Week 2: Dissecting a Time Series

# Time Series using ZOO

library(tseries)

## Registered S3 method overwritten by 'xts':  
## method from  
## as.zoo.xts zoo

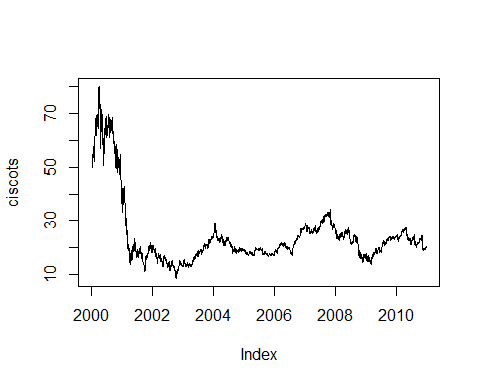
## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

library(zoo)

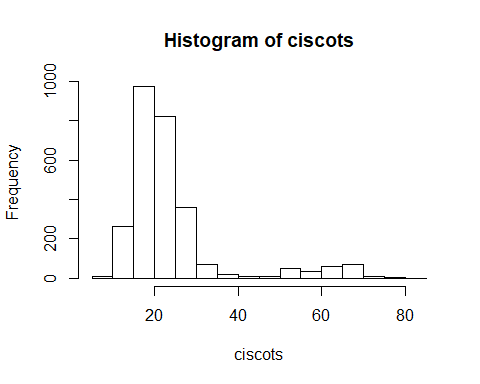
##   
## Attaching package: 'zoo'

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

#import dataset into a dataframe  
cisco <- read.table('C:/Users/Xuan Pham/Dropbox/Fall\_2019/BIA6315/code/Week 2/Data/cisco\_00-10.csv', header=T, sep=',')  
  
# create time series for cisco prices  
ciscots <- zoo(cisco$Price, as.Date(as.character(cisco$Date), format = "%m/%d/%y"))  
  
  
plot(ciscots)



hist(ciscots)



mean(ciscots)

## [1] 24.4816

#Creating new variables

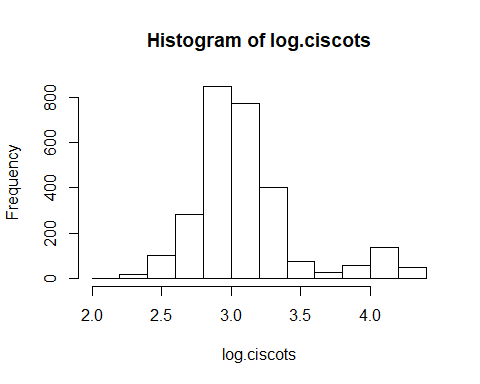
# Four moments of a normal distribution:

-First moment: Mean  
-Second moment: Variance  
-Third moment: Skewness  
-Fourth moment: Kurtosis

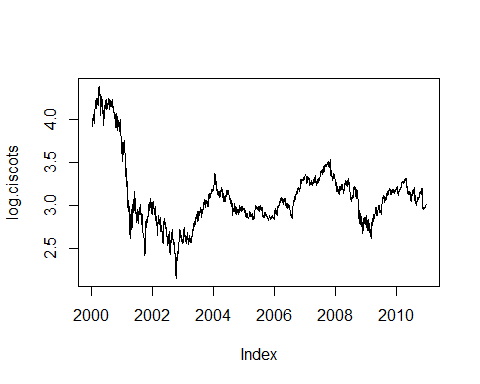
Is Cisco’s daily rate of returns come from a normal distribution?

Looking at the histogram & time series plot, what do you see?

#Natural log transformation makes the distribution less skewed.  
#DEFINE LOG RETURNS  
#rts is a time series object since it is created from a TS object  
log.ciscots <- log(ciscots)  
hist(log.ciscots) #any difference?



plot(log.ciscots) #any difference?



mean(log.ciscots)

## [1] 3.110066

Another problem: If the processes that give rise to the Cisco time series change over time, we cannot “model” these processes? Hence, can we do something about the trend?

What about taking the day-to-day difference in log price?

