National College of Ireland

Configuration Manual

MSc Research Project
FinTech

Kamran Raiysat
Student ID: x19102429

School of Computing
National College of Ireland

Supervisor: Noel Cosgrave

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Kamran Raiysat
Student ID:	x19102429
Programme:	MSc in FinTech **Year:** 2019-2020
Module:	Research Project
Lecturer:	Noel Cosgrave
Submission Due Date:	August 17, 2020
Project Title:	Portfolio Optimization Using ARIMA – Global Minimum Variance Approach

Word Count: 2310 **Page Count:** 15

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Kamran Raiysat

Date: 17/08/2020

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Kamran Raiysat
Student ID: x19102429

1 Introduction

The configuration manual is used to describe all the steps required to produce the same research. This document will highlight all the requirements followed to produce the thesis titled “Portfolio Optimization Using ARIMA – Global Minimum Variance Approach”. The detail of the hardware and software used will be shown. The configuration manual will also indicate the software used, versions and libraries. It will also describe the purpose for each software application. It will also include all the codes and graphs produced during research.

2 Hardware

The following set of computer hardware are used to produce the report:

Computer System:
HP Laptop
Model:
HP Laptop 15s-fq1xxx
Professor:
Intel ® Core™ i5 -1035G4
CPU:
1.10GHz 1.50GHz
RAM:
4.00 GB (3.76 GB usable)
Hard Disk:
238 GB SSD

3 Software

Operating System:
Microsoft Windows
Edition:
Windows 10 Home
Microsoft Word:
Microsoft word is used to draft the research report.
EndNote:
Endnote is used for referencing.
Microsoft Excel:
The Microsoft Excel is used for the following two reasons:

1. To Produce graph and table

2. To merge the stock values obtained from two different sources because KSE100 values are received from yahoo finance whereas other stocks values are taken from Quandl mode because the values of the KSE100 index were not available on the Quandl and therefore had to separately imported from yahoo finance. Similarly, the values of the other stocks were also not available on the yahoo finance which means data was collected from two different sources and therefore the dates were varying. In order to remove the difference, the weekly closing prices of the stocks were combined into one file using excel for analysis purpose. **Update the current file path (D:\Codes\ARIMA_forecasted_xls) to new path where file will be saved. The file is already available in the Code folders.**

Programming Environment:
R Studio

Version of the Environment:
R version 3.6.2

Language of the Environment:
R Language

Packages and Libraries

Package (Quantmod)
This is a R Package designed to assist the quantative traders to develop, test and deploy the trading models based on the statistics [1]
Package (PerformanceAnalytics)
According to R, "Collection of econometric functions for performance and risk analysis. In addition to standard risk and performance metrics, this package aims to aid practitioners and researchers in utilizing the latest research in analysis of non-normal return streams. In general, it is most tested on return (rather than price) data on a regular scale, but most functions will work with irregular return data as well, and increasing numbers of functions will work with P&L or price data where possible" [2].
Package (Quandl)
The quandl is a database that provides financial, economic and other datasets to the professionals through API to meet their needs.
Package (Forecast)
According to the official document of R, "Methods and tools for displaying and analysing univariate time series forecasts including exponential smoothing via state space models and automatic ARIMA modelling" [3].
Other Packages
Package(tseries)
Package(readxl)

The Following libraries are also used
library(quantmod)
library(PerformanceAnalytics)
library(Quandl)
library(forecast)
library(tseries)
library(readxl)

Key to get access to Quandl database
Quandl.api_key("KxuTs9kcHf_aNvQk-k6s")

