$residuals
DSMA3res2 <- DSMA3res^2
N6 <- length(DSMA3res2)
ARCH6<-lm(DSMA3res2[-(1:6)]~DSMA3res2[-c(1:5,N6)]+DSMA3res2[-c(1:4,(N6-1):N6)]+DSMA3res2[-c(1,2,3,(N6-2
          +DSMA3res2[-c(1,(N6-4):N6)]+DSMA3res2[-((N6-5):N6)])

R<-summary(ARCH6)$r.squared
R
```

```
## [1] 0.128407
```

```r
LMstat<-N6*R
LMstat
```

```
## [1] 23.62688
```

```r
1-pchisq(LMstat,6)
```

```
## [1] 0.0006115423
```

```r
############8

SPdata<- read_excel("C:/alice/Waterloo Stuff/STUDYMATERIALS/05. STAT443/Project/S&P_data_for_Q8.xls")
Pt<-as.numeric(as.character(SPdata$P))
N3<-length(Pt)
SPlm<-lm(log(Pt[-1])~log(Pt[-N3]))
summary(SPlm)
```

```
##
## Call:
## lm(formula = log(Pt[-1]) ~ log(Pt[-N3]))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.279323 -0.023709  0.002527  0.029122  0.146871
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.005455   0.006955   0.784    0.433
## log(Pt[-N3]) 1.000014   0.001638 610.622   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04363 on 645 degrees of freedom
## Multiple R-squared:  0.9983, Adjusted R-squared:  0.9983
## F-statistic: 3.729e+05 on 1 and 645 DF,  p-value: < 2.2e-16
```

```r
2 * (1-pnorm((1.000014-1)/  0.001638))
```

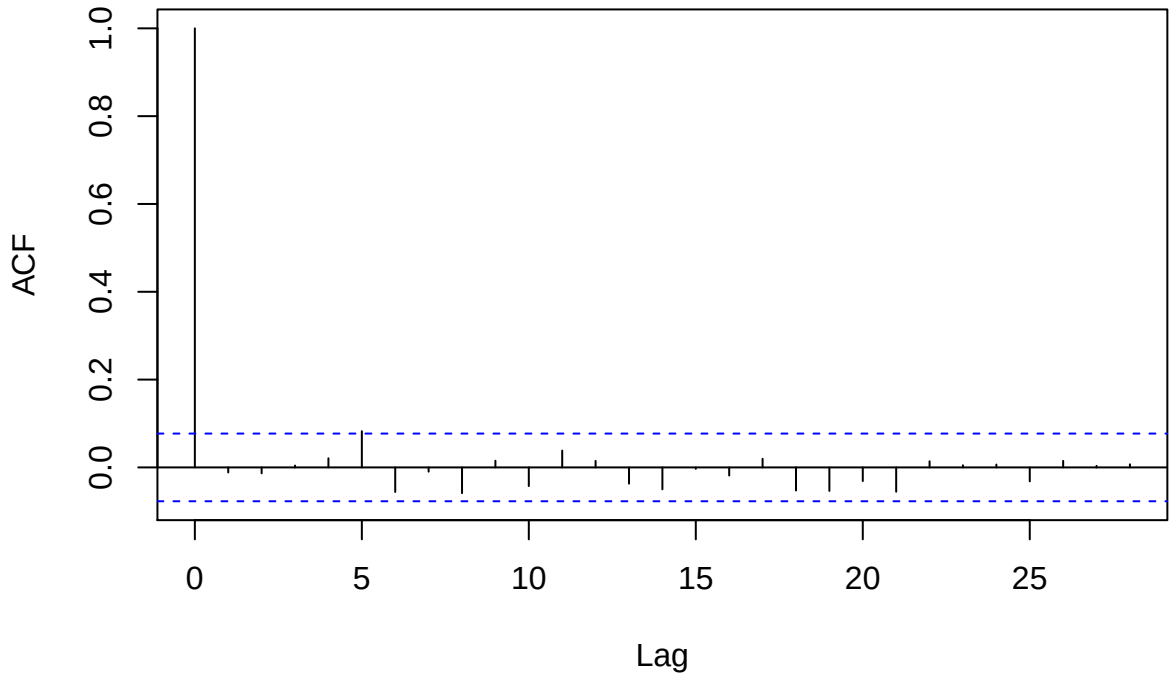
```
## [1] 0.9931806
```

