Application of Econometric Models with RStudio

Deri Siswara

2025-07-06

Table of contents

Welcome **1**

1 Basics of R **3**
1.1 Introduction 3
1.2 Types of Objects in R 5
1.2.1 Vector 5
1.2.2 Factor 7
1.2.3 List 7
1.2.4 Data Frame 9
1.3 Data Frame Management 9
1.3.1 R Package 10
1.3.2 Data Management With `dplyr` 10
1.4 Visualization 13
1.4.1 Histogram 13
1.4.2 Box Plot 13
1.4.3 Barplot 14
1.4.4 Pie Chart 15
1.4.5 Scatter Plot 16

2 Basic Linear Regression **17**
2.1 EDA 17
2.2 Ordinary Least Square (OLS) 19
2.3 Diagnostic Gauss Markov 19

3 Univariate Time Series **23**
3.1 Data EDA 23
3.2 ACF and PACF Plot 24
3.3 ARIMA 27
3.4 ARCH-GARCH 31

4 Multivariate Time Series **43**
4.1 VAR/VECM 43
4.2 SVAR 58

4.3 ARDL . . . 67

5 Panel Data Regression 77
5.1 Static Panel Data . . . 77
5.1.1 Pooled OLS . . . 78
5.1.2 Fixed Effects Model . . . 79
5.1.3 Random Effects Model . . . 80
5.1.4 Hausman Test . . . 81
5.1.5 Model Diagnostics . . . 81
5.2 Dynamic Panel Data . . . 82
5.2.1 First Difference GMM . . . 83
5.2.2 System GMM . . . 84
5.2.3 Model Diagnotics . . . 84
5.2.4 Speed of Adjustment . . . 87
5.2.5 Half Time . . . 87
5.2.6 Short Run and Long Run Coefficients . . . 87

6 Spatial Regression 89
6.1 Library . . . 89
6.2 Cross-Section . . . 90
6.2.1 OLS Model . . . 90
6.2.2 Weight Matrix . . . 91
6.2.3 Moran Test and Plot . . . 92
6.2.4 LM Test . . . 93
6.2.5 SAR Model . . . 94
6.2.6 Impacts (Spillover) . . . 95
6.2.7 SEM Model . . . 95
6.3 Spatial Panel . . . 96
6.3.1 Static Panel Regression . . . 96
6.3.2 Depndency Test . . . 97
6.3.3 Maps Visualization . . . 98
6.3.4 Function for Spatial Panel Evalutation . . . 99
6.3.5 Spatial Panel Model with Contiguity Weight Matrix . . . 100
6.3.6 Spatial Panel Model with KNN Weight Matrix . . . 101
6.3.7 Best Model . . . 102
6.3.8 Impacts (Spillovr) - Only for SAR Model . . . 103

7 Time Series Spillover - GVAR 105
7.1 Library . . . 105
7.2 Data: Diebold-Yilmaz 2012 . . . 106
7.3 VAR Model . . . 106
7.4 Volatility Spillover DY-2012 . . . 109
7.5 Dynamic Spillover Index / rolling-sample total volatility spillover 110
7.6 Directional volatility spillovers . . . 111
7.7 Connectedness Network . . . 114

8 Multivariate GARCH **117**
8.1 DCC-GARCH 117

9 NARDL **127**

Welcome

This is the code version of *Application of Econometric Models with RStudio*, a book released in 2023 by IPB Press. The book was written by Muhammad Firdaus, Tony Irawan, Fahmi Salam Ahmad, Hermanto Siregar, Deri Siswara, and Rodi Jakariya. You can order the full version here, which includes more detailed explanations.

This book aims to help students and researchers apply econometric models in analysis using RStudio software. It begins with an introduction to the use of R programming language and RStudio as an IDE. The book covers topics on theory and application of OLS models, including testing Gauss-Markov conditions. Additionally, it presents the application of time-series models such as ARMA, GARCH, VAR, VECM, ARDL, SVAR., static and dynamic panel data analysis using RStudio's instruments that accommodate heteroscedasticity and autocorrelation assumptions. Finally, there is a discussion of spatial model applications for analysis with spatial or regional elements. In this latest publication update includes three additional chapters: spillover analysis on time series data; DCC-GARCH; Non-linear ARDL. Updates will be made frequently.

This book may contain bugs/errors which readers can report at *derisiswarads@gmail.com*

Chapter 1

Basics of R

1.1 Introduction

```
A <- 2
```

```
A # Print A
```

```
[1] 2
```

```
A = 2
A
```

```
[1] 2
```

```
B <- "Halo Semua"
B
```

```
[1] "Halo Semua"
```

```
a<-10 # Space is not sensitive but lettercase is sensitive.
A
```

```
[1] 2
```

```
a
```

```
[1] 10
```

```
# Arithmetic operation
x <- 5
y <- 3
x + y
```

```
[1] 8
```

```r
x - y
```

```
[1] 2
```

```r
x * y
```

```
[1] 15
```

```r
x / y
```

```
[1] 1.666667
```

```r
# Logic operation
a <- TRUE
b <- FALSE
a & b
```

```
[1] FALSE
```

```r
a | b
```

```
[1] TRUE
```

```r
!a
```

```
[1] FALSE
```

```r
x <- 5
y <- 3
x > y
```

```
[1] TRUE
```

```r
x < y
```

```
[1] FALSE
```

```r
x == y
```

```
[1] FALSE
```

```r
x >= y
```

```
[1] TRUE
```

```r
x <= y
```

```
[1] FALSE
```

1.2 Types of Objects in R

1.2.1 Vector

```
a1 <- c(2,4,7,3) # Numeric vector
a2 <- c("one","two","three") # Character vector
a3 <- c(TRUE,TRUE,TRUE,FALSE,TRUE,FALSE) # Logical vector
```

```
a1
```

```
[1] 2 4 7 3
```

```
a3[4]
```

```
[1] FALSE
```

```
a2[c(1,3)]
```

```
[1] "one"   "three"
```

```
a1[-1]
```

```
[1] 4 7 3
```

```
a1[2:4]
```

```
[1] 4 7 3
```

```
a <- c(1, 2, 3)
b <- c(4, 5, 6)
c <- c(a, b)
c
```

```
[1] 1 2 3 4 5 6
```

```
c[1:3]
```

```
[1] 1 2 3
```

```
d <- a + b
d
```

```
[1] 5 7 9
```

```
a4 <- 1:12
b1 <- matrix(a4,3,4)
b2 <- matrix(a4,3,4,byrow=TRUE)
b3 <- matrix(1:14,4,4)
```

```
Warning in matrix(1:14, 4, 4): data length [14] is not a sub-multiple or
multiple of the number of rows [4]
```

```
b1
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
```

```
b2
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
```

```
b3
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    2    6   10   14
[3,]    3    7   11    1
[4,]    4    8   12    2
```

```
b2[2,3]
```

```
[1] 7
```

```
b2[1:2,]
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
```

```
b2[c(1,3),-2]
```

```
     [,1] [,2] [,3]
[1,]    1    3    4
[2,]    9   11   12
```

```
dim(b2)
```

```
[1] 3 4
```

```
m1 <- matrix(c(1, 2, 3, 4, 5, 6), nrow = 2, ncol = 3)
m2 <- matrix(c(7, 8, 9, 10, 11, 12), nrow = 2, ncol = 3)
```

```
m3 <- m1 + m2
m3
```

```
     [,1] [,2] [,3]
[1,]    8   12   16
[2,]   10   14   18
```

```
m4 <- m1 %*% t(m2)
m4
```

```
     [,1] [,2]
[1,]   89   98
[2,]  116  128
```

1.2.2 Factor

```
a5 <- c("A","B","AB","O")
d1 <- factor(a5)
levels(d1)
```

```
[1] "A"  "AB" "B"  "O"
```

```
levels(d1) <- c("Darah A","Darah AB","Darah B","Darah O")
d1
```

```
[1] Darah A  Darah B  Darah AB Darah O
Levels: Darah A Darah AB Darah B Darah O
```

```
a6 <- c("SMA","SD","SMP","SMA","SMA","SMA","SMA","SMA","SMA","SMA","SMA","SMA","SMA")
d5 <- factor(a6, levels=c("SD","SMP","SMA")) # Skala pengukuran ordinal
levels(d5)
```

```
[1] "SD"  "SMP" "SMA"
```

```
d5
```

```
 [1] SMA SD  SMP SMA SMA SMA SMA SMA SMA SMA SMA SMA SMA
Levels: SD SMP SMA
```

1.2.3 List

```
a1; b2; d1
```

```
[1] 2 4 7 3
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
```

```
[1] Darah A  Darah B  Darah AB Darah O
Levels: Darah A Darah AB Darah B Darah O
```

```
e1 <- list(a1,b2,d1)
e2 <- list(vect=a1,mat=b2,fac=d1)
e1
```

```
[[1]]
[1] 2 4 7 3

[[2]]
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12

[[3]]
[1] Darah A  Darah B  Darah AB Darah O
Levels: Darah A Darah AB Darah B Darah O
```

```
e2
```

```
$vect
[1] 2 4 7 3

$mat
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12

$fac
[1] Darah A  Darah B  Darah AB Darah O
Levels: Darah A Darah AB Darah B Darah O
```

```
e1[[1]][2]
```

```
[1] 4
```

```
e2$fac
```

```
[1] Darah A  Darah B  Darah AB Darah O
Levels: Darah A Darah AB Darah B Darah O
```

```
e2[2]
```

```
$mat
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
```

```
names(e2)
```

```
[1] "vect" "mat"  "fac"
```

1.2.4 Data Frame

```
Angka <- 11:15
Huruf <- factor(LETTERS[6:10])
f1 <- data.frame(Angka,Huruf)
f1
```

```
  Angka Huruf
1    11     F
2    12     G
3    13     H
4    14     I
5    15     J
```

```
f1[1,2]
```

```
[1] F
Levels: F G H I J
```

```
f1$Angka
```

```
[1] 11 12 13 14 15
```

```
f1[,"Huruf"]
```

```
[1] F G H I J
Levels: F G H I J
```

```
colnames(f1)
```

```
[1] "Angka" "Huruf"
```

```
str(f1)
```

```
'data.frame':   5 obs. of  2 variables:
 $ Angka: int  11 12 13 14 15
 $ Huruf: Factor w/ 5 levels "F","G","H","I",..: 1 2 3 4 5
```