4 R Codes

```
################################################################################
###################### PORTFOLIO OPTIMIZATION GMV VS ARIMA - GMV
################################################################################
# Predicting Stock Pricing using ARIMA

# Load Libraries

library(quantmod)
library(PerformanceAnalytics)
library(Quandl)
library(forecast)
Quandl.api_key("KxuTs9kcHf_aNvQk-k6s") # Key to retrieve information from Quandl

# Setting start and ending dates

start <- "2015-01-01"
end <- "2019-12-31"

# Multiple Stocks selected from KSE 100 Index based on Market Cap (Top 6)

getSymbols("^KSE", from=start, to=end, periodicity = "weekly")
KSE <- KSE$KSE.Close
m_kse = sum(is.na(KSE))

ENGRO <- Quandl("PSX/ENGRO.5", type="xts", start_date=start, end_date=end, collapse =
"weekly")
m_engro = sum(is.na(ENGRO))
FFC <- Quandl("PSX/FFC.5", type="xts", start_date=start, end_date=end, collapse =
"weekly")
m_ffc = sum(is.na(FFC))
HUBC <- Quandl("PSX/HUBC.5", type="xts", start_date=start, end_date=end, collapse =
"weekly")
m_hubc = sum(is.na(HUBC))
OGDC <- Quandl("PSX/OGDC.5", type="xts", start_date=start, end_date=end, collapse =
"weekly")
m_ogdc = sum(is.na(OGDC))
HBL <- Quandl("PSX/HBL.5", type="xts", start_date=start, end_date=end, collapse =
"weekly")
m_hbl = sum(is.na(HBL))
MCB <- Quandl("PSX/MCB.5", type="xts", start_date=start, end_date=end, collapse =
"weekly")
m_mcb = sum(is.na(MCB))

# Checking Missing Values
```

```
All_tickers = c( "KSE","ENGRO", "FFC", "HUBC", "OGDC", "HBL", "MCB")
Missing_values = cbind.data.frame(m_kse, m_engro, m_ffc, m_hubc, m_ogdc, m_hbl, m_mcb) #, sum(is.na(HUBC), sum(is.na(OGDC), sum(is.na(HBL), sum(is.na(MCB)))
colnames(Missing_values) <- All_tickers
Missing_values

# Ploting the Stock Prices to Explore the data

par(mfrow=c(3,3))
plot(KSE, type = "l", main = "KSE")
plot(ENGRO, type = "l", main = "ENGRO ")
plot(FFC, type = "l", main = "FFC ")
plot(HUBC, type = "l",main = "HUBC ")
plot(OGDC, type = "l", main = "OGDC")
plot(HBL, type = "l", main = "HBL")
plot(MCB, type = "l", main = "MCB")

# Checking Serial correlation using Ljung-Box

# AT LAG VALUE OF 20
Box.test(KSE, lag = 20, type = "Ljung-Box")
Box.test(ENGRO, lag = 20, type = "Ljung-Box")
Box.test(FFC, lag = 20, type = "Ljung-Box")
Box.test(HUBC, lag = 20, type = "Ljung-Box")
Box.test(OGDC, lag = 20, type = "Ljung-Box")
Box.test(HBL, lag = 20, type = "Ljung-Box")
Box.test(MCB, lag = 20, type = "Ljung-Box")

# Converting series to Ln format using initial 80% data as training sample

lnKSE = log(KSE[1:208])
lnENGRO = log(ENGRO[1:208])
lnFFC = log(FFC[1:208])
lnHUBC = log(HUBC[1:208])
lnOGDC = log(OGDC[1:208])
lnHBL = log(HBL[1:208])
lnMCB = log(MCB[1:208])

