Project 1 – HO2 Analysis

Live Sessions: weeks 1 and 2

## Part 1

In this set we will build a data set using filters and if and diff statements. We will then answer some questions using plots and a pivot table report. We will then write a function to house our approach in case we would like to run the same analysis on other data sets.

### Problem

Supply chain managers at our company continue to note we have a significant exposure to heating oil prices (Heating Oil No. 2, or HO2), specifically New York Harbor. The exposure hits the variable cost of producing several products. When HO2 is volatile, so is earnings. Our company has missed earnings forecasts for five straight quarters. To get a handle on HO2 we download this data set and review some basic aspects of the prices.

# Read in data  
# package EIAdata  
#  
HO2 = read.csv("../data/nyhh02.csv", header = T, stringsAsFactors = F)  
# stringsAsFactors sets dates as character type  
head(HO2)

## DATE DHOILNYH  
## 1 6/2/1986 0.402  
## 2 6/3/1986 0.393  
## 3 6/4/1986 0.378  
## 4 6/5/1986 0.390  
## 5 6/6/1986 0.385  
## 6 6/9/1986 0.373

HO2 = na.omit(HO2) ## to clean up any missing data  
# use na.approx() as well  
str(HO2) # review the structure of the data so far

## 'data.frame': 7697 obs. of 2 variables:  
## $ DATE : chr "6/2/1986" "6/3/1986" "6/4/1986" "6/5/1986" ...  
## $ DHOILNYH: num 0.402 0.393 0.378 0.39 0.385 0.373 0.365 0.389 0.394 0.398 ...

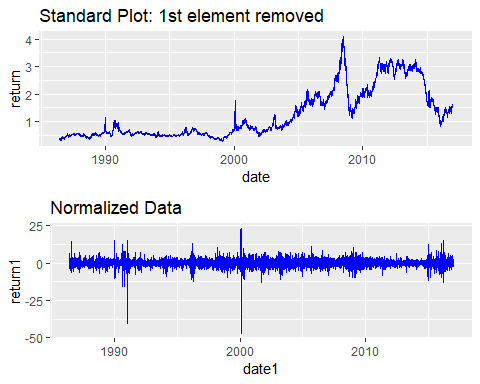
### Questions

1. What is the nature of HO2 returns? We want to reflect the ups and downs of price movements, something of prime interest to management. First, we calculate percentage changes as log returns. Our interest is in the ups and downs. To look at that we use if and else statements to define a new column called direction. We will build a data frame to house this analysis.

# Construct expanded data frame  
#  
# size + direction, indicator of volatility  
# date, length of DATE is length of return +1: omit 1st observation  
# price, length of DHOILNYH is length of return +1: omit first observation  
#  
return1 = as.numeric(diff(log(HO2$DHOILNYH))) \* 100 # Euler  
size1 = abs(return1)  
direction1 = ifelse(return1 > 0, "up", ifelse(return1 < 0, "down", "same"))  
date1 = as.Date(HO2$DATE[-1], "%m/%d/%Y")  
price1 = as.numeric(HO2$DHOILNYH[-1])  
price1 = as.numeric(diff(log(HO2$DHOILNYH)))  
HO2.df1 = na.omit(data.frame(  
 date = date1,  
 price = price1,  
 return = return1,  
 size = size1,  
 direction = direction1  
))  
  
return = HO2$DHOILNYH  
size = abs(return)  
direction = ifelse(return > 0, "up", ifelse(return < 0, "down", "same"))  
date = as.Date(HO2$DATE, "%m/%d/%Y")  
price = as.numeric(HO2$DHOILNYH)  
HO2.df = na.omit(data.frame(  
 date = date,  
 price = price,  
 return = return,  
 size = size,  
 direction = direction  
))

