lab7\_fall2017.R

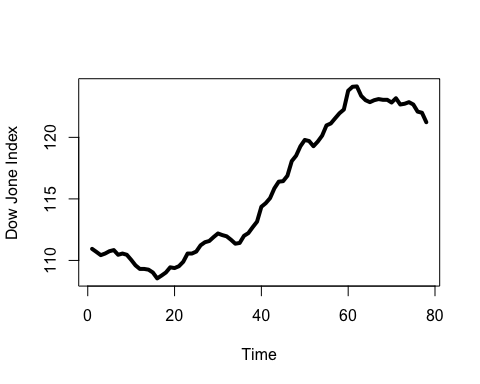
nhanhuynh

Thu Nov 16 13:45:35 2017

# change working directory to folder that contains the data file  
setwd("~/Dropbox/UCSB Classes/274/lab7\_fall2017")  
list.files()

## [1] "barley.txt" "dowj.txt"   
## [3] "lab7\_fall2017.html" "lab7\_fall2017.R"   
## [5] "lab7\_fall2017.spin.R" "lab7\_fall2017.spin.Rmd"  
## [7] "pstat174Lab8.pdf" "pstat174Lab8.Rmd"

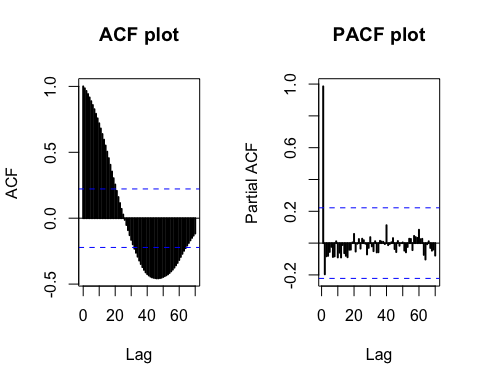
# use read.table() to import txt file into R.   
dowj= read.table("dowj.txt",header=FALSE) # Dow Jones Index data  
# convert this to time series class:  
dowj=ts(dowj)  
  
# I. Description of time series and behavior of ACF and PACF  
  
# display time series:  
plot.ts(dowj,lwd=4,ylab="Dow Jone Index")  
# Describe this plot:  
# (a) There is a significant increasing trend overtime (might or might not linear/ if it is linear trend, taking the first difference is enough). At some specific time points, there are local decreasing trends before peaking again.   
# (b) No seasonal trend   
# (c) Mean is not constant over time --> Indication of non-stationarity.   
library("tseries")



adf.test(dowj) # perform augmented dickey-fuller test for stationarity.

##   
## Augmented Dickey-Fuller Test  
##   
## data: dowj  
## Dickey-Fuller = -1.8053, Lag order = 4, p-value = 0.6552  
## alternative hypothesis: stationary

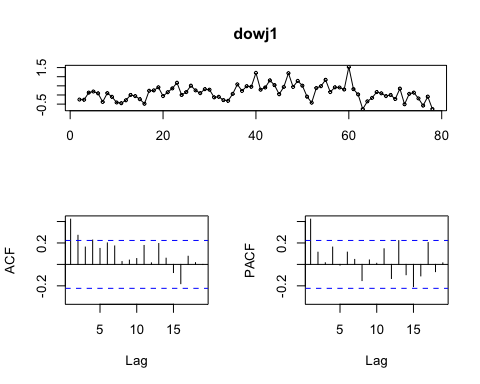
# H0: time series is not stationary  
# Ha: time series is stationary  
# p-value is large --> fail to reject the null. Therefore, time series is not stationary.   
  
# ACF and PACF:  
op=par(mfrow=c(1,2))  
acf(dowj,lwd=2,lag.max=70,main="ACF plot")  
pacf(dowj,lwd=2,lag.max=70,main="PACF plot")



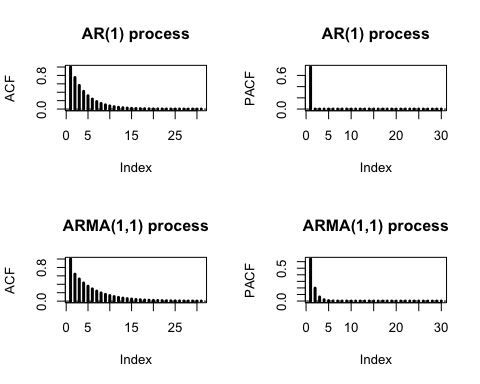
par(op)  
# Several things we can tell using these two graphs:  
# (a). ACF: we observe sinusoidal behavior (wave form) --> typical behavior of AR or ARMA models.  
# (b). PACF: the first lag is significant. After the first lag, the remaining lags are insignificant.  
  