# Finding Stationery of Data on Log values before and after differencing, using Dickey Fuller Test

# Stationery Without Differencing

s_lnkse = adf.test(lnKSE)
s_lnengro = adf.test(lnENGRO)
s_lnffc = adf.test(lnFFC)
s_lnhubc = adf.test(lnHUBC)
s_lnogdc = adf.test(lnOGDC)
s_lnhbl = adf.test(lnHBL)
s_lnmcb = adf.test(lnMCB)
```

```
S_data = cbind.data.frame(s_lnkse$p.value, s_lnengro$p.value, s_lnffc$p.value, s_lnhubc$p.value, s_lnogdc$p.value, s_lnhbl$p.value, s_lnmcb$p.value)

colnames(S_data) <- All_tickers
S_data

# stationery at difference

difflnKSE <- diff(lnKSE, 1)
difflnKSE <- na.omit(difflnKSE)
d_lnkse = adf.test(difflnKSE)

# ENGRO Stock

difflnENGRO <- diff(lnENGRO, 1)
difflnENGRO <- na.omit(difflnENGRO)
d_lnengro = adf.test(difflnENGRO)

# FFC Stock

difflnFFC <- diff(lnFFC, 1)
difflnFFC <- na.omit(difflnFFC)
d_lnffc = adf.test(difflnFFC)

# HUBC Stock

difflnHUBC <- diff(lnHUBC, 1)
difflnHUBC <- na.omit(difflnHUBC)
d_lnhubc <- adf.test(difflnHUBC)

# OGDC Stock

difflnOGDC <- diff(lnOGDC, 1)
difflnOGDC <- na.omit(difflnOGDC)
d_lnogdc = adf.test(difflnOGDC)

# HBL Stock

difflnHBL <- diff(lnHBL, 1)
difflnHBL <- na.omit(difflnHBL)
d_lnhbl <- adf.test(difflnHBL)

# MCB Stock

difflnMCB <- diff(lnMCB, 1)
difflnMCB <- na.omit(difflnMCB)
d_lnmcb = adf.test(difflnMCB)

d_data = cbind.data.frame(d_lnkse$p.value, d_lnengro$p.value, d_lnffc$p.value, d_lnhubc$p.value, d_lnogdc$p.value, d_lnhbl$p.value, d_lnmcb$p.value)
```

```
colnames(d_data) <- All_tickers
d_data

stationery <- rbind(S_data, d_data)
P_value_at = rbind("W/o Diff.", "With Diff")
stationery <- cbind(P_value_at, stationery)
stationery

# ploting ACF

par(mfrow=c(2, 4))
acf(lnKSE, lag.max = 20)
acf(lnENGRO, lag.max = 20)
acf(lnFFC, lag.max = 20)
acf(lnHUBC, lag.max = 20)
acf(lnOGDC, lag.max = 20)
acf(lnHBL, lag.max = 20)
acf(lnMCB, lag.max = 20)

# ploting PACF
par(mfrow=c(2, 4))
pacf(lnKSE, lag.max = 20)
pacf(lnENGRO, lag.max = 20)
pacf(lnFFC, lag.max = 20)
pacf(lnHUBC, lag.max = 20)
pacf(lnOGDC, lag.max = 20)
pacf(lnHBL, lag.max = 20)
pacf(lnMCB, lag.max = 20)

# Converting to time Series

PAKSE <- ts(lnKSE, start = c(2015, 1), frequency = 52)
PAENGRO <- ts(lnENGRO, start = c(2015, 1), frequency = 52)
PAFFC <- ts(lnFFC, start = c(2015, 1), frequency = 52)
PAHUBC <- ts(lnHUBC, start = c(2015, 1), frequency = 52)
PAOGDC <- ts(lnOGDC, start = c(2015, 1), frequency = 52)
PAHBL <- ts(lnHBL, start = c(2015, 1), frequency = 52)
PAMCB <- ts(lnMCB, start = c(2015, 1), frequency = 52)

# Running Auto ARIMA

fitlnKSE <- auto.arima(PAKSE)
fitlnENGRO <- auto.arima(PAENGRO)
fitlnFFC <- auto.arima(PAFFC)
fitlnHUBC <- auto.arima(PAHUBC)
fitlnOGDC <- auto.arima(PAOGDC)
fitlnHBL <- auto.arima(PAHBL)
fitlnMCB <- auto.arima(PAMCB)