1.3 Data Frame Management

```
data(iris)
```

```
head(iris)
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
```

```
6          5.4         3.9          1.7         0.4  setosa
```

```r
tail(iris)
```

```
    Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
145          6.7         3.3          5.7         2.5 virginica
146          6.7         3.0          5.2         2.3 virginica
147          6.3         2.5          5.0         1.9 virginica
148          6.5         3.0          5.2         2.0 virginica
149          6.2         3.4          5.4         2.3 virginica
150          5.9         3.0          5.1         1.8 virginica
```

```r
str(iris)
```

```
'data.frame':	150 obs. of  5 variables:
 $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

1.3.1 R Package

```r
# install.packages("readxl") - code to install R package
library(readxl)
```

```r
#install.packages("dplyr")
library(dplyr)
```

```

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

1.3.2 Data Management With dplyr

```r
head(iris)
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
```

```
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

```
irisbaru  <- mutate(iris, sepal2 = Sepal.Length + Sepal.Width)
```

```
head(irisbaru)
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species sepal2
1          5.1         3.5          1.4         0.2  setosa    8.6
2          4.9         3.0          1.4         0.2  setosa    7.9
3          4.7         3.2          1.3         0.2  setosa    7.9
4          4.6         3.1          1.5         0.2  setosa    7.7
5          5.0         3.6          1.4         0.2  setosa    8.6
6          5.4         3.9          1.7         0.4  setosa    9.3
```

```
irisetosa <- filter(iris, Species=="setosa")
head(irisetosa)
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

```
levels(iris$Species)
```

```
[1] "setosa"     "versicolor" "virginica"
```

```
irisversicolor <- filter(iris, Species=="setosa"& Petal.Length==1.3)
head(irisversicolor)
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          4.7         3.2          1.3         0.2  setosa
2          5.4         3.9          1.3         0.4  setosa
3          5.5         3.5          1.3         0.2  setosa
4          4.4         3.0          1.3         0.2  setosa
5          5.0         3.5          1.3         0.3  setosa
6          4.5         2.3          1.3         0.3  setosa
```

```
iris3 <- select(iris, Sepal.Length, Species)
head(iris3)
```

```
  Sepal.Length Species
1          5.1  setosa
2          4.9  setosa
3          4.7  setosa
4          4.6  setosa
```

```
5          5.0  setosa
6          5.4  setosa
```

```r
iris4 <- arrange(iris, Petal.Width)
head(iris4)
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          4.9         3.1          1.5         0.1  setosa
2          4.8         3.0          1.4         0.1  setosa
3          4.3         3.0          1.1         0.1  setosa
4          5.2         4.1          1.5         0.1  setosa
5          4.9         3.6          1.4         0.1  setosa
6          5.1         3.5          1.4         0.2  setosa
```

```r
iris4 <- arrange(iris, Species, desc(Petal.Width))
head(iris4)
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.0         3.5          1.6         0.6  setosa
2          5.1         3.3          1.7         0.5  setosa
3          5.4         3.9          1.7         0.4  setosa
4          5.7         4.4          1.5         0.4  setosa
5          5.4         3.9          1.3         0.4  setosa
6          5.1         3.7          1.5         0.4  setosa
```

```r
names(iris4)[1] <- "length"
head(iris4)
```

```
  length Sepal.Width Petal.Length Petal.Width Species
1    5.0         3.5          1.6         0.6  setosa
2    5.1         3.3          1.7         0.5  setosa
3    5.4         3.9          1.7         0.4  setosa
4    5.7         4.4          1.5         0.4  setosa
5    5.4         3.9          1.3         0.4  setosa
6    5.1         3.7          1.5         0.4  setosa
```

```r
head(iris4[,c(-1,-3)])
```

```
  Sepal.Width Petal.Width Species
1         3.5         0.6  setosa
2         3.3         0.5  setosa
3         3.9         0.4  setosa
4         4.4         0.4  setosa
5         3.9         0.4  setosa
6         3.7         0.4  setosa
```

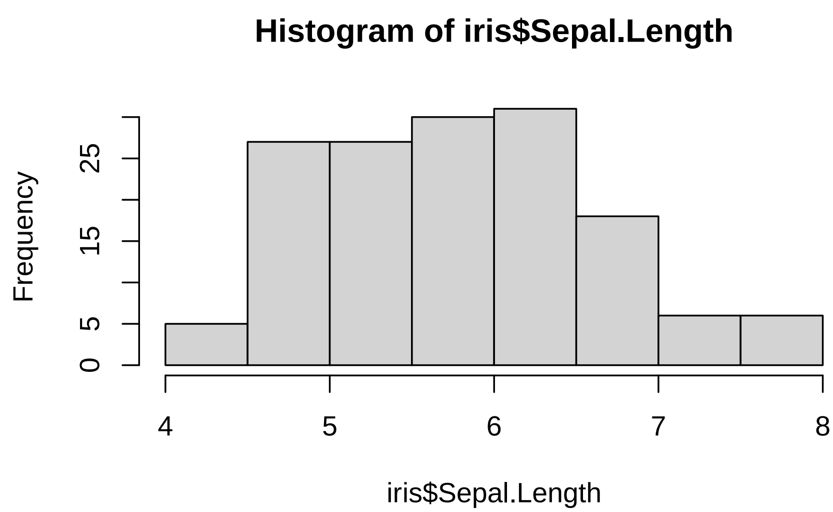
```r
iris %>% group_by(Species) %>% summarise(rata2_Sepal.Width = mean(Sepal.Width))
```

```
# A tibble: 3 x 2
  Species    rata2_Sepal.Width
```

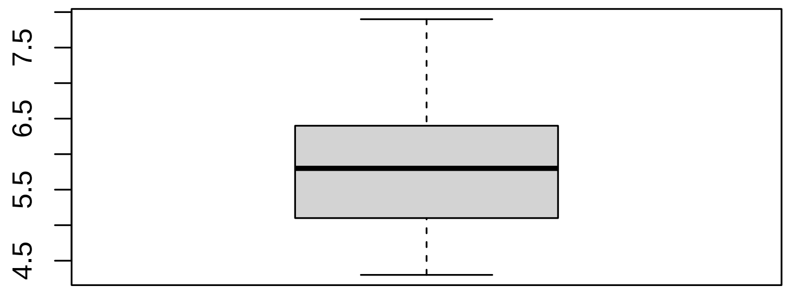
```
  <fct>                    <dbl>
1 setosa                    3.43
2 versicolor                2.77
3 virginica                 2.97
```

1.4 Visualization

```
hist(iris$Sepal.Length)
```




```
boxplot(iris$Sepal.Length)
```

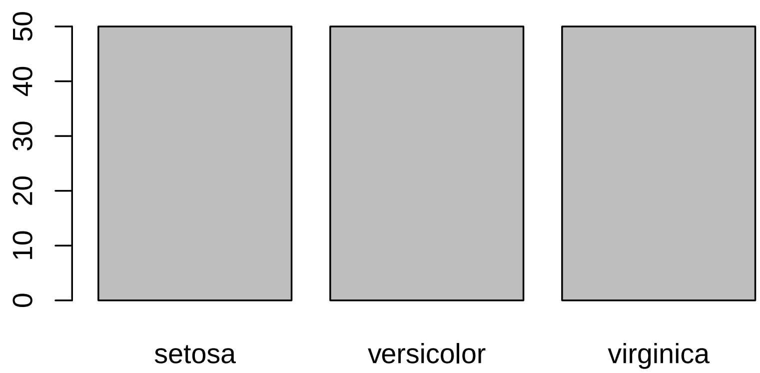



```
table(iris$Species)
```

1.4.3 Barplot

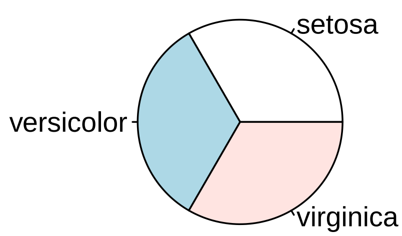
```
    setosa versicolor  virginica 
        50         50         50 
```

```
barplot(table(iris$Species))
```



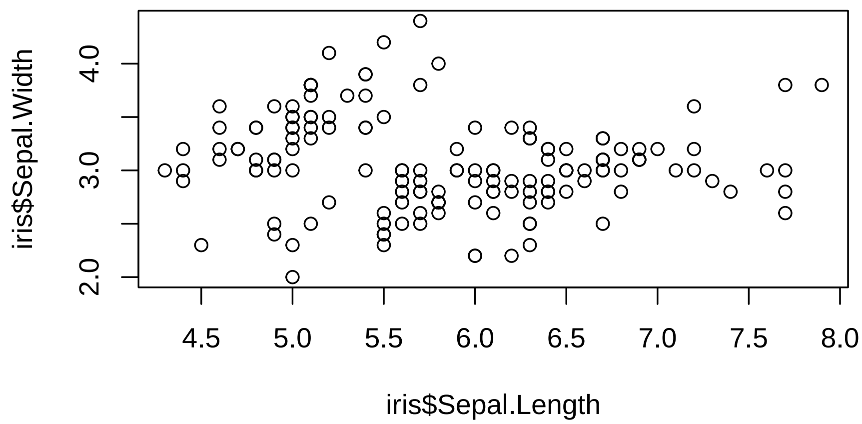
1.4.4 Pie Chart

```
pie(table(iris$Species))
```