```r
N3
```

```
## [1] 648
```

```r
SPYt<-SPlm$residuals

acf(SPYt,type="correlation")$acf
```




```
## , , 1
##
##               [,1]
##  [1,]  1.000000000
##  [2,] -0.011552297
##  [3,] -0.013532536
##  [4,]  0.004225043
##  [5,]  0.020818464
##  [6,]  0.082459702
##  [7,] -0.055993271
##  [8,] -0.009670618
##  [9,] -0.058582745
## [10,]  0.015241068
## [11,] -0.042526921
## [12,]  0.038412453
## [13,]  0.014855024
## [14,] -0.037045029
## [15,] -0.050003226
## [16,] -0.003299194
## [17,] -0.018603744
## [18,]  0.019610040
## [19,] -0.052647861
## [20,] -0.053633049
## [21,] -0.031077512
## [22,] -0.055482068
## [23,]  0.013967641
```

```
## [24,]  0.004873630
## [25,]  0.006513591
## [26,] -0.031784012
## [27,]  0.014917247
## [28,]  0.003560382
## [29,]  0.007129064
```

```
2/sqrt(N3)
```

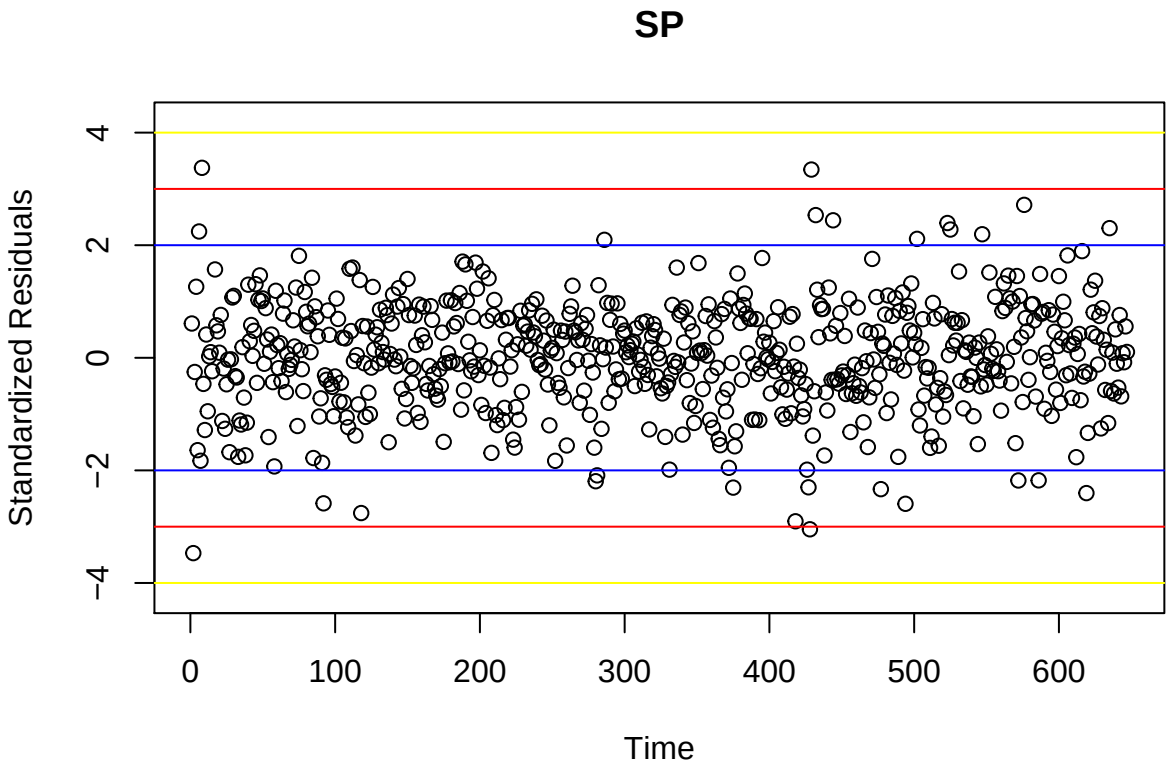
```
## [1] 0.07856742
```