# forecast Values from ARIMA
```

```
forKSE = forecast(fitlnKSE, h=52)
forENGRO = forecast(fitlnENGRO, h=52)
forFFC = forecast(fitlnFFC, h=52)
forHUBC = forecast(fitlnHUBC, h=52)
forOGDC = forecast(fitlnOGDC, h=52)
forHBL = forecast(fitlnHBL, h=52)
forMCB = forecast(fitlnMCB, h=52)

# Ploting the forecasted values

par(mfrow=c(3,3))
plot(forKSE, ylab="KSE")
plot(forENGRO, ylab="ENGRO")
plot(forFFC, ylab="FFC")
plot(forHUBC, ylab="HUBC")
plot(forOGDC, ylab="OGDC")
plot(forHBL, ylab="HBL")
plot(forMCB, ylab="MCB")

# EXTRACTING MEAN VALUES OF FORECASTING VALUES

forKSE_ext=as.numeric(forKSE$mean)
forENGRO_ext=as.numeric(forENGRO$mean)
forFFC_ext=as.numeric(forFFC$mean)
forHUBC_ext=as.numeric(forHUBC$mean)
forOGDC_ext=as.numeric(forOGDC$mean)
forHBL_ext=as.numeric(forHBL$mean)
forMCB_ext=as.numeric(forMCB$mean)

# CONVERTING FORECATED LOG VALUES

forKSE_final=exp(forKSE_ext)
forENGRO_final=exp(forENGRO_ext)
forFFC_final=exp(forFFC_ext)
forHUBC_final=exp(forHUBC_ext)
forOGDC_final=exp(forOGDC_ext)
forHBL_final=exp(forHBL_ext)
forMCB_final=exp(forMCB_ext)

#Merging the forecasted values

Forcasted_values= cbind(forKSE_final, forENGRO_final, forFFC_final, forHUBC_final,
forOGDC_final, forHBL_final, forMCB_final)
colnames(Forcasted_values) = c("KSE", "ENGRO", "FFC", "HUBC", "OGDC", "HBL",
"MCB")
Forcasted_Data = data.frame(Forcasted_values)
head(Forcasted_values)

# PERCENTAGE ERROR

# COLUMN HEADING OF MODEL EVALUATION
```

```
col_headinds <- c("Actual", "Forecasted")

# KSE
EveKSE <- data.frame(KSE[209:260], forKSE_final)
colnames(EveKSE) <- col_headinds
MPEKSE= mean((EveKSE$Actual-EveKSE$Forecasted)/EveKSE$Actual)
MAPEKSE= mean(abs(EveKSE$Actual-EveKSE$Forecasted)/EveKSE$Actual)

# ENGRO
EveENGRO <- data.frame(ENGRO[209:260], forENGRO_final)
colnames(EveENGRO) <- col_headinds
MPEENGRO= mean((EveENGRO$Actual- EveENGRO$Forecasted)/EveENGRO$Actual)
MAPEENGRO=mean(abs((EveENGRO$Actual-
EveENGRO$Forecasted)/EveENGRO$Actual))

# FFC
EveFFC <- data.frame(FFC[209:260], forFFC_final)
colnames(EveFFC) <- col_headinds
MPEFFC=mean((EveFFC$Actual- EveFFC$Forecasted)/EveFFC$Actual)
MAPEFFC=mean(abs((EveFFC$Actual- EveFFC$Forecasted)/EveFFC$Actual))

# HUBC
EveHUBC <- data.frame(HUBC[209:260], forHUBC_final)
colnames(EveHUBC) <- col_headinds
MPEHUBC= mean((EveHUBC$Actual- EveHUBC$Forecasted)/EveHUBC$Actual)
MAPEHUBC= mean(abs(EveHUBC$Actual- EveHUBC$Forecasted)/EveHUBC$Actual)

# OGDC
EveOGDC <- data.frame(OGDC[209:260], forOGDC_final)
colnames(EveOGDC) <- col_headinds
MPEOGDC=mean((EveOGDC$Actual- EveOGDC$Forecasted)/EveOGDC$Actual)
MAPEOGDC=mean(abs((EveOGDC$Actual- EveOGDC$Forecasted)/EveOGDC$Actual))

# HBL
EveHBL <- data.frame(HBL[209:260], forHBL_final)
colnames(EveHBL) <- col_headinds