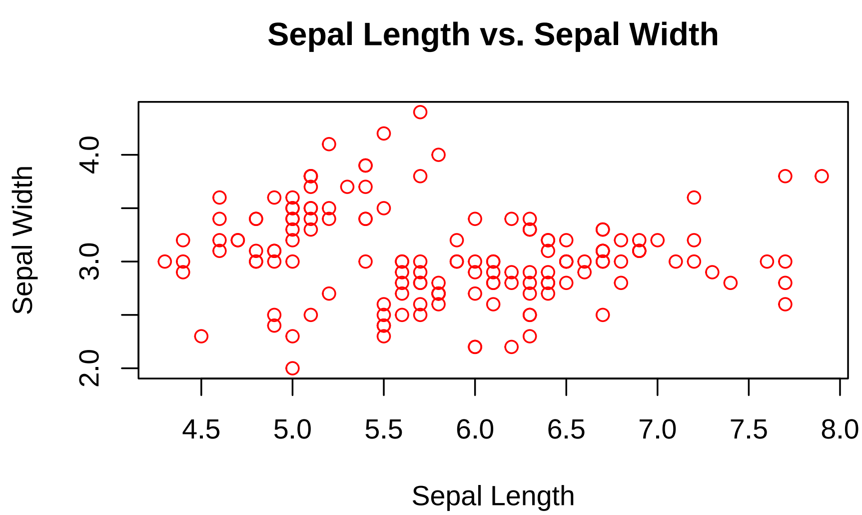


1.4.5 Scatter Plot

```
plot(iris$Sepal.Length,iris$Sepal.Width)
```




```
plot(iris$Sepal.Length, iris$Sepal.Width, main = "Sepal Length vs. Sepal Width",
     xlab = "Sepal Length", ylab = "Sepal Width", col = "red")
```



Chapter 2

Basic Linear Regression

2.1 EDA

```
library(readxl)
data1 <- read_excel("Data/Bab 2/data1.xlsx")
head(data1)
```

```
# A tibble: 6 x 11
  Time                `Outstanding KPR (miliar)` LnKPR `PDB Nominal` LnPDB
  <dttm>                                   <dbl> <dbl>         <dbl> <dbl>
1 2014-10-01 00:00:00                     38047.  10.5       723518.  13.5
2 2014-11-01 00:00:00                     38526.  10.6       719996.  13.5
3 2014-12-01 00:00:00                     39221.  10.6       718040.  13.5
4 2015-01-01 00:00:00                     39023.  10.6       715580.  13.5
5 2015-02-01 00:00:00                     39756.  10.6       718307.  13.5
6 2015-03-01 00:00:00                     39954.  10.6       724152.  13.5
# i 6 more variables: `Growth PDB YtY` <dbl>, Inflasi <dbl>, CreditRate <dbl>,
#   JII <dbl>, LnJII <dbl>, DFTV <dbl>
```

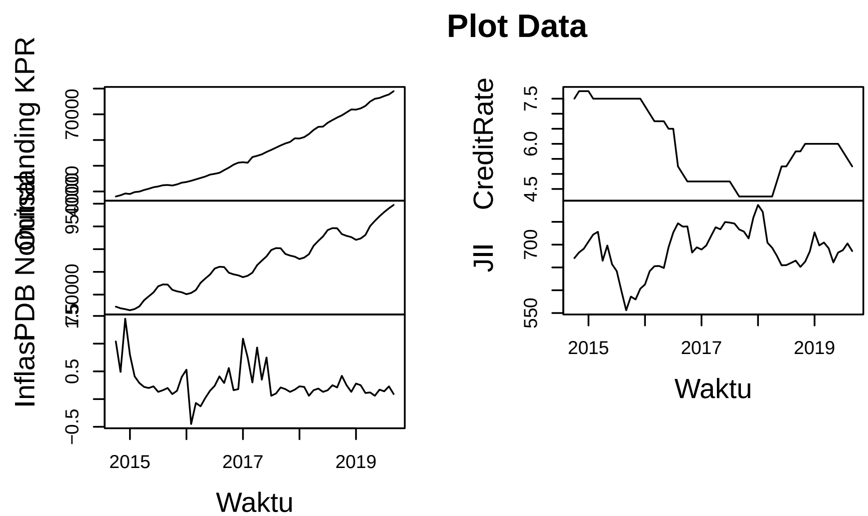
```
names(data1)[2] <- "Outstanding KPR"
names(data1)[7] <- "Inflasi"
names(data1)[8] <- "CreditRate"
```

```
data1baru = data1[,c(2,4,7,8,9)]
tsData = ts(data1baru, start=c(2014,10), frequency=12)
head(tsData,5)
```

```
     Outstanding KPR PDB Nominal Inflasi CreditRate    JII
[1,]        38047.46    723517.8    1.04       7.50 670.44
[2,]        38525.78    719995.7    0.49       7.75 683.02
[3,]        39220.50    718039.5    1.45       7.75 691.04
```

```
[4,]         39022.75     715580.1     0.80        7.75 706.68
[5,]         39755.80     718307.4     0.41        7.50 722.10
```

```
# Exploration
plot(tsData, type="l", main="Plot Data", xlab="Waktu")
```




```
# correlation
round(cor(tsData),3)
```

```
                Outstanding KPR PDB Nominal Inflasi CreditRate    JII
Outstanding KPR           1.000       0.968  -0.267     -0.548  0.218
PDB Nominal               0.968       1.000  -0.315     -0.590  0.204
Inflasi                  -0.267      -0.315   1.000      0.074  0.179
CreditRate               -0.548      -0.590   0.074      1.000 -0.624
JII                       0.218       0.204   0.179     -0.624  1.000
```

```
# Descriptive
summary(tsData)
```

```
 Outstanding KPR  PDB Nominal        Inflasi          CreditRate
 Min.   :38047   Min.   :715580   Min.   :-0.4500   Min.   :4.250
 1st Qu.:43622   1st Qu.:766385   1st Qu.: 0.1300   1st Qu.:4.750
 Median :53612   Median :812954   Median : 0.2050   Median :5.875
 Mean   :55303   Mean   :818266   Mean   : 0.2838   Mean   :5.896
 3rd Qu.:65595   3rd Qu.:871216   3rd Qu.: 0.3125   3rd Qu.:7.312
 Max.   :78998   Max.   :947281   Max.   : 1.4500   Max.   :7.750
      JII
 Min.   :556.1
 1st Qu.:659.2
```

```
 Median :691.5
 Mean   :688.1
 3rd Qu.:726.7
 Max.   :787.1
```

2.2 Ordinary Least Square (OLS)

```
# OLS
regresi1 = lm(LnKPR ~ LnPDB + Inflasi + CreditRate + LnJII + DFTV, data=data1)
summary(regresi1)
```

```
Call:
lm(formula = LnKPR ~ LnPDB + Inflasi + CreditRate + LnJII + DFTV,
    data = data1)

Residuals:
      Min        1Q    Median        3Q       Max
-0.138338 -0.029350  0.004568  0.029305  0.073667

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -21.79826    2.19407  -9.935 8.61e-14 ***
LnPDB         2.29744    0.14887  15.432  < 2e-16 ***
Inflasi      -0.02407    0.02223  -1.083 0.283737
CreditRate    0.04397    0.01081   4.067 0.000156 ***
LnJII         0.16152    0.10803   1.495 0.140715
DFTV          0.18359    0.03372   5.444 1.31e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.04381 on 54 degrees of freedom
Multiple R-squared:  0.9659,    Adjusted R-squared:  0.9627
F-statistic: 305.6 on 5 and 54 DF,  p-value: < 2.2e-16
```

2.3 Diagnostic Gauss Markov

```
# Normality, Linearity, Heteroscedasticity
library(gvlma)
gvlma(regresi1)
```

```
Call:
lm(formula = LnKPR ~ LnPDB + Inflasi + CreditRate + LnJII + DFTV,
```

```
    data = data1)

Coefficients:
(Intercept)        LnPDB      Inflasi   CreditRate        LnJII         DFTV
  -21.79826      2.29744     -0.02407      0.04397      0.16152      0.18359


ASSESSMENT OF THE LINEAR MODEL ASSUMPTIONS
USING THE GLOBAL TEST ON 4 DEGREES-OF-FREEDOM:
Level of Significance =  0.05

Call:
 gvlma(x = regresi1)

                     Value p-value                Decision
Global Stat        7.04469 0.13355 Assumptions acceptable.
Skewness           3.60383 0.05765 Assumptions acceptable.
Kurtosis           0.86268 0.35299 Assumptions acceptable.
Link Function      2.52749 0.11188 Assumptions acceptable.
Heteroscedasticity 0.05069 0.82187 Assumptions acceptable.
```

```
# Heteroscedasticity Test
library(car)
```

```
Loading required package: carData
```

```
# White test
ncvTest(regresi1)
```

```
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 0.4905034, Df = 1, p = 0.4837
```

```
# Autocorrelation Test
library(lmtest)
```

```
Loading required package: zoo


Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric
```

```
dwtest(regresi1)
```

```

    Durbin-Watson test
```

```
data:  regresi1
DW = 0.58729, p-value = 9.399e-13
alternative hypothesis: true autocorrelation is greater than 0
```

```
# Assumption: No perfect multicollinearity
vif(regresi1)
```

```
     LnPDB    Inflasi CreditRate      LnJII       DFTV
  4.011533   1.412398   5.368989   1.828592   8.404443
```

```
# Re-estimate Standard Error
library(sandwich)
# Account for heteroskedasticity
coeftest(regresi1, vcov = vcovHC(regresi1, "HC1"))
```

```
t test of coefficients:

              Estimate Std. Error  t value  Pr(>|t|)    
(Intercept) -21.798262   1.581079 -13.7870 < 2.2e-16 ***
LnPDB         2.297444   0.118364  19.4100 < 2.2e-16 ***
Inflasi      -0.024070   0.019603  -1.2279    0.2248    
CreditRate    0.043970   0.009386   4.6846 1.937e-05 ***
LnJII         0.161515   0.099960   1.6158    0.1120    
DFTV          0.183586   0.028221   6.5052 2.615e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Account for heteroskedasticity and autocorrelation
coeftest(regresi1, vcov = vcovHAC)
```

```
t test of coefficients:

              Estimate Std. Error  t value  Pr(>|t|)    
(Intercept) -21.798262   1.500741 -14.5250 < 2.2e-16 ***
LnPDB         2.297444   0.134419  17.0917 < 2.2e-16 ***
Inflasi      -0.024070   0.020238  -1.1894  0.239504    
CreditRate    0.043970   0.016296   2.6981  0.009286 ** 
LnJII         0.161515   0.117986   1.3689  0.176686    
DFTV          0.183586   0.041167   4.4595 4.197e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

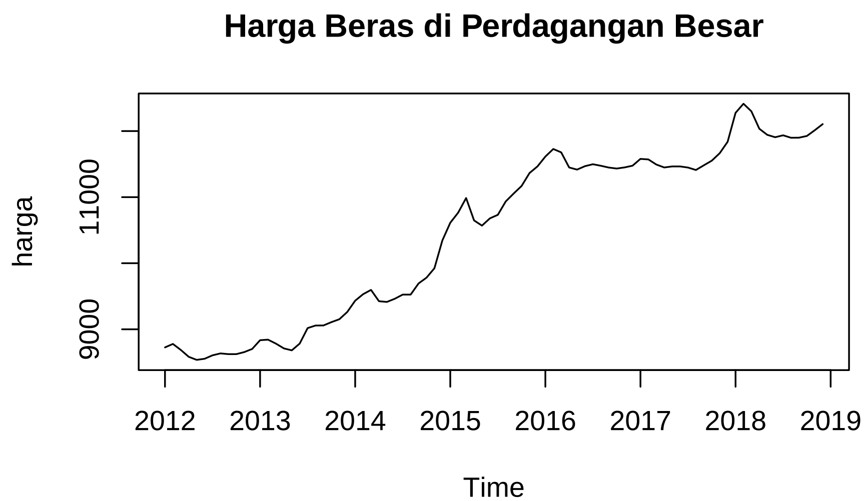
Chapter 3

Univariate Time Series

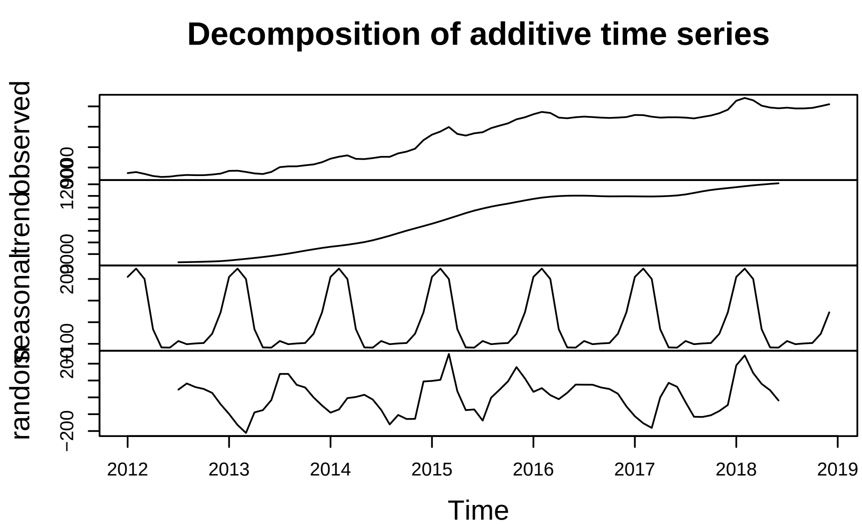
```
library(readxl)
hargaberas <- read_excel("Data/Bab 3/ARIMA.xlsx")
hargaberas = hargaberas[,c(-1)]
hargaberas = ts(hargaberas, start=c(2012,1), frequency=12)
hargaberas
```