## we want to build model on stationary time series to make prediction more reliable. In this case, let's detrend the data by taking difference at lag 1:  
dowj1=diff(dowj,lag=1)  
library("forecast")

## Warning in as.POSIXlt.POSIXct(Sys.time()): unknown timezone 'default/  
## America/Los\_Angeles'

tsdisplay(dowj1)



## observation? after taking the difference at lag 1, we remove the linear trend. We improve the original time series quite alot. Looking at ACF and PACF:  
# a. ACF: a significant correlation at lag 1. After that, it remains insignificant.   
# b. PACF: a significant partial correlation at lag 1. After that, it remains insignificant.   
## Some models we can try according to these behaviors:  
# 1. AR(1) (using PACF plot) on differenced series. If on the original series, ARIMA(1,1,0)  
# 2. MA(1) (using ACF plot) on differenced series. If on the original series, ARIMA(0,1,1)  
# 3. ARMA(1,1) (it often has similar behavior in ACF as AR(1) model - See small experiment below) on different series. If on original series, ARIMA(1,1,1)  
  
######################## SMALL EXPERIMENT ##########################  
# examine theoretical ACF and PACF for AR(1) and ARMA(1,1)  
# AR(1) with phi=0.75  
acfAR1=ARMAacf(ar=0.75,lag.max=30)  
pacfAR1=ARMAacf(ar=0.75,pacf=TRUE,lag.max=30)  
# ARMA(1,1) with phi=0.89, and theta=0.54  
acfARMA11=ARMAacf(ar=0.821,ma=-.32,lag.max=30)  
pacfARMA11=ARMAacf(ar=0.821,ma=-.32,pacf=TRUE,lag.max=30)  
  
# display them altogether:  
par(mfrow=c(2,2))  
plot(acfAR1,type="h",lwd=3,ylab="ACF",main="AR(1) process")  
abline(h=0,lty=2)  
plot(pacfAR1,type="h",lwd=3,ylab="PACF",main="AR(1) process")  
abline(h=0,lty=2)  
plot(acfARMA11,type="h",lwd=3,ylab="ACF",main="ARMA(1,1) process")  
abline(h=0,lty=2)  
plot(pacfARMA11,type="h",lwd=3,ylab="PACF",main="ARMA(1,1) process")  
abline(h=0,lty=2)



par(op)  
####################################################################  
  
# II. Fitting model:  
  
# 1. Fitting AR model using different methods:  
# (a) Yule-Walker equations:  
arYW=ar(dowj1,method="yule-walker") # AR(1) using differenced series  
arYW

##   
## Call:  
## ar(x = dowj1, method = "yule-walker")  
##   
## Coefficients:  
## 1   
## 0.4219   
##   
## Order selected 1 sigma^2 estimated as 0.1518

# (b) Maximum likelihood:  
arML=arima(dowj,order=c(1,1,0),method="ML") # ARIMA(1,1,0) using orginal data  
arML

##   
## Call:  
## arima(x = dowj, order = c(1, 1, 0), method = "ML")  
##   
## Coefficients:  
## ar1  
## 0.4992  
## s.e. 0.1001  
##   
## sigma^2 estimated as 0.1493: log likelihood = -36.19, aic = 76.38

# both methods yield AR(1).  
  
# 2. Fitting MA model using different methods:  
# (a) Innovative algorithm method (review lab6\_fall2017.R file)  
# (b) Maximum likelihood:  
maML=arima(dowj,order=c(0,1,2),method="ML")  
maML

##   
## Call:  
## arima(x = dowj, order = c(0, 1, 2), method = "ML")  
##   
## Coefficients:  
## ma1 ma2  
## 0.4485 0.2343  
## s.e. 0.1185 0.0933  
##   
## sigma^2 estimated as 0.1518: log likelihood = -36.82, aic = 79.63

# 3. Fitting ARIMA model:  
# (a) Write a for loop to pick model with smaller AICc value  
library(qpcR,quietly = TRUE)  
for (i in 1:3){  
 for (j in 1:3){  
 current=arima(dowj,order=c(i,1,j),method="ML")  
 print(c(i,j,AICc(current)))  
 }  
}

## [1] 1.00000 1.00000 75.53827  
## [1] 1.00000 2.00000 76.34577  
## [1] 1.00000 3.00000 78.12429  
## [1] 2.00000 1.00000 79.13451  
## [1] 2.00000 2.00000 77.09623  
## [1] 2.00000 3.00000 78.98859  
## [1] 3.00000 1.00000 78.36411  
## [1] 3.00000 2.00000 78.62128  
## [1] 3.00000 3.00000 80.02977

# a good candiate is ARIMA(1,1,1) with smallest AICc of 75.53827  
  
# (b) Use auto.arima():  
autofit=auto.arima(dowj)  
autofit

## Series: dowj   
## ARIMA(1,1,1)   
##   
## Coefficients:  
## ar1 ma1  
## 0.8510 -0.5263  
## s.e. 0.1383 0.2548  
##   
## sigma^2 estimated as 0.1474: log likelihood=-34.69  
## AIC=75.38 AICc=75.71 BIC=82.41

## now it is a good time to turn our head back and compare all models we have been building so far:  
# ARIMA(1,1,0) (using MLE method) has AICc:  
AICc(arML)