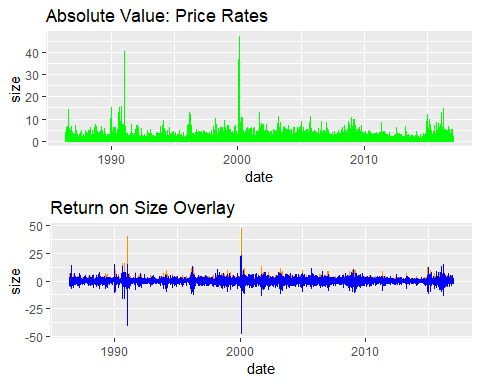
We can plot with the ggplot2 package. In the ggplot statements we use aes, “aesthetics”, to pick x (horizontal) and y (vertical) axes. Use group =1 to ensure that all data is plotted. The added (+) geom\_line is the geometrical method that builds the line plot.

library(ggplot2)  
library(gridExtra)  
p1 = ggplot(HO2.df, aes(x = date, y = return, group = 1)) +  
 geom\_line(colour = "blue") +  
 ggtitle('Standard Plot: 1st element removed')  
  
p2 = ggplot(HO2.df1, aes(x = date1, y = return1, group = 1)) +  
 geom\_line(colour = "blue") +  
 ggtitle('Normalized Data')  
  
grid.arrange(p1, p2, nrow=2)



Let’s try a bar graph of the absolute value of price rates. We use geom\_bar to build this picture.

library(ggplot2)  
p3 = ggplot(HO2.df1, aes(x = date, y = size, group = 1)) +  
 geom\_bar(stat = "identity", colour = "green") +  
 ggtitle('Absolute Value: Price Rates')  
  
p4 = ggplot(HO2.df1, aes(date, size)) +  
 geom\_bar(stat = "identity", colour = "darkorange") +  
 geom\_line(data = HO2.df1, aes(date, return), colour = "blue") +  
 ggtitle('Return on Size Overlay')  
  
grid.arrange(p3, p4, nrow=2)



1. Let’s dig deeper and compute mean, standard deviation, etc. Load the data\_moments() function. Run the function using the HO2.df1$return subset of the data and write a knitr::kable() report.

data\_moments = function(data) {  
 library(moments)  
 mean.r = mean(data)  
 sd.r = sd(data)  
 median.r = median(data)  
 skewness.r = skewness(data)  
 kurtosis.r = kurtosis(data)  
 result = data.frame(  
 mean = mean.r,  
 std\_dev = sd.r,  
 median = median.r,  
 skewness = skewness.r,  
 kurtosis = kurtosis.r  
 )  
  
 return(result)  
}  
  
# Run data\_moments()  
answer = data\_moments(HO2.df1$return)  
  
# Build pretty table  
answer = round(answer, 4)  
knitr::kable(answer)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| mean | std\_dev | median | skewness | kurtosis |
| 0.0179 | 2.5236 | 0 | -1.4353 | 38.2595 |

1. Let’s pivot size and return on direction. What is the average and range of returns by direction? How often might we view positive or negative movements in HO2?

# Counting  
table(HO2.df1$return < 0) # one way

##   
## FALSE TRUE   
## 4039 3657

table(HO2.df1$return > 0)

##   
## FALSE TRUE   
## 3936 3760

table(HO2.df1$direction) # this counts 0 returns as negative

##   
## down same up   
## 3657 279 3760

table(HO2.df1$return == 0)

##   
## FALSE TRUE   
## 7417 279

# Pivoting  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following object is masked from 'package:gridExtra':  
##   
## combine

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

## 1: filter to those houses with fairly high prices  
# pivot.table =filter(HO2.df, size > 0.5\*max(size))  
  
## 2: set up data frame for by-group processing  
pivot.table =group\_by(HO2.df1, direction)  
  
## 3: calculate the summary metrics  
options(dplyr.width = Inf) ## to display all columns  
HO2.count = length(HO2.df1$return)  
pivot.table = summarise(  
 pivot.table,  
 return.avg = round(mean(return), 4),  
 return.sd = round(sd(return), 4),  
 quantile.5 = round(quantile(return, 0.05), 4),  
 quantile.95 = round(quantile(return, 0.95), 4),  
 percent = round((length(return)/HO2.count)\*100, 2)  
)  
# Build visual  
knitr::kable(pivot.table, digits = 2)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| direction | return.avg | return.sd | quantile.5 | quantile.95 | percent |
| down | -1.77 | 1.99 | -4.78 | -0.19 | 47.52 |
| same | 0.00 | 0.00 | 0.00 | 0.00 | 3.63 |
| up | 1.76 | 1.75 | 0.18 | 4.82 | 48.86 |