```
       Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
2012  8726  8778  8687  8583  8537  8554  8606  8635  8624  8624  8655  8702
2013  8835  8843  8783  8711  8681  8784  9018  9057  9058  9108  9152  9262
2014  9433  9531  9596  9425  9414  9462  9525  9525  9694  9781  9924 10344
2015 10612 10766 10987 10648 10569 10679 10732 10935 11055 11169 11365 11465
2016 11614 11729 11678 11449 11417 11469 11498 11475 11448 11433 11450 11476
2017 11579 11571 11494 11449 11465 11465 11448 11411 11482 11552 11665 11838
2018 12276 12414 12299 12035 11943 11907 11936 11899 11900 11926 12013 12106
```

```
plot(hargaberas, main="Harga Beras di Perdagangan Besar")
```

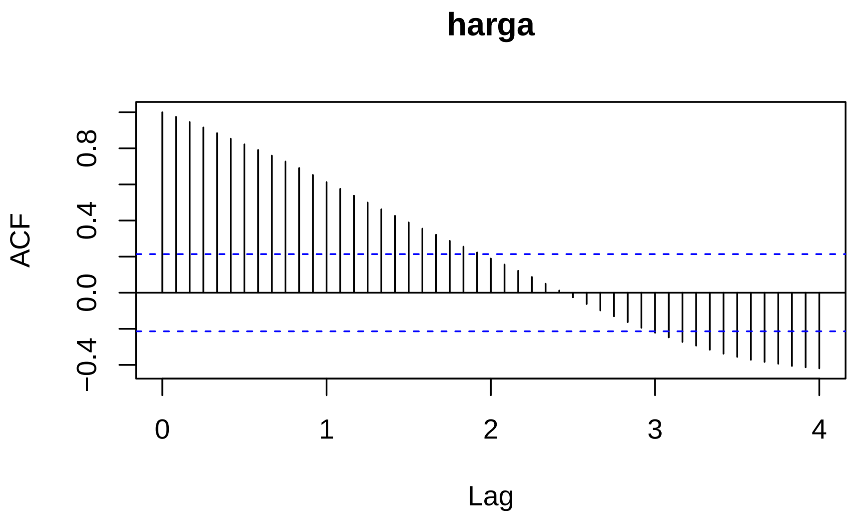



```r
dekomposisi = decompose(hargaberas)
plot(dekomposisi)
```

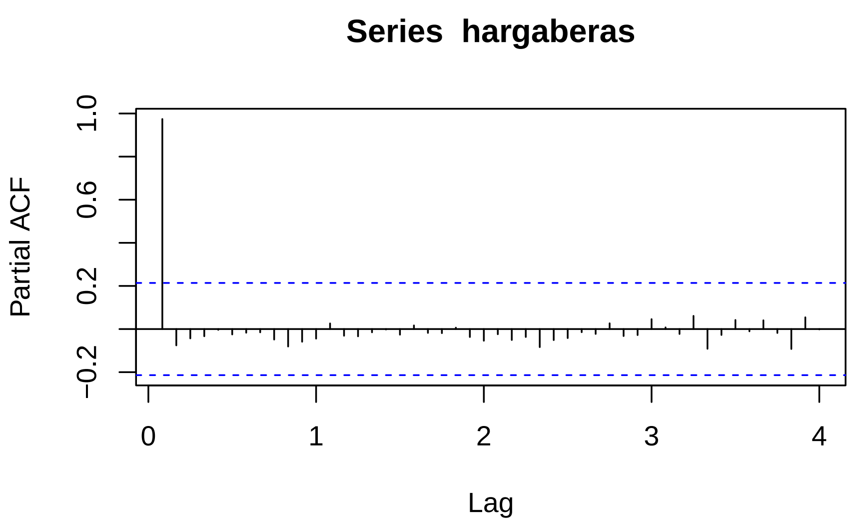


3.2 ACF and PACF Plot

```
# plot acf dan pacf
acf(hargaberas, lag=48)
```




```
pacf(hargaberas, lag=48)
```



##Stationary Test

```r
library(aTSA)
```

```
Attaching package: 'aTSA'

The following object is masked from 'package:graphics':

    identify
```

```r
# Augmented Dickey-Fuller Test
adf.test(hargaberas)
```

```
Augmented Dickey-Fuller Test
alternative: stationary

Type 1: no drift no trend
     lag  ADF p.value
[1,]   0 2.96   0.990
[2,]   1 1.65   0.975
[3,]   2 2.04   0.990
[4,]   3 2.18   0.990
Type 2: with drift no trend
     lag    ADF p.value
[1,]   0 -0.537   0.858
[2,]   1 -0.717   0.795
[3,]   2 -0.831   0.755
[4,]   3 -0.917   0.724
Type 3: with drift and trend
     lag   ADF p.value
[1,]   0 -1.42   0.814
[2,]   1 -2.56   0.340
[3,]   2 -2.00   0.570
[4,]   3 -1.78   0.663
----
Note: in fact, p.value = 0.01 means p.value <= 0.01
```

```r
# Firs Difference Form
adf.test(diff(hargaberas))
```

```
Augmented Dickey-Fuller Test
alternative: stationary

Type 1: no drift no trend
     lag   ADF p.value
[1,]   0 -5.11    0.01
[2,]   1 -5.02    0.01
[3,]   2 -4.42    0.01
```

```
[4,]   3 -4.36    0.01
Type 2: with drift no trend
     lag   ADF p.value
[1,]   0 -5.47    0.01
[2,]   1 -5.58    0.01
[3,]   2 -5.10    0.01
[4,]   3 -5.27    0.01
Type 3: with drift and trend
     lag   ADF p.value
[1,]   0 -5.43    0.01
[2,]   1 -5.55    0.01
[3,]   2 -5.10    0.01
[4,]   3 -5.28    0.01
----
Note: in fact, p.value = 0.01 means p.value <= 0.01
```

3.3 ARIMA

```
library(forecast)
```

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


Attaching package: 'forecast'

The following object is masked from 'package:aTSA':

    forecast
```

```
auto.arima(hargaberas, trace=TRUE)
```

```

 ARIMA(2,1,2)(1,1,1)[12]                    : Inf
 ARIMA(0,1,0)(0,1,0)[12]                    : 884.0934
 ARIMA(1,1,0)(1,1,0)[12]                    : 872.3993
 ARIMA(0,1,1)(0,1,1)[12]                    : Inf
 ARIMA(1,1,0)(0,1,0)[12]                    : 876.5573
 ARIMA(1,1,0)(2,1,0)[12]                    : 863.6643
 ARIMA(1,1,0)(2,1,1)[12]                    : Inf
 ARIMA(1,1,0)(1,1,1)[12]                    : Inf
 ARIMA(0,1,0)(2,1,0)[12]                    : 867.3178
 ARIMA(2,1,0)(2,1,0)[12]                    : 865.871
 ARIMA(1,1,1)(2,1,0)[12]                    : 865.1084
 ARIMA(0,1,1)(2,1,0)[12]                    : 862.9856
```

```
 ARIMA(0,1,1)(1,1,0)[12]                    : 871.9363
 ARIMA(0,1,1)(2,1,1)[12]                    : Inf
 ARIMA(0,1,1)(1,1,1)[12]                    : Inf
 ARIMA(0,1,2)(2,1,0)[12]                    : 861.8543
 ARIMA(0,1,2)(1,1,0)[12]                    : 872.2305
 ARIMA(0,1,2)(2,1,1)[12]                    : Inf
 ARIMA(0,1,2)(1,1,1)[12]                    : Inf
 ARIMA(1,1,2)(2,1,0)[12]                    : 867.0498
 ARIMA(0,1,3)(2,1,0)[12]                    : 854.8026
 ARIMA(0,1,3)(1,1,0)[12]                    : 865.2557
 ARIMA(0,1,3)(2,1,1)[12]                    : Inf
 ARIMA(0,1,3)(1,1,1)[12]                    : Inf
 ARIMA(1,1,3)(2,1,0)[12]                    : Inf
 ARIMA(0,1,4)(2,1,0)[12]                    : 857.2177
 ARIMA(1,1,4)(2,1,0)[12]                    : 859.7882

 Best model: ARIMA(0,1,3)(2,1,0)[12]
```

```
Series: hargaberas
ARIMA(0,1,3)(2,1,0)[12]

Coefficients:
         ma1     ma2     ma3     sar1     sar2
      0.3775  0.0028  0.4180  -0.4831  -0.4956
s.e.  0.1166  0.1301  0.1403   0.1231   0.1239

sigma^2 = 7757:  log likelihood = -420.75
AIC=853.49   AICc=854.8   BIC=867.07
```

```
library(lmtest)
```

```
Loading required package: zoo


Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric
```