```
#no autocorrelation so independent

Box.test(x = SPlm$residuals, type = "Box-Pierce",lag=25)
```

```
##
##  Box-Pierce test
##
## data:  SPlm$residuals
## X-squared = 21.702, df = 25, p-value = 0.6529
```

```
#std residual
SPstdres<-SPYt/sqrt(sum(resid(SPlm)^2)/N3)
plot(SPstdres,
     ylab="Standardized Residuals",
     xlab="Time",
     main="SP", ylim = c(-4.2,4.2), type ="p")
abline(h=-2, col="blue")
abline(h=2, col="blue")
abline(h=-3, col="red")
abline(h=3, col="red")
abline(h=-4, col="yellow")
abline(h=4, col="yellow")
```




```
SPskew<-sum(SPstdres^3)/N3
SPkur<-sum(SPstdres^4)/N3
SPt3 <- sqrt(N2/6) * SPskew
SPt4<- sqrt(N2/24)* (SPkur-3)
SPt3
```

```
## [1] -4.477661
```

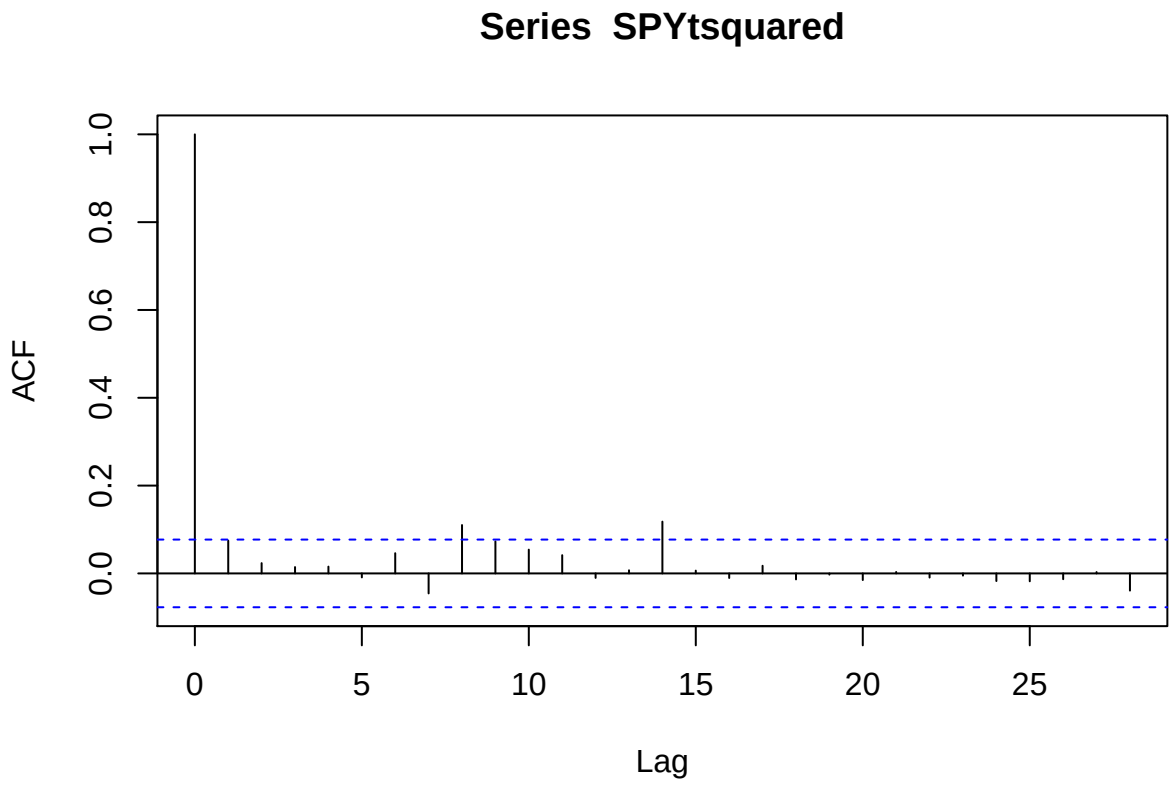
```
SPt4
```

```
## [1] 11.73496
```

```
SPJBstat<-N3*(SPskew^2/6+(SPkur-3)^2/24)
SPJBstat
```

```
## [1] 555.5851
```

```
#autocorrelation for at^2
SPYtsquared<-SPYt^2
acf(SPYtsquared,type="correlation")$acf
```




```
## , , 1
##
##               [,1]
##  [1,]  1.000000000
##  [2,]  0.074510479
##  [3,]  0.023447078
##  [4,]  0.014429417
##  [5,]  0.015468567
##  [6,] -0.009132434
##  [7,]  0.046045932
##  [8,] -0.045541619
##  [9,]  0.110185597
## [10,]  0.072910821
## [11,]  0.054187913
## [12,]  0.041554175
## [13,] -0.010260150
## [14,]  0.007172335
## [15,]  0.117973378
## [16,]  0.006383891
## [17,] -0.010469924
## [18,]  0.017424109
## [19,] -0.013664231
## [20,] -0.002785649
## [21,] -0.015102637
## [22,]  0.003301877
## [23,] -0.009453854
```

```
## [24,] -0.005024318
## [25,] -0.017382318
## [26,] -0.017953626
## [27,] -0.013180921
## [28,]  0.003126356
## [29,] -0.039203809
```

```
garchFit(formula = ~garch(1, 1), data = SPYt)
```

```