MPEHBL=mean((EveHBL$Actual- EveHBL$Forecasted)/EveHBL$Actual)
MAPEHBL=mean(abs((EveHBL$Actual- EveHBL$Forecasted)/EveHBL$Actual))

# MCB
EveMCB <- data.frame(MCB[209:260], forMCB_final)
colnames(EveMCB) <- col_headinds
MPEMCB=mean((EveMCB$Actual- EveMCB$Forecasted)/EveMCB$Actual)
MAPEMCB=mean(abs((EveMCB$Actual- EveMCB$Forecasted)/EveMCB$Actual))

# MEAN ABSOLUTE PERCENTAGE ERRORS

MPError = c(MPEKSE*100, MPEENGRO*100, MPEFFC*100, MPEHUBC*100,
MPEOGDC*100, MPEHBL*100, MPEMCB*100)
```

```
MAPError = c(MAPEKSE*100, MAPEENGRO*100, MAPEFFC*100, MAPEHUBC*100, MAPEOGDC*100, MAPEHBL*100, MAPEMCB*100)

ModelEveluation = round(rbind(MPError, MAPError), digits = 2)
colnames(ModelEveluation) = c("KSE", "ENGRO", "FFC", "HUBC", "OGDC", "HBL", "MCB")
ModelEveluation

#####################################################
####### Global Minimim Variance ###################
#####################################################

####### Portfolio Based on Actual Values###########
# Converting back the values from log

Tr_KSE = exp(lnKSE)
Tr_ENGRO = exp(lnENGRO)
Tr_FFC = exp(lnFFC)
Tr_HUBC = exp(lnHUBC)
Tr_OGDC = exp(lnOGDC)
Tr_HBL = exp(lnHBL)
Tr_MCB = exp(lnMCB)
Tr_portprice = merge(Tr_ENGRO, Tr_FFC, Tr_HUBC, Tr_OGDC, Tr_HBL, Tr_MCB)
tickers = c("ENGRO", "FFC", "HUBC", "OGDC", "HBL", "MCB")
colnames(Tr_portprice) = tickers
head(Tr_portprice)

Con_KSE =data.frame((Tr_KSE))
Total_sample = cbind(Tr_portprice, Con_KSE$KSE.Close)
Tr_KSE1 = Total_sample$Con_KSE.KSE.Close
head(Tr_KSE1)

Tr_portprice_returns = ROC(Tr_portprice, type = "discrete")[-1,]

Tr_KSE_returns = dailyReturn(Tr_KSE1)

# To calculate the beta of a portfolio, First calculate the beta of each stock in the final Portfolio.

# Calculate individual betas
BetaTrS <- c()
for (i in 1:ncol(Tr_portprice))
{
  b2 <- dailyReturn(Tr_portprice[, i])
  BetaTrS[i] <- CAPM.beta(b2, Tr_KSE_returns)
}

BetaTrS

### The Regression Solution ###
```

```
numAssets <- length(tickers)

GMSPortRets <- as.matrix(Tr_portprice_returns)

## choose 1st asset as regressand
y <- GMSPortRets[ , 1]

## compute minus excess returns
X <- GMSPortRets[ , 1] - GMSPortRets[ , 2:numAssets]

## run regression
solR <- lm(y ~ X)

# The optimal weights of the assets in the portfolio are:"
opt_weights <- as.vector(c(1 - sum(coef(solR)[-1L]), coef(solR)[-1L]))

IndStockWeights<-data.frame(opt_weights*100, tickers)
IndStockWeights$opt_weights...100 <- round((IndStockWeights$opt_weights...100), digits =
2)
IndStockWeights


# The portfolio beta is therefore:
GMSport.Betas <- sum(BetaTrS*opt_weights)
GMSport.Betas
mean(BetaTrS)


####### Portfolio Based on Actual and Forecated Values###########

############## KSE forecated Values ########################


Actual_Port = merge(ENGRO, FFC, HUBC, OGDC, HBL, MCB)