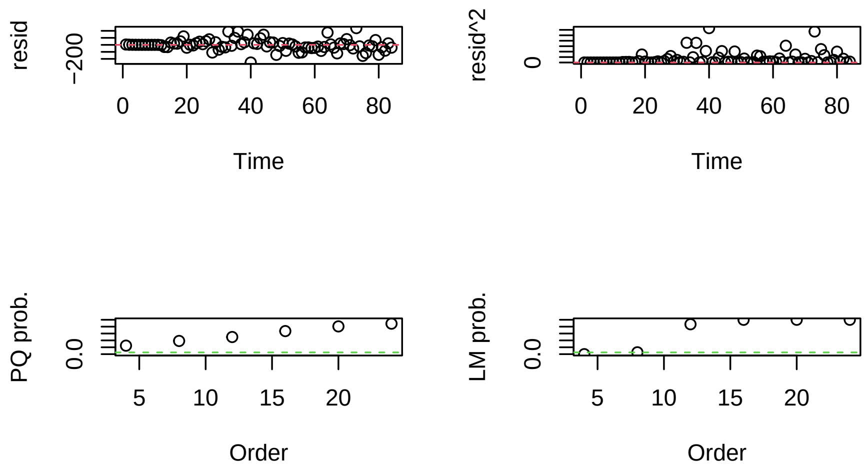
```
# Best model: ARIMA(0,1,3)(2,1,0)[12]
model1 = arima(hargaberas, order=c(0,1,3), seasonal=list(order=c(2,1,0), period=12))
coeftest(model1)
```

```
z test of coefficients:

      Estimate Std. Error z value  Pr(>|z|)
ma1   0.377526   0.116627  3.2370  0.001208 **
```

```
ma2   0.002799   0.130115  0.0215  0.982837
ma3   0.417985   0.140264  2.9800  0.002883 **
sar1 -0.483126   0.123055 -3.9261 8.634e-05 ***
sar2 -0.495630   0.123883 -4.0008 6.313e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Arch Test
arch.test(model1)
```

```
ARCH heteroscedasticity test for residuals
alternative: heteroscedastic

Portmanteau-Q test:
     order    PQ p.value
[1,]     4  5.41   0.247
[2,]     8  8.53   0.384
[3,]    12 11.36   0.498
[4,]    16 13.03   0.670
[5,]    20 14.44   0.808
[6,]    24 16.04   0.887
Lagrange-Multiplier test:
     order    LM  p.value
[1,]     4 41.16 6.04e-09
[2,]     8 13.84 5.42e-02
[3,]    12  6.06 8.69e-01
[4,]    16  3.39 9.99e-01
[5,]    20  2.16 1.00e+00
[6,]    24  1.16 1.00e+00
```



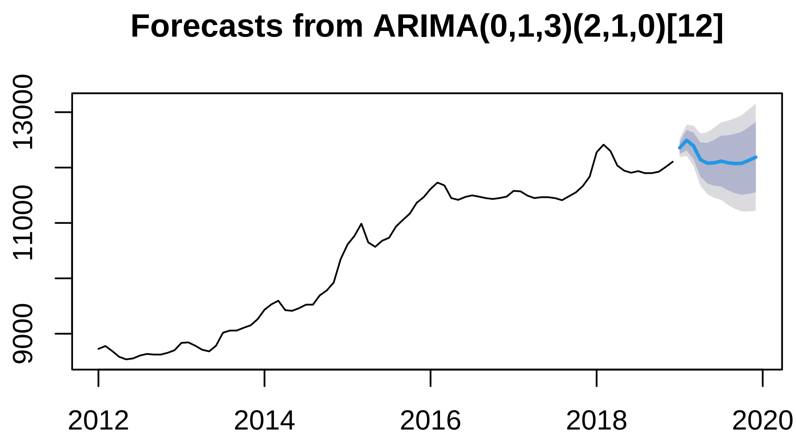

```
# Autocorrelartion Test
Box.test(model1$residuals, lag = 1, type = c("Ljung-Box"), fitdf = 0)
```

```
    Box-Ljung test

data:  model1$residuals
X-squared = 0.0041444, df = 1, p-value = 0.9487
```

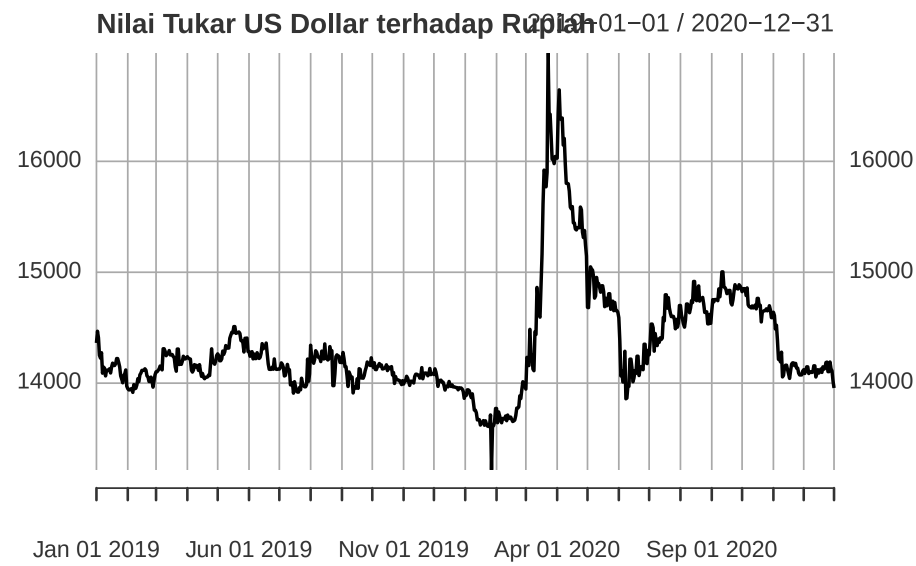
```
# Forecasting
forecast(model1, h=12)
```

```
         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
Jan 2019       12355.67 12246.95 12464.39 12189.39 12521.95
Feb 2019       12493.09 12308.02 12678.16 12210.05 12776.13
Mar 2019       12392.80 12154.53 12631.07 12028.40 12757.21
Apr 2019       12143.41 11835.19 12451.63 11672.03 12614.80
May 2019       12079.80 11714.79 12444.80 11521.57 12638.02
Jun 2019       12086.96 11672.89 12501.04 11453.70 12720.23
Jul 2019       12116.54 11658.63 12574.45 11416.22 12816.85
Aug 2019       12086.48 11588.57 12584.38 11325.00 12847.96
Sep 2019       12072.73 11537.81 12607.64 11254.64 12890.81
Oct 2019       12077.85 11508.32 12647.38 11206.83 12948.88
Nov 2019       12129.83 11527.68 12731.99 11208.92 13050.75
Dec 2019       12188.63 11555.52 12821.73 11220.38 13156.88
```

```
plot(forecast(model1, h=12))
```




```
library(readxl)
kurs <- read_excel("Data/Bab 3/ARCH-GARCH.xlsx")
kurs = kurs[,c(-1)]
Dates = seq(as.Date("2019-01-01"), as.Date("2020-12-31"), "day")

library(xts)
kurs = xts(kurs, order.by = Dates)
plot(kurs, main="Nilai Tukar US Dollar terhadap Rupiah")
```




```
# ARIMA
auto.arima(kurs, trace=TRUE)
```

```
Fitting models using approximations to speed things up...

ARIMA(2,1,2) with drift         : 8738.664
ARIMA(0,1,0) with drift         : 8755.631
ARIMA(1,1,0) with drift         : 8745.702
ARIMA(0,1,1) with drift         : 8744.836
ARIMA(0,1,0)                    : 8753.644
ARIMA(1,1,2) with drift         : 8742.017
ARIMA(2,1,1) with drift         : 8748.12
ARIMA(3,1,2) with drift         : 8746.649
ARIMA(2,1,3) with drift         : 8740.697
ARIMA(1,1,1) with drift         : 8746.629
ARIMA(1,1,3) with drift         : 8743.813
ARIMA(3,1,1) with drift         : 8748.657
ARIMA(3,1,3) with drift         : 8751.34
ARIMA(2,1,2)                    : 8736.651
ARIMA(1,1,2)                    : 8740.011
ARIMA(2,1,1)                    : 8746.122
ARIMA(3,1,2)                    : 8744.611
ARIMA(2,1,3)                    : 8738.68
ARIMA(1,1,1)                    : 8744.657
ARIMA(1,1,3)                    : 8741.802
ARIMA(3,1,1)                    : 8746.636
```

```
 ARIMA(3,1,3)                    : 8749.317

 Now re-fitting the best model(s) without approximations...

 ARIMA(2,1,2)                    : 8739.165

 Best model: ARIMA(2,1,2)
```

```
Series: kurs
ARIMA(2,1,2)

Coefficients:
         ar1      ar2      ma1     ma2
      1.2304  -0.7997  -1.3560  0.8726
s.e.  0.0650   0.0506   0.0599  0.0425

sigma^2 = 9181:  log likelihood = -4364.54
AIC=8739.08   AICc=8739.16   BIC=8762.05
```