# Here is how we can produce a LaTeX formatted and rendered table  
library(xtable)  
HO2.caption = "Heating Oil No. 2: 1986-2016"  
print(xtable(t(pivot.table), digits = 2, caption = HO2.caption, align=rep("r", 4), table.placement="V"))

## % latex table generated in R 3.5.2 by xtable 1.8-3 package  
## % Sun Jan 27 19:01:16 2019  
## \begin{table}[ht]  
## \centering  
## \begin{tabular}{rrrr}  
## \hline  
## & 1 & 2 & 3 \\   
## \hline  
## direction & down & same & up \\   
## return.avg & -1.7718 & 0.0000 & 1.7598 \\   
## return.sd & 1.9862 & 0.0000 & 1.7460 \\   
## quantile.5 & -4.7761 & 0.0000 & 0.1817 \\   
## quantile.95 & -0.1894 & 0.0000 & 4.8203 \\   
## percent & 47.52 & 3.63 & 48.86 \\   
## \hline  
## \end{tabular}  
## \caption{Heating Oil No. 2: 1986-2016}   
## \end{table}

print(xtable(answer), digits = 2)

## % latex table generated in R 3.5.2 by xtable 1.8-3 package  
## % Sun Jan 27 19:01:16 2019  
## \begin{table}[ht]  
## \centering  
## \begin{tabular}{rrrrrr}  
## \hline  
## & mean & std\\_dev & median & skewness & kurtosis \\   
## \hline  
## 1 & 0.02 & 2.52 & 0.00 & -1.44 & 38.26 \\   
## \hline  
## \end{tabular}  
## \end{table}

## Part 2

We will use the data from Part 1 to investigate the distribution of returns we generated. This will entail fitting the data to some parametric distributions as well as

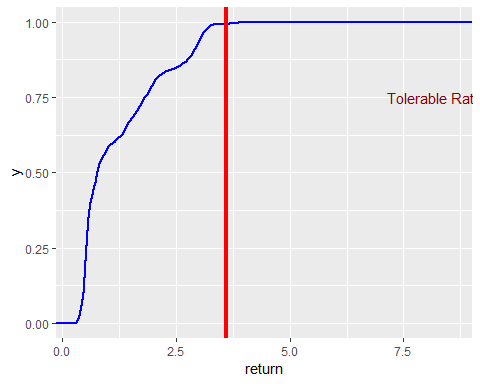
### Problem

We want to further characterize the distribution of up and down movements visually. Also we would like to repeat the analysis periodically for inclusion in management reports.

### Questions

1. How can we show the differences in the shape of ups and downs in HO2, especially given our tolerance for risk? We can use the HO2.df1 data frame with ggplot2 and the cumulative relative frequency function stat\_ecdf to begin to understand this data.

HO2.tol.pct = 0.95  
HO2.tol = quantile(HO2.df1$return, HO2.tol.pct)  
HO2.tol.label = paste("Tolerable Rate = ", round(HO2.tol, 2), sep = "")  
ggplot(HO2.df, aes(return, fill = direction)) +  
 stat\_ecdf(colour = "blue", size = 0.75) +  
 geom\_vline(xintercept = HO2.tol, colour = "red", size = 1.5) +  
 annotate("text", x = HO2.tol+5 , y = 0.75, label = HO2.tol.label, colour = "darkred")



1. How can we regularly, and reliably, analyze HO2 price movements? For this requirement, let’s write a function similar to data\_moments. Name this new function HO2\_movement().