KSE_convert = data.frame(KSE)
KSE_convert = KSE_convert$KSE.Close
Actual_Data = cbind(KSE_convert, Actual_Port)
Actual_Data
training_Data = Actual_Data[1:208]
test_Data = Actual_Data[209:260]
Forcasted_Data

# Import Combined File

ARIMA_forecasted <- read_excel("D:/Project/Old/code/ARIMA_forecasted_xls.xlsx")
Mix_Data = as.xts(as.data.table(ARIMA_forecasted))


AR_KSE = Mix_Data$KSE
AR_portprice = Mix_Data[,-1]
```

```
AR_portprice_returns = ROC(AR_portprice, type = "discrete")[-1]

AR_KSE_returns = ROC(AR_KSE, type = "discrete")[-1]

# To calculate the beta of a portfolio, First calculate the beta of each stock in the final Portfolio.

# Calculate individual betas
BetaAR <- c()
for (i in 1:ncol(AR_portprice))
{
  b3 <- AR_portprice_returns[, i]
  BetaAR[i] <- CAPM.beta(b3, AR_KSE_returns)
}

BetaAR

### The Regression Solution ###

AR_numAssets <- length(tickers)

ARSPortRets <- as.matrix(AR_portprice_returns)

## choose 1st asset as regressand
ay <- ARSPortRets[ , 1]

## compute minus excess returns
aX <- ARSPortRets[ , 1] - ARSPortRets[ , 2:AR_numAssets]

## run regression
solR_ar <- lm(ay ~ aX)

# The optimal weights of the assets in the portfolio are:"
opt_weights_ar <- as.vector(c(1 - sum(coef(solR_ar)[-1L]), coef(solR_ar)[-1L]))

IndStockWeights_ar<-data.frame(opt_weights_ar*100, tickers)
IndStockWeights_ar$opt_weights_ar...100                                    <-
round(IndStockWeights_ar$opt_weights_ar...100, digits = 2)
IndStockWeights_ar

# The portfolio beta is therefore:
ARSport.Betas <- sum(BetaAR*opt_weights_ar)
ARSport.Betas
mean(BetaAR)

Beta_values = cbind.data.frame(GMSport.Betas, ARSport.Betas)
Beta_values
beta_mean = cbind(mean(BetaAR)*100, mean((BetaTrS)*100))

#Weights
```

```
weights_com = cbind.data.frame(tickers, IndStockWeights$opt_weights...100 , IndStockWeights_ar$opt_weights_ar...100)
colnames(weights_com) = c("tickers", "GMV Weights", "ARIMA-GMV Weights")
weights_com$Change = round((-opt_weights + opt_weights_ar)*100/opt_weights, digits = 2)
weights_com

Last_year = (Actual_Port[208])
Current_year = (Actual_Port[260])

RoChange= rbind.data.frame(as.numeric(Last_year), as.numeric(Current_year))
RoChange = transpose(RoChange)
RoChange$Rate = round((RoChange$V2 - RoChange$V1)*100/RoChange$V1, digits = 2)

Final_result = cbind.data.frame(weights_com$tickers, RoChange$Rate, weights_com$Change)
colnames(Final_result) = c("Symbols", "Price Change", "Weight Change")
Final_result

# Optional Graph not inluding on the report
par(mfrow=c(1,2))
plot(Final_result$`Price Change`, type = "l", xlab = "Stock", ylab = "% Change", main = "Change in Price")
plot(Final_result$`Weight Change`, type = "l", xlab = "Stock", ylab = "% Change", main = "Change in Weight")
```