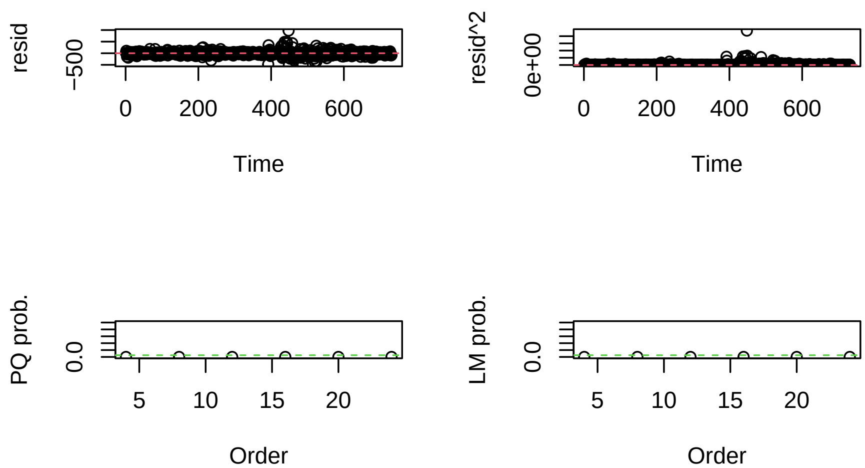
```
#  Best model: ARIMA(2,1,2)
model2 = arima(kurs, order=c(2,1,2))
coeftest(model2)
```

```
z test of coefficients:

     Estimate Std. Error z value  Pr(>|z|)
ar1  1.230421   0.065027  18.922 < 2.2e-16 ***
ar2 -0.799713   0.050577 -15.812 < 2.2e-16 ***
ma1 -1.355960   0.059905 -22.635 < 2.2e-16 ***
ma2  0.872641   0.042457  20.554 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# ARCH Test
arch.test(model2)
```

```
ARCH heteroscedasticity test for residuals
alternative: heteroscedastic

Portmanteau-Q test:
     order    PQ p.value
[1,]     4  70.7 1.6e-14
[2,]     8 134.2 0.0e+00
[3,]    12 188.4 0.0e+00
[4,]    16 206.3 0.0e+00
[5,]    20 216.8 0.0e+00
[6,]    24 225.0 0.0e+00
```

```
Lagrange-Multiplier test:
     order   LM  p.value
[1,]     4 1471 0.00e+00
[2,]     8  446 0.00e+00
[3,]    12  251 0.00e+00
[4,]    16  179 0.00e+00
[5,]    20  135 0.00e+00
[6,]    24  106 1.14e-12
```



if p.value <- 0.05 = ARCH/GARCH

```r
library(fGarch)
```

```
Warning: package 'fGarch' was built under R version 4.4.3

NOTE: Packages 'fBasics', 'timeDate', and 'timeSeries' are no longer
attached to the search() path when 'fGarch' is attached.

If needed attach them yourself in your R script by e.g.,
        require("timeSeries")
```

```r
# Stationary Test
# Phillips-Perron Unit Root Test
pp.test(kurs)
```

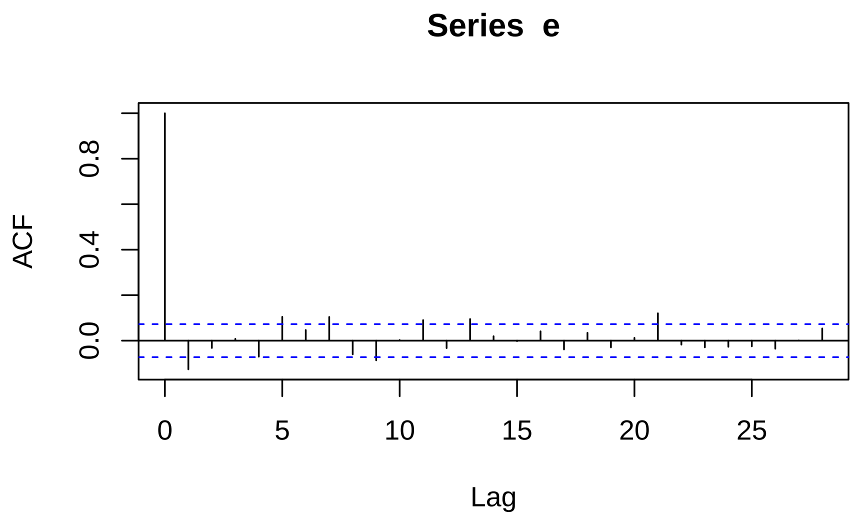
```
Phillips-Perron Unit Root Test
alternative: stationary

Type 1: no drift no trend
```

```
 lag   Z_rho p.value
   6 -0.0408   0.683
-----
 Type 2: with drift no trend
 lag Z_rho p.value
   6 -11.1   0.107
-----
 Type 3: with drift and trend
 lag Z_rho p.value
   6 -12.1   0.367
---------------
Note: p-value = 0.01 means p.value <= 0.01
```

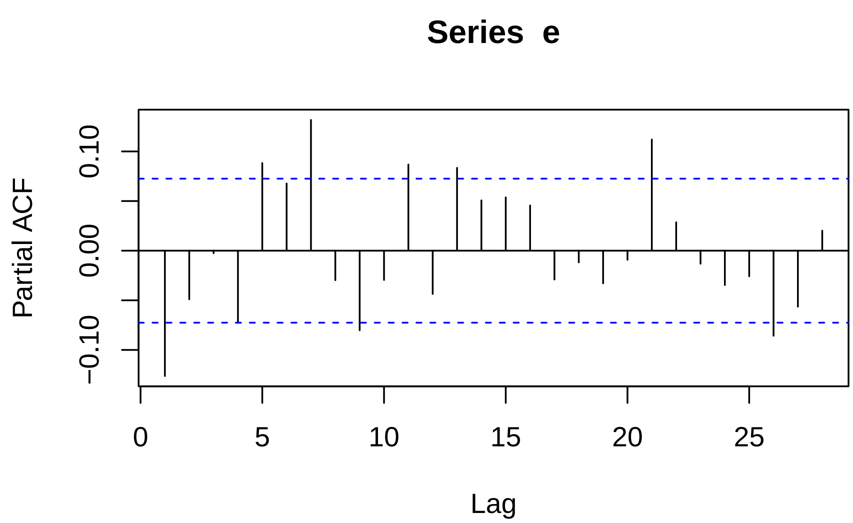
```
pp.test(diff(kurs))
```

```
Phillips-Perron Unit Root Test
alternative: stationary

Type 1: no drift no trend
 lag Z_rho p.value
   6  -804    0.01
-----
 Type 2: with drift no trend
 lag Z_rho p.value
   6  -804    0.01
-----
 Type 3: with drift and trend
 lag Z_rho p.value
   6  -804    0.01
---------------
Note: p-value = 0.01 means p.value <= 0.01
```

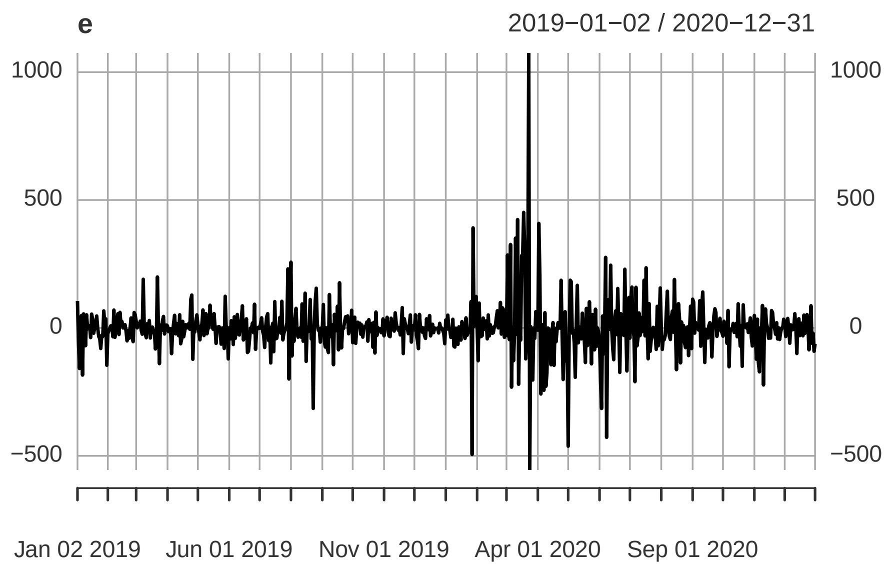
```
e = diff(kurs)[-1]
par(mfrow=c(1,1))
acf(e)
```




```r
pacf(e)
```




```r
plot(e)
```




```r
# ARCH(1) = GARCH(1,0)
model10 = garchFit(~garch(1,0), data=e, trace=FALSE)
summary(model10)
```

```
Title:
 GARCH Modelling

Call:
 garchFit(formula = ~garch(1, 0), data = e, trace = FALSE)

Mean and Variance Equation:
 data ~ garch(1, 0)
<environment: 0x000001e50ee15070>
 [data = e]

Conditional Distribution:
 norm

Coefficient(s):
        mu       omega      alpha1
  -1.15887  5442.67260     0.61303

Std. Errors:
 based on Hessian

Error Analysis:
```

```
        Estimate  Std. Error  t value Pr(>|t|)    
mu       -1.1589      2.8293   -0.410    0.682    
omega  5442.6726    374.5058   14.533  < 2e-16 ***
alpha1    0.6130      0.1073    5.713 1.11e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Log Likelihood:
 -4303.768    normalized:  -5.895573 

Description:
 Sat Jul  5 23:45:35 2025 by user: derik 


Standardised Residuals Tests:
                                  Statistic      p-Value
 Jarque-Bera Test   R    Chi^2  4839.704460 0.000000e+00
 Shapiro-Wilk Test  R    W         0.839958 0.000000e+00
 Ljung-Box Test     R    Q(10)    29.630084 9.844469e-04
 Ljung-Box Test     R    Q(15)    39.676117 5.074432e-04
 Ljung-Box Test     R    Q(20)    41.959542 2.799349e-03
 Ljung-Box Test     R^2  Q(10)    91.109411 3.219647e-15
 Ljung-Box Test     R^2  Q(15)   147.246211 0.000000e+00
 Ljung-Box Test     R^2  Q(20)   159.654019 0.000000e+00
 LM Arch Test       R    TR^2    114.581427 0.000000e+00

Information Criterion Statistics:
     AIC      BIC      SIC     HQIC 
11.79937 11.81824 11.79933 11.80665 
```

```r
# GARCH(1,1)
model11 = garchFit(~garch(1,1), data=e, trace=FALSE)
summary(model11)
```

```


Title:
 GARCH Modelling 

Call:
 garchFit(formula = ~garch(1, 1), data = e, trace = FALSE) 

Mean and Variance Equation:
 data ~ garch(1, 1)
<environment: 0x000001e4fd2919d8>
 [data = e]
```