##
## Series Initialization:
##  ARMA Model:                arma
##  Formula Mean:              ~ arma(0, 0)
##  GARCH Model:               garch
##  Formula Variance:          ~ garch(1, 1)
##  ARMA Order:                0 0
##  Max ARMA Order:            0
##  GARCH Order:               1 1
##  Max GARCH Order:           1
##  Maximum Order:             1
##  Conditional Dist:          norm
##  h.start:                   2
##  llh.start:                 1
##  Length of Series:          647
##  Recursion Init:            mci
##  Series Scale:              0.04359233
##
## Parameter Initialization:
##  Initial Parameters:          $params
##  Limits of Transformations:   $U, $V
##  Which Parameters are Fixed?  $includes
##  Parameter Matrix:
##                         U            V        params includes
##     mu     -2.730614e-16 2.730614e-16 -7.255523e-14     TRUE
##     omega   1.000000e-06 1.000000e+02  1.000000e-01     TRUE
##     alpha1  1.000000e-08 1.000000e+00  1.000000e-01     TRUE
##     gamma1 -1.000000e+00 1.000000e+00  1.000000e-01    FALSE
##     beta1   1.000000e-08 1.000000e+00  8.000000e-01     TRUE
##     delta   0.000000e+00 2.000000e+00  2.000000e+00    FALSE
##     skew    1.000000e-01 1.000000e+01  1.000000e+00    FALSE
##     shape   1.000000e+00 1.000000e+01  4.000000e+00    FALSE
##  Index List of Parameters to be Optimized:
##     mu  omega alpha1  beta1
##      1      2      3      5
##  Persistence:                  0.9
##
##
## --- START OF TRACE ---
## Selected Algorithm: nlminb
##
## R coded nlminb Solver:
##
##   0:     909.23967: -2.73061e-16 0.100000 0.100000 0.800000
##   1:     909.05627: -2.73061e-16 0.104234 0.101022 0.807923
##   2:     908.49769: -2.73061e-16 0.0994402 0.0936436 0.810002
```