## HO2\_movement(file, caption)  
## input: HO2 csv file from /data directory  
## output: result for input to kable in $table and xtable in $xtable;   
## data frame for plotting and further analysis in $df.  
## Example: HO2.data = HO2\_movement(file = "../data/nyhh02.csv", caption = "HO2 NYH")  
HO2\_movement = function(file = "../data/nyhh02.csv", caption = "Heating Oil No. 2: 1986-2016") {  
 # Read file and deposit into variable  
 HO2 = read.csv(file, header = T, stringsAsFactors = F)  
  
 # stringsAsFactors sets dates as character type  
 HO2 = na.omit(HO2) ## to clean up any missing data  
  
 #  
 # Construct expanded data frame  
 #  
 # size + direction, indicator of volatility  
 # date, length of DATE is length of return +1: omit 1st observation  
 # price, length of DHOILNYH is length of return +1: omit first observation  
 #  
 return = as.numeric(diff(log(HO2$DHOILNYH))) \* 100  
 size = as.numeric(abs(return))  
 direction = ifelse(return > 0, "up", ifelse(return < 0, "down", "same"))  
 date = as.Date(HO2$DATE[-1], "%m/%d/%Y")  
 price = as.numeric(HO2$DHOILNYH[-1])  
 HO2.df = na.omit(data.frame(  
 date = date,  
 price = price,  
 return = return,  
 size = size,  
 direction = direction  
 ))  
  
 require(dplyr)  
  
 ## 1: filter if necessary  
 pivot.table =filter(HO2.df, size > 0.5\*max(size))  
  
 ## 2: set up data frame for by-group processing  
 pivot.table =group\_by(HO2.df, direction)  
  
 ## 3: calculate the summary metrics  
 options(dplyr.width = Inf) ## to display all columns  
 HO2.count = length(HO2.df$return)  
 pivot.table =summarise(  
 pivot.table,  
 return.avg = mean(return),  
 return.sd = sd(return),  
 quantile.5 = quantile(return, 0.05),  
 quantile.95 = quantile(return, 0.95),  
 percent = (length(return)/HO2.count)\*100  
 )  
  
 # Construct transpose of pivot table with xtable()  
 require(xtable)  
 pivot.xtable = xtable(  
 t(pivot.table),  
 digits = 2,  
 caption = caption,  
 align=rep("r", 4),  
 table.placement="V"  
 )  
 HO2.caption = "Heating Oil No. 2: 1986-2016"  
 output.list = list(  
 table = pivot.table,  
 xtable = pivot.xtable,  
 df = HO2.df  
 )  
  
 return(output.list)  
}

Test HO2\_movement() with data and display results in a table with 2 decimal places.

knitr::kable(HO2\_movement(file = "../data/nyhh02.csv")$table, digits = 2)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| direction | return.avg | return.sd | quantile.5 | quantile.95 | percent |
| down | -1.77 | 1.99 | -4.78 | -0.19 | 47.52 |
| same | 0.00 | 0.00 | 0.00 | 0.00 | 3.63 |
| up | 1.76 | 1.75 | 0.18 | 4.82 | 48.86 |

Morale: more work today (build the function) means less work tomorrow (write yet another report).

1. Suppose we wanted to simulate future movements in HO2 returns. What distribution might we use to run those scenarios? Here, let’s use the MASS package’s fitdistr() function to find the optimal fit of the HO2 data to a parametric distribution. We will use the gamma distribution to simulate future heating oil #2 price scenarios.

library(MASS)

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':  
##   
## select

HO2.data = HO2\_movement(file = "../data/nyhh02.csv", caption = "HO2 NYH")$df  
str(HO2.data)

## 'data.frame': 7696 obs. of 5 variables:  
## $ date : Date, format: "1986-06-03" "1986-06-04" ...  
## $ price : num 0.393 0.378 0.39 0.385 0.373 0.365 0.389 0.394 0.398 0.379 ...  
## $ return : num -2.26 -3.89 3.13 -1.29 -3.17 ...  
## $ size : num 2.26 3.89 3.13 1.29 3.17 ...  
## $ direction: Factor w/ 3 levels "down","same",..: 1 1 3 1 1 1 3 3 3 1 ...

fit.gamma.up = fitdistr(  
 HO2.data[HO2.data$direction == "up", "return"],  
 "gamma",  
 hessian = TRUE  
)

## Warning in densfun(x, parm[1], parm[2], ...): NaNs produced

# fit.t.same = fitdistr(HO2.data[HO2.data$direction == "same", "return"], "gamma", hessian = TRUE) # a problem here is all observations = 0  
  