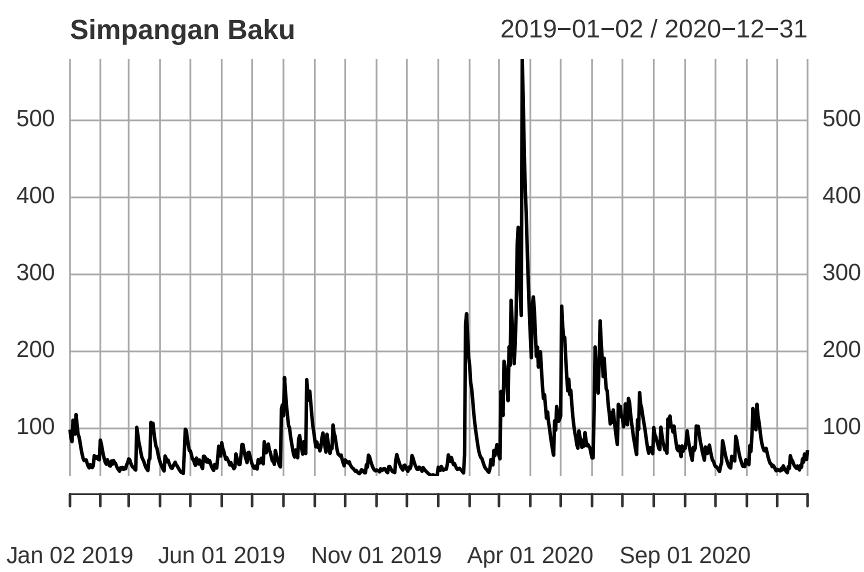
```
Conditional Distribution:
 norm

Coefficient(s):
       mu      omega     alpha1      beta1
 -2.41759  366.97243    0.25035    0.74326

Std. Errors:
 based on Hessian

Error Analysis:
        Estimate  Std. Error  t value Pr(>|t|)
mu      -2.41759     2.21121   -1.093    0.274
omega  366.97243    87.52212    4.193 2.75e-05 ***
alpha1   0.25035     0.04215    5.940 2.85e-09 ***
beta1    0.74326     0.03132   23.734  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Log Likelihood:
 -4188.199    normalized:  -5.73726

Description:
 Sat Jul  5 23:45:35 2025 by user: derik


Standardised Residuals Tests:
                                   Statistic     p-Value
 Jarque-Bera Test   R    Chi^2  1527.8454237 0.000000000
 Shapiro-Wilk Test  R    W         0.9121068 0.000000000
 Ljung-Box Test     R    Q(10)    28.3515452 0.001585409
 Ljung-Box Test     R    Q(15)    31.1437863 0.008403721
 Ljung-Box Test     R    Q(20)    32.4515827 0.038717818
 Ljung-Box Test     R^2  Q(10)     4.8877685 0.898547895
 Ljung-Box Test     R^2  Q(15)     9.1420867 0.869971676
 Ljung-Box Test     R^2  Q(20)    11.4700908 0.933109279
 LM Arch Test       R    TR^2      5.5858038 0.935507011

Information Criterion Statistics:
     AIC      BIC      SIC     HQIC
11.48548 11.51065 11.48542 11.49519
```

```r
# GARCH(1,1) with mean equation ARMA(0,1)
model11b = garchFit(~arma(0,1)+garch(1,1), data=e, trace=FALSE)
summary(model11b)
```

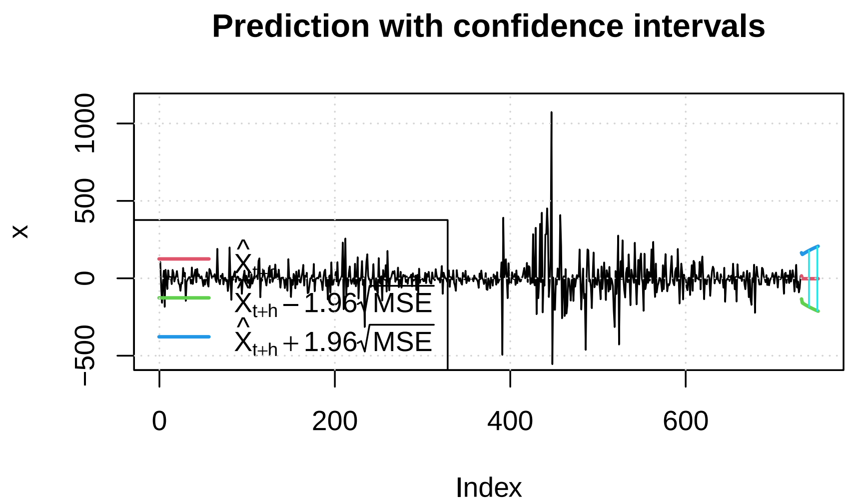
```
Title:
 GARCH Modelling

Call:
 garchFit(formula = ~arma(0, 1) + garch(1, 1), data = e, trace = FALSE)

Mean and Variance Equation:
 data ~ arma(0, 1) + garch(1, 1)
<environment: 0x000001e50486f900>
 [data = e]

Conditional Distribution:
 norm

Coefficient(s):
       mu        ma1      omega     alpha1      beta1
 -2.53982   -0.22537  337.43876    0.23716    0.75465

Std. Errors:
 based on Hessian

Error Analysis:
        Estimate  Std. Error  t value Pr(>|t|)
mu      -2.53982     1.74379   -1.456    0.145
ma1     -0.22537     0.04724   -4.771 1.83e-06 ***
omega  337.43876    85.21204    3.960 7.50e-05 ***
alpha1   0.23716     0.04038    5.874 4.26e-09 ***
beta1    0.75465     0.03129   24.118  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Log Likelihood:
 -4177.446    normalized:  -5.722529

Description:
 Sat Jul  5 23:45:35 2025 by user: derik


Standardised Residuals Tests:
                                  Statistic    p-Value
 Jarque-Bera Test   R    Chi^2  1331.0510650 0.00000000
 Shapiro-Wilk Test  R    W         0.9147019 0.00000000
 Ljung-Box Test     R    Q(10)    20.1208930 0.02812976
 Ljung-Box Test     R    Q(15)    23.9601552 0.06577322
 Ljung-Box Test     R    Q(20)    26.1490496 0.16094547
```

```
 Ljung-Box Test     R^2  Q(10)     4.7499032 0.90724638
 Ljung-Box Test     R^2  Q(15)     8.8134669 0.88706688
 Ljung-Box Test     R^2  Q(20)    11.6275315 0.92829976
 LM Arch Test       R    TR^2      5.4233826 0.94232413

Information Criterion Statistics:
     AIC      BIC      SIC     HQIC
11.45876 11.49021 11.45866 11.47089
```

```
# Best Model = "model11b"
Dates2 = seq(as.Date("2019-01-02"), as.Date("2020-12-31"), "day")
stdev = xts(model11b@sigma.t, order.by = Dates2)
plot(stdev, main="Simpangan Baku")
```




```
# Forecasting
predict(model11b, n.ahead=20, plot=TRUE, nx=731)
```



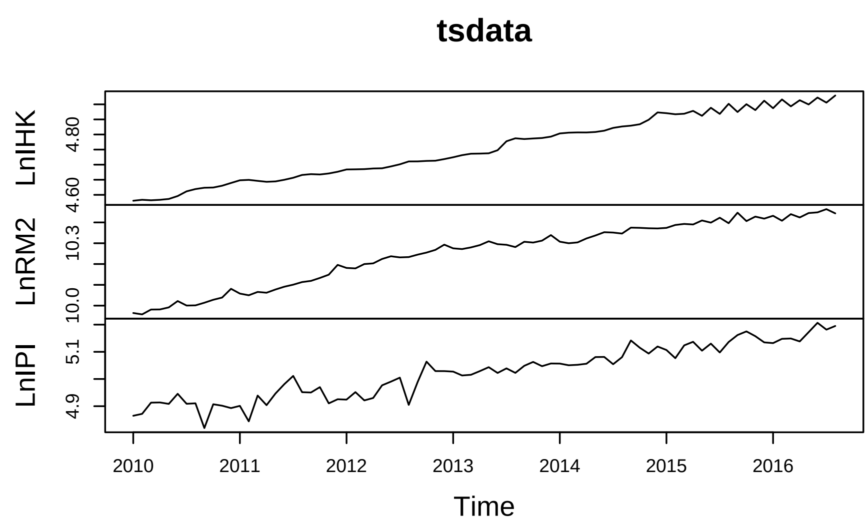
	meanForecast	meanError	standardDeviation	lowerInterval	upperInterval
1	15.748251	76.11087	76.11087	-133.4263	164.9228
2	-2.539819	79.85676	77.99279	-159.0562	153.9765
3	-2.539819	81.72802	79.81549	-162.7238	157.6442
4	-2.539819	83.54257	81.58306	-166.2802	161.2006
5	-2.539819	85.30415	83.29911	-169.7329	164.6532
6	-2.539819	87.01608	84.96689	-173.0882	168.0086
7	-2.539819	88.68137	86.58929	-176.3521	171.2725
8	-2.539819	90.30269	88.16892	-179.5298	174.4502
9	-2.539819	91.88248	89.70815	-182.6262	177.5465
10	-2.539819	93.42296	91.20912	-185.6454	180.5658
11	-2.539819	94.92613	92.67380	-188.5916	183.5120
12	-2.539819	96.39384	94.10397	-191.4683	186.3886
13	-2.539819	97.82780	95.50128	-194.2788	189.1991
14	-2.539819	99.22954	96.86724	-197.0261	191.9465
15	-2.539819	100.60052	98.20325	-199.7132	194.6336
16	-2.539819	101.94206	99.51061	-202.3426	197.2630
17	-2.539819	103.25541	100.79051	-204.9167	199.8371
18	-2.539819	104.54171	102.04409	-207.4378	202.3582
19	-2.539819	105.80203	103.27236	-209.9080	204.8283
20	-2.539819	107.03737	104.47632	-212.3292	207.2496