```
##   3:     907.94210: -2.73061e-16 0.0971247 0.0852780 0.825865
##   4:     906.96882: -2.73061e-16 0.0742810 0.0697560 0.849213
##   5:     906.44460: -2.73061e-16 0.0505827 0.0815920 0.873835
##   6:     906.34047: -2.73061e-16 0.0642038 0.0500121 0.885019
##   7:     906.26256: -2.73061e-16 0.0628779 0.0498223 0.884128
##   8:     906.20917: -2.73061e-16 0.0623816 0.0513106 0.884482
##   9:     906.15564: -2.73061e-16 0.0594495 0.0511863 0.885799
##  10:     906.10592: -2.73061e-16 0.0561655 0.0508882 0.891323
##  11:     906.08852: -2.73061e-16 0.0526342 0.0501047 0.893836
##  12:     906.03684: -2.73061e-16 0.0548225 0.0524050 0.890783
##  13:     906.01629: -2.73061e-16 0.0536371 0.0549583 0.887750
##  14:     905.96115: -2.73061e-16 0.0490882 0.0543819 0.894641
##  15:     905.95828: -2.73061e-16 0.0516142 0.0564792 0.891470
##  16:     905.92491: -2.73061e-16 0.0514682 0.0577117 0.889555
##  17:     905.91657: -2.73061e-16 0.0496383 0.0583321 0.890770
##  18:     905.91068: -2.73061e-16 0.0499900 0.0597975 0.889056
##  19:     905.90954: -2.73061e-16 0.0498544 0.0607341 0.888537
##  20:     905.90951: -2.73061e-16 0.0499351 0.0607251 0.888422
##  21:     905.90951: -2.73061e-16 0.0499263 0.0607163 0.888442
##  22:     905.90951: -2.73061e-16 0.0499265 0.0607165 0.888442
##
## Final Estimate of the Negative LLH:
##  LLH:  -1121.06    norm LLH:  -1.732705
##            mu         omega        alpha1         beta1
## -1.190338e-17  9.487481e-05  6.071648e-02  8.884417e-01
##
## R-optimhess Difference Approximated Hessian Matrix:
##                   mu         omega        alpha1         beta1
## mu     -382406.51418      339021.8 -8.917389e+01 -2.762139e+02
## omega   339021.76091 -9070450534.0 -1.294927e+07 -1.562978e+07
## alpha1     -89.17389   -12949271.3 -2.266720e+04 -2.422311e+04
## beta1     -276.21391   -15629777.1 -2.422311e+04 -2.902222e+04
## attr(,"time")
## Time difference of 0.01300001 secs
##
## --- END OF TRACE ---
##
##
## Time to Estimate Parameters:
##  Time difference of 0.0836761 secs
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~garch(1, 1), data = SPYt)
##
## Mean and Variance Equation:
##  data ~ garch(1, 1)
## <environment: 0x00000000123e1558>
##  [data = SPYt]
##
## Conditional Distribution:
```

```
##  norm
##
## Coefficient(s):
##          mu        omega       alpha1        beta1
## -1.1903e-17   9.4875e-05   6.0716e-02   8.8844e-01
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##          Estimate  Std. Error  t value Pr(>|t|)
## mu     -1.190e-17   1.618e-03    0.000  1.00000
## omega   9.487e-05   3.926e-05    2.416  0.01567 *
## alpha1  6.072e-02   2.026e-02    2.996  0.00273 **
## beta1   8.884e-01   2.867e-02   30.988  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  1121.06    normalized:  1.732705
##
## Description:
##  Thu Jul 27 10:28:47 2017 by user: Alice Li
```