# gamma distribution properties  
fit.t.down = fitdistr(HO2.data[HO2.data$direction == "down", "return"], "t", hessian = TRUE)

## Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

## Warning in dt((x - m)/s, df, log = TRUE): NaNs produced  
  
## Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

## Warning in log(s): NaNs produced  
  
## Warning in log(s): NaNs produced

## Warning in dt((x - m)/s, df, log = TRUE): NaNs produced  
  
## Warning in dt((x - m)/s, df, log = TRUE): NaNs produced  
  
## Warning in dt((x - m)/s, df, log = TRUE): NaNs produced  
  
## Warning in dt((x - m)/s, df, log = TRUE): NaNs produced  
  
## Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

## Warning in log(s): NaNs produced  
  
## Warning in log(s): NaNs produced  
  
## Warning in log(s): NaNs produced

## Warning in dt((x - m)/s, df, log = TRUE): NaNs produced

## Warning in log(s): NaNs produced

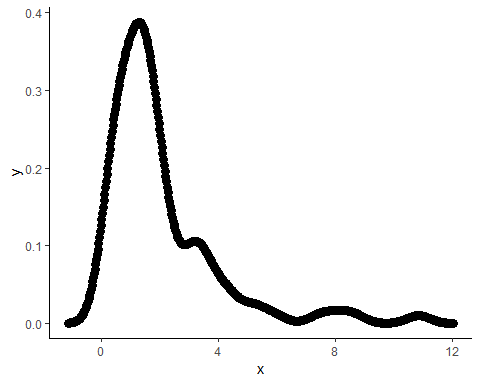
fit.t.down

## m s df   
## -1.30565487 0.91307703 2.50894659   
## ( 0.02170850) ( 0.02061868) ( 0.12442996)

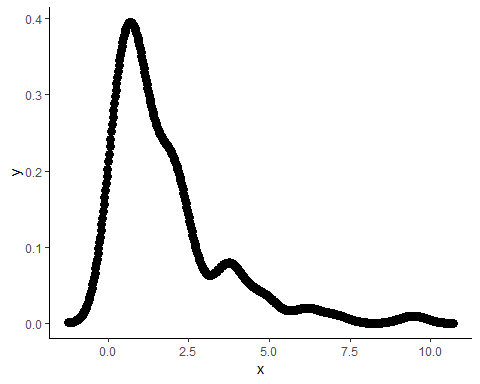
fit.gamma.down = fitdistr(-HO2.data[HO2.data$direction == "down", "return"], "gamma", hessian = TRUE)  
fit.gamma.down

## shape rate   
## 1.31056202 0.73969342   
## (0.02761041) (0.01889467)

# gamma density: up  
x\_up = rgamma(  
 100,  
 shape = fit.gamma.up$estimate[1],  
 scale = 1/fit.gamma.up$estimate[2]  
)  
density\_up = density(x\_up)  
d\_up = data.frame(x = density\_up$x, y = density\_up$y)  
  
# plot density as points  
ggplot(data = d\_up, aes(x = x, y = y)) +  
 geom\_point(size = 3) +  
 theme\_classic()



# gamma density: down  
x\_down = rgamma(  
 100,  
 shape = fit.gamma.down$estimate[1],  
 scale = 1/fit.gamma.down$estimate[2]  
)  
density\_down = density(x\_down)  
d\_down = data.frame(x = density\_down$x, y = density\_down$y)  
  
# plot density as points  
ggplot(data = d\_down, aes(x = x, y = y)) +  
 geom\_point(size = 3) +  
 theme\_classic()



## Conclusion

### Skills and Tools

The main tools implemented for this report include RStudio, with associated built-in functions, as well as the following additional packages:

* ggplot2
* gridExtra
* dplyr
* xtable
* MASS
* tseries
* stats

Since timeseries data on “New York Harbor No. 2” (HO2) oil prices was used, the ability to interpret stationarity, as well as autocorrelation and partial autocorrelation was required. Standard plotting techniques using ggplot2, followed by the “dickey-fuller” test, via the tseries package, help determine whether data transformations were required. In this assignment, the log transformation followed by a difference of 1-step was implemented on the data. Having the data properly transformed, allowed the use of pivot tables from the dplyr package. Particular aspects were highlighted, and related to real world events. Additionally, a number of steps were taken to generate a gamma distribution. As later explained in the “Business Remarks” (below), the ability to translate properties of the gamma distribution, including shape, and size allowed an explanation regarding key properites for the given data distribution.