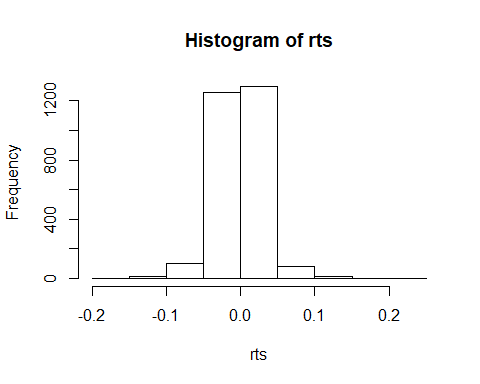
#first the hard way of doing it...  
lagged.ciscots <- lag(log.ciscots, k=-1)  
price.diff <- (log.ciscots - lagged.ciscots)  
head(price.diff)

## 2000-01-04 2000-01-05 2000-01-06 2000-01-07 2000-01-10 2000-01-11   
## -0.05771382 -0.00294551 -0.01685712 0.05713619 0.03635415 -0.03051554

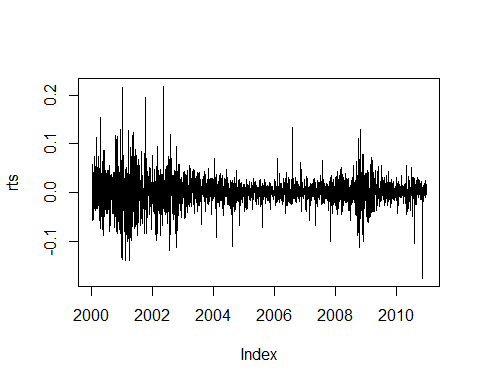
#now the easier way...  
rts = diff(log.ciscots, lag=1) #default is lag = 1  
head(rts)

## 2000-01-04 2000-01-05 2000-01-06 2000-01-07 2000-01-10 2000-01-11   
## -0.05771382 -0.00294551 -0.01685712 0.05713619 0.03635415 -0.03051554

hist(rts) #any difference?



plot(rts) #any difference?



mean(rts)

## [1] -0.0003551601

This is called “de-trending” our time series. The mean is now close to 0.

rt <- coredata(rts) #keeping just the price data. No time index.  
library(fBasics)

## Loading required package: timeDate

## Loading required package: timeSeries

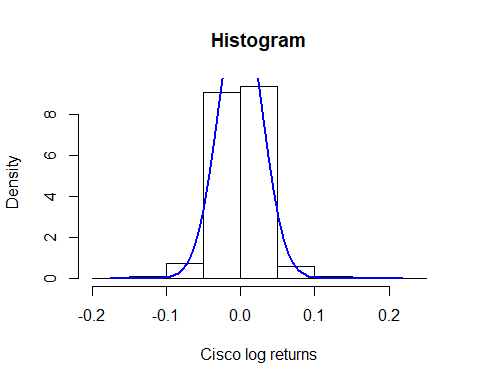
##   
## Attaching package: 'timeSeries'

## The following object is masked from 'package:zoo':  
##   
## time<-

# COMPUTE SUMMARY STATISTICS  
basicStats(rt)

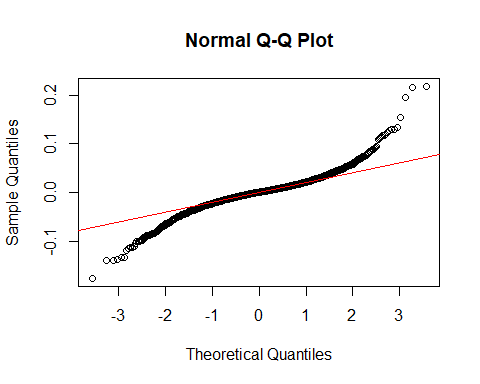
## rt  
## nobs 2766.000000  
## NAs 0.000000  
## Minimum -0.176865  
## Maximum 0.218239  
## 1. Quartile -0.013890  
## 3. Quartile 0.013411  
## Mean -0.000355  
## Median 0.000449  
## Sum -0.982373  
## SE Mean 0.000560  
## LCL Mean -0.001453  
## UCL Mean 0.000742  
## Variance 0.000867  
## Stdev 0.029437  
## Skewness 0.187810  
## Kurtosis 6.053895

# CREATE HISTOGRAM   
# OPTIONAL creates 2 by 2 display for 4 plots   
# par(mfcol=c(2,2))   
hist(rt, xlab="Cisco log returns", prob=TRUE, main="Histogram")   
# add approximating normal density curve   
xfit<-seq(min(rt),max(rt),length=60)   
yfit<-dnorm(xfit,mean=mean(rt),sd=sd(rt))   
lines(xfit, yfit, col="blue", lwd=2)

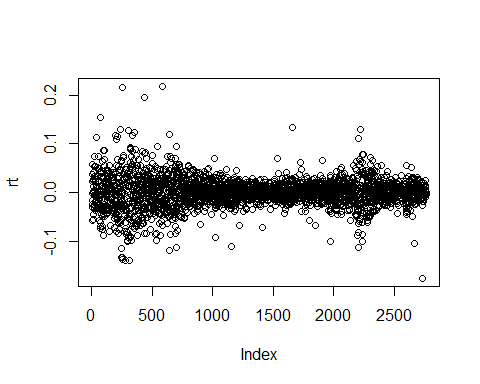


Is there skewness? What about kurtosis?