5 Graphs

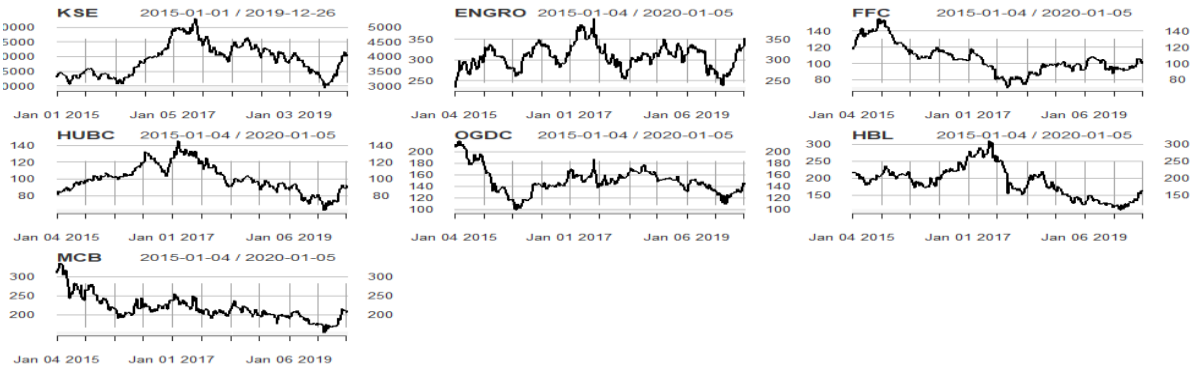

Figure 2: Weekly Closing Prices

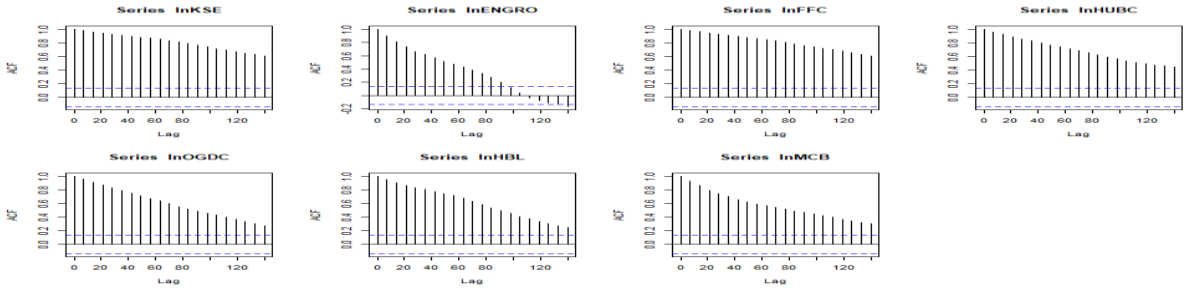

Figure 3: Autocorrelation Function

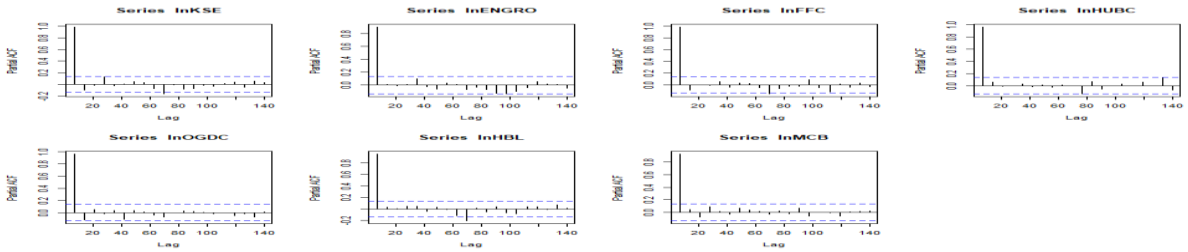

Figure 4: Partial Autocorrelation Function

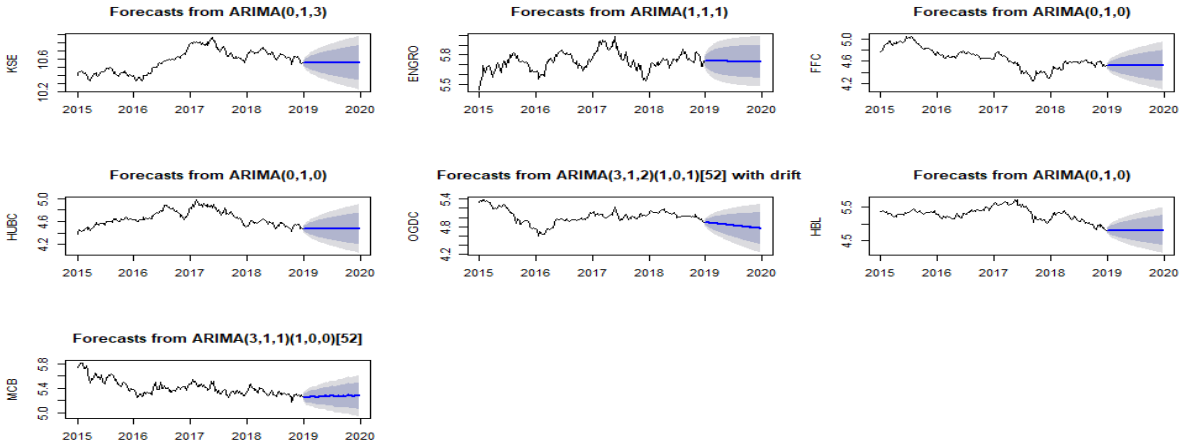

Figure 5: Weekly Closing Prices

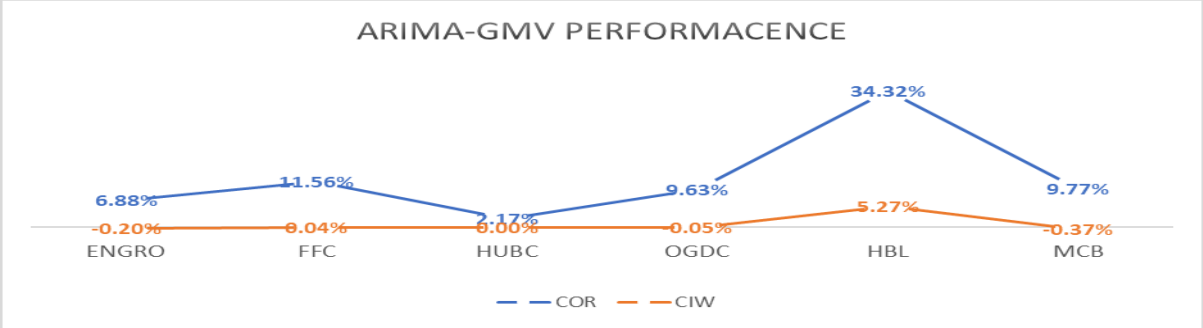

Figure 7: Change of Rate (COR) vs Change in Weights (CIW)

References

[1] Quantmod, "Quantitative Financial Modelling & Trading Framework for R," 2008. [Online]. Available: https://www.quantmod.com/. [Accessed 16 August 2020].

[2] B. G. Peterson, P. Carl, K. Boudt, R. Bennett, J. Ulrich, E. Zivot and D. C. e. al, "Econometric Tools for Performance and Risk Analysis," 2020.

[3] R. Hyndman, G. Athanasopoulos, C. Bergmeir, G. Caceres, L. Chhay and e. al, "Forecasting Functions for Time Series and Linear Models," 2020.