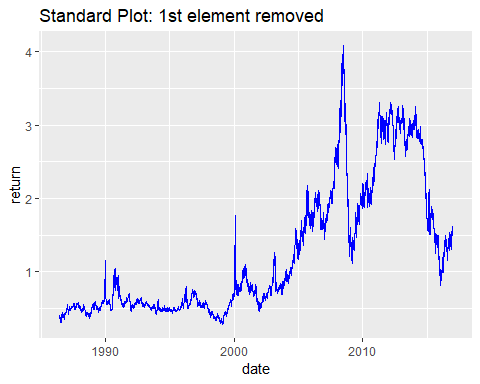
### Data Insights

A csv dataset containing prices for the “New York Harbor No. 2” (HO2) oil prices was analyzed for volatility, and associated earnings. The corresponding data was a two column dataset containing the DATE, as well as the DHOILNYH oil price. Simple computations were performed, in order to obtain the required dataframe structure:

return = HO2$DHOILNYH  
size = abs(return)  
direction = ifelse(return > 0, "up", ifelse(return < 0, "down", "same"))  
date = as.Date(HO2$DATE, "%m/%d/%Y")  
price = as.numeric(HO2$DHOILNYH)  
  
HO2.df = na.omit(data.frame(  
 date = date,  
 price = price,  
 return = return,  
 size = size,  
 direction = direction  
))

The first two columns of the HO2.df dataframe are self-evident, while the return is the original oil price, corresponding to the DHOILNYH column. The size was the absolute value of the oil price, which indicates the volatility of the price. A plot was easily generated:

library(ggplot2)  
library(gridExtra)  
ggplot(HO2.df, aes(x = date, y = return, group = 1)) +  
 geom\_line(colour = "blue") +  
 ggtitle('Standard Plot: 1st element removed')



It is evident that the corresponding data doesn’t exhibit any noticeable patterns. Specifically, the mean, variance, and autocorrelation is not constant over time. However, to prove non-stationarity, the dickey-fuller test was applied to the HO2.df$price:

library(tseries)  
adf.test(HO2.df$price)

##   
## Augmented Dickey-Fuller Test  
##   
## data: HO2.df$price  
## Dickey-Fuller = -2.1553, Lag order = 19, p-value = 0.5126  
## alternative hypothesis: stationary

Since the p-value = 0.5126, we fail to reject the null hypothesis, and conclude the given data is not stationary. To better prepare the data for analysis, some tranformation was required. Specifically, the log function was applied to the oil price, then the diff was aplied with the default lag=1 and difference=1.

return1 = as.numeric(diff(log(HO2$DHOILNYH)), 2)  
size1 = as.numeric(abs(return1)) # size is indicator of volatility  
direction1 = ifelse(return1 > 0, "up", ifelse(return1 < 0, "down", "same"))  
date1 = as.Date(HO2$DATE[-1], "%m/%d/%Y")  
price1 = as.numeric(diff(log(HO2$DHOILNYH)))  
HO2.df1 = na.omit(data.frame(  
 date = date1,  
 price = price1,  
 return = return1,  
 size = size1,  
 direction = direction1  
))

The same dickey-fuller test was executed on the adjusted dataset:

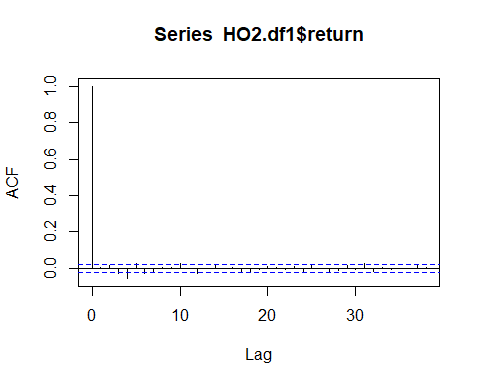
adf.test(HO2.df1$price)