# CREATE NORMAL PROBABILITY PLOT   
qqnorm(rt)   
qqline(rt, col = 2)



plot(rt)



library(moments)

##   
## Attaching package: 'moments'

## The following objects are masked from 'package:timeDate':  
##   
## kurtosis, skewness

skewness(rt)

## [1] 0.1879119

kurtosis(rt)

## [1] 9.060445

# NORMALITY TESTS   
# Perform Jarque-Bera normality test.   
#H0: Data is normally distributed  
#H1: Data is not normally distributed  
normalTest(rt,method=c("jb"))

##   
## Title:  
## Jarque - Bera Normalality Test  
##   
## Test Results:  
## STATISTIC:  
## X-squared: 4249.2945  
## P VALUE:  
## Asymptotic p Value: < 2.2e-16   
##   
## Description:  
## Fri Sep 27 09:55:25 2019 by user: Xuan Pham

#If you want to see the critical values for chi-square distribution, check here:  
#https://www.itl.nist.gov/div898/handbook/eda/section3/eda3674.htm

So the Cisco price data set is not normally distributed.

Now let’s turn our attention to the “independent, identically distributed” assumption.

Independent: Each sample observation is uncorrelated with another sample observation. Identically distributed: The sample observations are drawn from the same probability distribution.

It’s obvious that time series violates this assumption.

How do we examine this?

In traditional statistics, we have covariance & correlation to examine the linear relationship between two variables, X & Y.

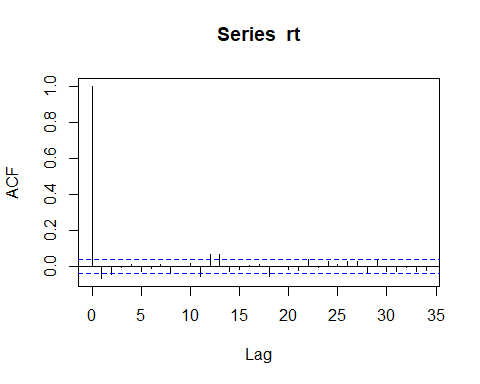
Covariance: Direction of the relationship Correlation: Direction & strength of the relationship

In time series analysis, we change X & Y into Y(t) and Y(t-k) where k = some number of lagged periods. Hence, we have autocovariance and autocorrelation.

# COMPUTE ACF AND PLOT CORRELOGRAM   
#prints acf values to console   
acf(rt, plot=F)

##   
## Autocorrelations of series 'rt', by lag  
##   
## 0 1 2 3 4 5 6 7 8 9   
## 1.000 -0.065 -0.046 -0.003 0.011 -0.026 -0.009 0.010 -0.038 -0.001   
## 10 11 12 13 14 15 16 17 18 19   
## 0.019 -0.056 0.067 0.065 -0.026 -0.015 0.005 0.010 -0.055 0.001   
## 20 21 22 23 24 25 26 27 28 29   
## -0.014 -0.023 0.037 -0.007 0.028 0.014 0.030 0.030 -0.038 0.040   
## 30 31 32 33 34   
## -0.026 -0.025 -0.003 -0.025 -0.020

#plot acf values on graph (correlogram)   
acf(rt, plot=T)

 Two things to look for in an ACF plot: 1. Do I have significant lags? 2. Do I have rapid/gradual decay?

Is there a hypothesis test we can run to test for autocorrelation?

# COMPUTE LJUNG-BOX TEST FOR WHITE NOISE (NO AUTOCORRELATION)  
#H0: p(1) = p(2) = p(k) = 0  
#H1: p(k) is not equal to 0  
  
# to Lag 3  
Box.test(rt,lag=3,type='Ljung')

##   
## Box-Ljung test  
##   
## data: rt  
## X-squared = 17.638, df = 3, p-value = 0.0005222

# to Lag 13   
Box.test(rt,lag=13,type='Ljung')

##   
## Box-Ljung test  
##   
## data: rt  
## X-squared = 58.224, df = 13, p-value = 1.089e-07