## [1] 76.43379

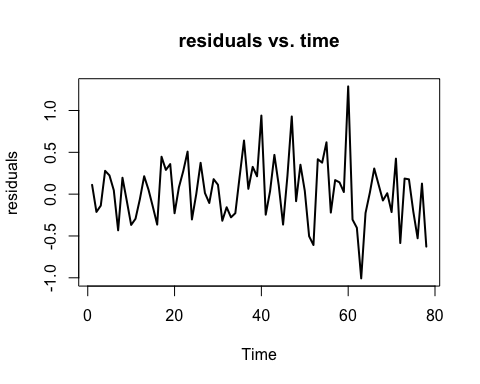
# ARIMA(0,1,2) (using MLE method) has AICc:  
AICc(maML)

## [1] 79.79421

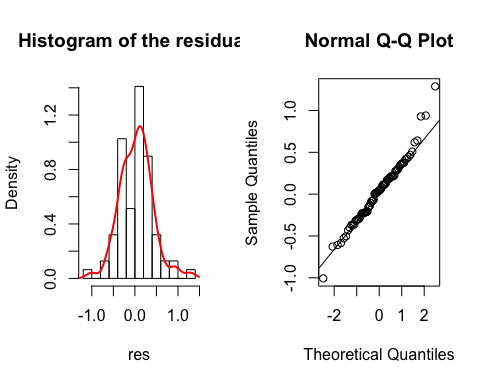
# ARIMA(1,1,1) has AICc:  
AICc(autofit)

## [1] 75.53826

# Best candidate is ARIMA(1,1,1)  
  
#### III. Model diagnostic ######  
# Check for 2 important things: normality and independence using the residuals:  
res=residuals(autofit)  
plot(res,lwd=2,type="l",ylab="residuals",main="residuals vs. time")



# (a) Check normality:  
op=par(mfrow=c(1,2))  
hist(res,breaks=10,probability = TRUE,main="Histogram of the residuals")  
lines(density(res),col="red",lwd=2)  
qqnorm(res)  
qqline(res)



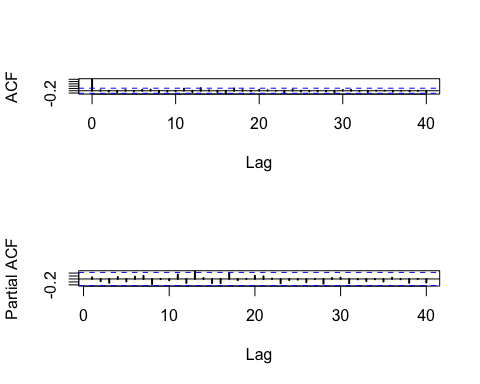
par(op)  
# Some potential outliers but most of the points are lying near qqline  
# Using hypothesis test such as Shapiro-Wilk test for normality assumption. Keep in mind that these hypothesis tests are conservative; in practice --> qqplot/histogram might be enough!  
shapiro.test(res)

##   
## Shapiro-Wilk normality test  
##   
## data: res  
## W = 0.97899, p-value = 0.2248

# H0: the distribution follows normal distribution  
# Ha: not H0  
# using p-value from ouput --> normality assumption is ok.   
  
# (b) Test for independence  
# If model does a good job at fitting the data, model should explain most of systematic part, what is left should be just white noise.  
Box.test(res)

##   
## Box-Pierce test  
##   
## data: res  
## X-squared = 0.19685, df = 1, p-value = 0.6573

# p-value suggests that residuals are uncorrelated. If we want to visualize the result, we can construct ACF and PACF:  
op=par(mfrow=c(2,1))  
acf(res,lag.max = 40,lwd=2,main=" ")  
pacf(res,lag.max = 40,lwd=2,main=" ")



par(op)  
  
##### IV. Forecast #####  
### Forecast the next 5 observations using the model:  
pred = predict(autofit,n.ahead=5)  
  
# 95% lower CI of the prediction:  
lower= pred$pred-1.96\*pred$se  
lower

## Time Series:  
## Start = 79   
## End = 83   
## Frequency = 1   
## [1] 120.1524 119.3793 118.6575 117.9743 117.3261

# 95% upper CI of the prediction:  
upper=pred$pred+1.96\*pred$se  
upper

## Time Series:  
## Start = 79   
## End = 83   
## Frequency = 1   
## [1] 121.6572 121.8768 122.1276 122.4100 122.7171

par(mfrow=c(1,1))  
ts.plot(dowj,lwd=4,xlim=c(0,85))  
polygon(c(79:83, rev(79:83)),c(lower,rev(upper)),  
 col = "thistle", border = NA) # shaded the CI region first  
points(79:83,pred$pred,pch=19) # add predicted points onto the plot  
# the next 10 predicted steps show decreasing pattern in the future  
lines(79:83,lower,col="red",lwd=2) # add 95% confidence interval for each predicted points  
lines(79:83,upper,col="red",lwd=2)