## Warning in adf.test(HO2.df1$price): p-value smaller than printed p-value

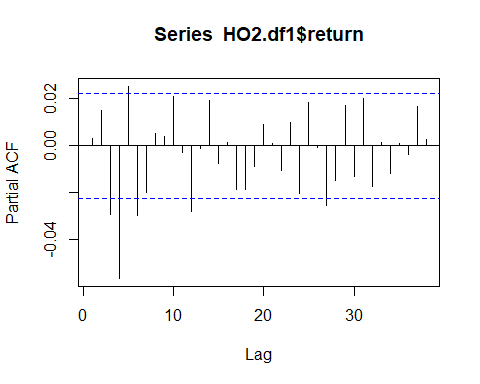
##   
## Augmented Dickey-Fuller Test  
##   
## data: HO2.df1$price  
## Dickey-Fuller = -20.58, Lag order = 19, p-value = 0.01  
## alternative hypothesis: stationary

Results indicate a p-value=0.01, which indicates the null hypothesis can be rejected. Specifically, the tranformed data is assumed to be stationary. Furthermore, autocorrelation seems very minor at lag=4, then a complete decay. Additionally, small partial-autocorrelation seem to exist with decaying characteristics.

require(stats)  
acf(HO2.df1$return)



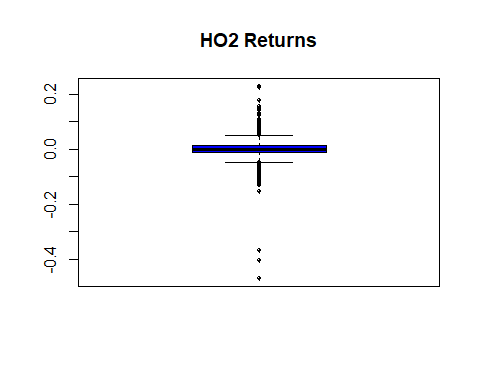
pacf(HO2.df1$return)



Overall, the transformed data indicate appropriate properties for time-series analysis.

Some initial analysis on the size, and return indicate volatility in both measures. Specifically, early 1990’s and 2000’s exhibit significant volatility for $DHOILNYH oil price’s, with some noticeable volatility around 1996. Corresponding skewness=-1.4353, and kurtosis=38.2595 suggests a non-normal distribution, with a longer negative tail indicating greater occurence of negative return.

boxplot(  
 as.vector(HO2.df1$return),  
 title = FALSE,  
 main = "HO2 Returns",  
 col = "blue",  
 cex = 0.5,  
 pch = 10  
)



Based on earlier generated visualization, both the size, and return have the same pattern, with a general difference of scale. The return has much smaller values ranging from 0 to 5, while size has a range of roughly 15, with volatility values spiking between 40-50.

Another price measure, include the price direction between timesteps (n, n+1). In the case for $DHOILNYH, oil price decreasing, or going “down”, occurred 3657 times. Similarly, oil price increasing, or going “up”, occurred 3760 times, while no change (i.e. “same”) occured 279 times.

To better summarize the above for all possible cases:

table(HO2.df1$return < 0)

##   
## FALSE TRUE   
## 4039 3657

table(HO2.df1$return > 0)

##   
## FALSE TRUE   
## 3936 3760

table(HO2.df1$direction)

##   
## down same up   
## 3657 279 3760

table(HO2.df1$return == 0)

##   
## FALSE TRUE   
## 7417 279

The above summary seem equally dispersed, when accounting only for directional changes. Specifically, the difference between TRUE, or FALSE for each given case is not significantly different. However, this uniformity may be a result of the implemented transformation. Earlier generated pivot table help visualize the overall distribution in a more concise fashion:

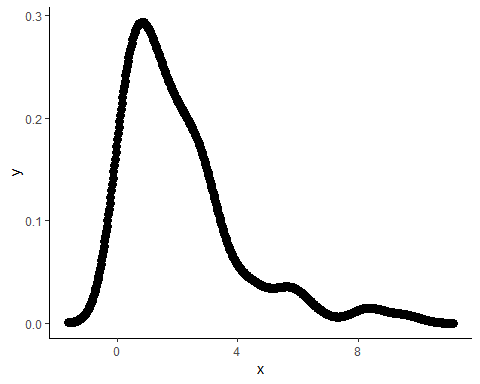
direction return.avg return.sd quantile.5 quantile.95 percent  
down -1.77 1.99 -4.78 -0.19 47.52  
same 0.00 0.00 0.00 0.00 3.63  
up 1.76 1.75 0.18 4.82 48.86

Additionally, creating a cumulative distribution function for HO2 oil price, with a tolerance 0.95, indicates that 95% of samples has a value adjusted return of 0.03595899 or less. The associated pivot table on the oil price:

direction return.avg return.sd quantile.5 quantile.95 percent  
down -1.77 1.99 -4.78 -0.19 47.52  
same 0.00 0.00 0.00 0.00 3.63  
up 1.76 1.75 0.18 4.82 48.86

Subsetting the HO2 dataset using only HO2.data$direction == "up" along with the return price, the fitdistr incidates a shape=1.30753665, with rate=0.74299635. These two parameters are related to the “erlang distribution”. Similarly, subsetting the HO2 dataset using only (-HO2.data[HO2.data$direction == "down" along with the return price, the corresponding erlang distribution patterns were shape=1.31056202, with rate=0.73969342. Generating a plot for the latter two distributions can easily be accomplished (as performed above):

x = rgamma(100, shape = fit.gamma.up$estimate[1], scale = 1/fit.gamma.up$estimate[2])  
den = density(x)  
dat = data.frame(x = den$x, y = den$y)  
  
# plot density as points  
ggplot(data = dat, aes(x = x, y = y)) +  
 geom\_point(size = 3) +  
 theme\_classic()



For mearningful interpretation, more attention may be required to reduce noise from the dataset.

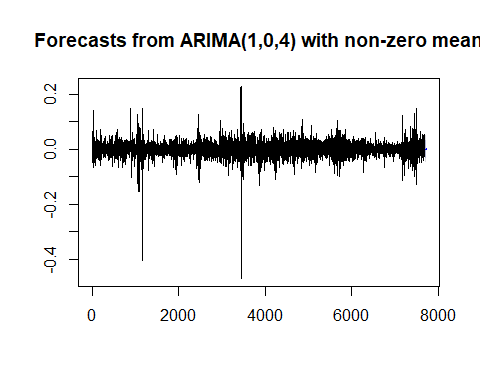
The original dataset indicated significant volatilty, with seasonal patterns, and nonstationarity. Specifically, spikes generally occurred during periods of war:

* 1990’s: gulf war
* 2000’s: aftermath of 9/11
* 2003-2011: war on Iraq

To restructure the dataset for timeseries analysis, the log function was applied, followed by a one step difference. Using the “dicker-fuller” test confirmed the latter produced a stationary distribution. Corresponding ACF and Partial ACF were computed, suggesting a moving average q=4.

An ARIMA model was computed for the return value of HO2. Forecasting 30 steps into the future:

library('forecast')  
library('tseries')  
  
fit = arima(HO2.df1$return, order=c(1, 0, 4))  
fcast = forecast(fit, h=30)  
plot(fcast)



### Business Remarks

The earlier calculated gamma distribution, was predicted on generating a random gamma distribution based on particular parameters. Since this computation does not directly provide means for prediction, the ARIMA model was chosen for timeseries modeling and prediction. However, without properly spliting the dataset into a train, and test set, no statistical measure can indicate the accuracy of the model.

Determining a mechanism for measuring the test set, could involve defining a tolerance level (p=0.05). Specifically, if predicted values were within the accepted tolerance, the prediction would be considered correct. The ARIMA model in this study did not utilize a test set to verify the accuracy of the model. Therefore, only forecasted return values can be generated, and at face value. Additionally, since predictions were performed on an adjusted dataset, an inverse log could rescale return value(s) to the original units.

The following options may reduce the variable costs associated with producing several products:

* set up contracts to set future prices
* buy raw material in bulk to supply a longer period of time
* establish an aggregated demand profile for H02
* find H02 substitutes
* (re)design product that do not need H02