Chapter 4

Multivariate Time Series

4.1 VAR/VECM

```
library(readxl)
data1 <- read_excel("Data/Bab 4/VECM.xlsx")
tsdata = data1[,c(-1,-2,-3,-4)]
tsdata = ts(tsdata, start=c(2010,1), frequency=12)
plot(tsdata)
```




```
# Stationary Test
library(aTSA)
```

```
Attaching package: 'aTSA'

The following object is masked from 'package:graphics':

    identify
```

```r
adf.test(tsdata[,"LnIHK"])
```

```
Augmented Dickey-Fuller Test
alternative: stationary

Type 1: no drift no trend
     lag  ADF p.value
[1,]   0 3.25    0.99
[2,]   1 7.29    0.99
[3,]   2 3.75    0.99
[4,]   3 5.84    0.99
Type 2: with drift no trend
     lag    ADF p.value
[1,]   0 -0.506   0.869
[2,]   1 -0.541   0.856
[3,]   2 -0.727   0.791
[4,]   3 -0.914   0.725
Type 3: with drift and trend
     lag   ADF p.value
[1,]   0 -5.31  0.0100
[2,]   1 -2.02  0.5601
[3,]   2 -3.33  0.0732
[4,]   3 -1.77  0.6648
----
Note: in fact, p.value = 0.01 means p.value <= 0.01
```

```r
adf.test(tsdata[,"LnRM2"])
```

```
Augmented Dickey-Fuller Test
alternative: stationary

Type 1: no drift no trend
     lag  ADF p.value
[1,]   0 3.09    0.99
[2,]   1 4.89    0.99
[3,]   2 4.21    0.99
[4,]   3 4.52    0.99
Type 2: with drift no trend
     lag   ADF p.value
[1,]   0 -1.53   0.509
[2,]   1 -2.12   0.284
```

```
[3,]   2 -2.04   0.315
[4,]   3 -2.39   0.177
Type 3: with drift and trend
     lag   ADF p.value
[1,]   0 -2.31   0.441
[2,]   1 -1.50   0.776
[3,]   2 -1.36   0.836
[4,]   3 -1.09   0.918
----
Note: in fact, p.value = 0.01 means p.value <= 0.01
```

```
adf.test(tsdata[,"LnIPI"])
```

```
Augmented Dickey-Fuller Test
alternative: stationary

Type 1: no drift no trend
     lag  ADF p.value
[1,]   0 1.03   0.916
[2,]   1 1.45   0.961
[3,]   2 1.81   0.981
[4,]   3 2.09   0.990
Type 2: with drift no trend
     lag    ADF p.value
[1,]   0 -1.588   0.487
[2,]   1 -1.069   0.671
[3,]   2 -0.493   0.873
[4,]   3 -0.218   0.927
Type 3: with drift and trend
     lag   ADF p.value
[1,]   0 -6.68    0.01
[2,]   1 -5.35    0.01
[3,]   2 -4.47    0.01
[4,]   3 -4.11    0.01
----
Note: in fact, p.value = 0.01 means p.value <= 0.01
```

```
adf.test(diff(tsdata[,"LnIHK"]))
```

```
Augmented Dickey-Fuller Test
alternative: stationary

Type 1: no drift no trend
     lag    ADF p.value
[1,]   0 -15.36    0.01
[2,]   1  -3.52    0.01
[3,]   2  -4.20    0.01
```

```
[4,]   3  -2.61    0.01
Type 2: with drift no trend
     lag    ADF p.value
[1,]   0 -21.22    0.01
[2,]   1  -5.37    0.01
[3,]   2  -7.78    0.01
[4,]   3  -5.55    0.01
Type 3: with drift and trend
     lag    ADF p.value
[1,]   0 -21.09    0.01
[2,]   1  -5.36    0.01
[3,]   2  -7.79    0.01
[4,]   3  -5.59    0.01
----
Note: in fact, p.value = 0.01 means p.value <= 0.01
```

```
adf.test(diff(tsdata[,"LnRM2"]))
```

```
Augmented Dickey-Fuller Test
alternative: stationary

Type 1: no drift no trend
     lag    ADF p.value
[1,]   0 -11.73    0.01
[2,]   1  -5.61    0.01
[3,]   2  -4.49    0.01
[4,]   3  -3.31    0.01
Type 2: with drift no trend
     lag    ADF p.value
[1,]   0 -14.27    0.01
[2,]   1  -7.53    0.01
[3,]   2  -6.83    0.01
[4,]   3  -5.62    0.01
Type 3: with drift and trend
     lag    ADF p.value
[1,]   0 -14.60    0.01
[2,]   1  -7.85    0.01
[3,]   2  -7.34    0.01
[4,]   3  -6.33    0.01
----
Note: in fact, p.value = 0.01 means p.value <= 0.01
```

```
adf.test(diff(tsdata[,"LnIPI"]))
```

```
Augmented Dickey-Fuller Test
alternative: stationary
```

```
Type 1: no drift no trend
     lag    ADF p.value
[1,]   0 -12.46    0.01
[2,]   1  -9.27    0.01
[3,]   2  -7.24    0.01
[4,]   3  -5.75    0.01
Type 2: with drift no trend
     lag    ADF p.value
[1,]   0 -12.64    0.01
[2,]   1  -9.59    0.01
[3,]   2  -7.70    0.01
[4,]   3  -6.32    0.01
Type 3: with drift and trend
     lag    ADF p.value
[1,]   0 -12.56    0.01
[2,]   1  -9.54    0.01
[3,]   2  -7.67    0.01
[4,]   3  -6.31    0.01
----
Note: in fact, p.value = 0.01 means p.value <= 0.01
```

```r
library(urca)
library(vars)
```

```
Loading required package: MASS

Loading required package: strucchange

Loading required package: zoo


Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric

Loading required package: sandwich

Loading required package: lmtest


Attaching package: 'vars'

The following object is masked from 'package:aTSA':

    arch.test
```

```r
# Lag Optimum
VARselect(tsdata, lag.max = 10)
```

```
$selection
AIC(n)  HQ(n)  SC(n) FPE(n)
     9      2      2      9

$criteria
                  1             2             3             4             5
AIC(n) -2.454158e+01 -2.517704e+01 -2.527724e+01 -2.529876e+01 -2.531948e+01
HQ(n)  -2.438847e+01 -2.490910e+01 -2.489447e+01 -2.480116e+01 -2.470705e+01
SC(n)  -2.415612e+01 -2.450249e+01 -2.431360e+01 -2.404603e+01 -2.377765e+01
FPE(n)  2.197302e-11  1.165805e-11  1.058765e-11  1.043683e-11  1.034191e-11
                  6             7             8             9            10
AIC(n) -2.531117e+01 -2.509185e+01 -2.516992e+01 -2.557138e+01 -2.549790e+01
HQ(n)  -2.458391e+01 -2.424976e+01 -2.421299e+01 -2.449963e+01 -2.431131e+01
SC(n)  -2.348025e+01 -2.297184e+01 -2.276081e+01 -2.287319e+01 -2.251061e+01
FPE(n)  1.060910e-11  1.353419e-11  1.293440e-11  9.038835e-12  1.028471e-11
```

```
# Cointegration Test
cointest_eigen = ca.jo(tsdata, K=2, type="eigen", ecdet="const", spec="longrun")
summary(cointest_eigen)
```

```
######################
# Johansen-Procedure #
######################

Test type: maximal eigenvalue statistic (lambda max) , without linear trend and constan

Eigenvalues (lambda):
[1]  5.404113e-01  2.481996e-01  6.245216e-02 -2.633450e-15

Values of teststatistic and critical values of test:

          test 10pct  5pct  1pct
r <= 2 |  5.03  7.52  9.24 12.97
r <= 1 | 22.25 13.75 15.67 20.20
r = 0  | 60.64 19.77 22.00 26.81

Eigenvectors, normalised to first column:
(These are the cointegration relations)

           LnIHK.l2    LnRM2.l2   LnIPI.l2   constant
LnIHK.l2  1.0000000  1.00000000  1.0000000  1.0000000
LnRM2.l2 -0.1153114  0.08719842 -1.0510635 -0.6085127
LnIPI.l2 -0.8107822 -1.31468643  0.2406645  0.1913803
constant  0.4060887  0.96758544  4.8089173  0.5277719
```

```
Weights W:
(This is the loading matrix)

           LnIHK.l2    LnRM2.l2    LnIPI.l2      constant
LnIHK.d -0.07391103 -0.02494780 -0.03114995 -2.277816e-13
LnRM2.d -0.06399547  0.08952368  0.07223974  4.294637e-13
LnIPI.d -0.01046286  0.46857303 -0.03192575  8.303978e-13
```

```
# VECM
modelvecm = cajorls(cointest_eigen)
```

```
summary(modelvecm$rlm)
```

```
Response LnIHK.d :

Call:
lm(formula = LnIHK.d ~ ect1 + LnIHK.dl1 + LnRM2.dl1 + LnIPI.dl1 -
    1, data = data.mat)

Residuals:
      Min        1Q    Median        3Q       Max
-0.015021 -0.005668 -0.001157  0.003871  0.027827

Coefficients:
          Estimate Std. Error t value Pr(>|t|)
ect1      -0.07391    0.01199  -6.166 3.37e-08 ***
LnIHK.dl1 -0.70491    0.10168  -6.933 1.31e-09 ***
LnRM2.dl1  0.08470    0.06861   1.234    0.221
LnIPI.dl1 -0.01593    0.02695  -0.591    0.556
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.008005 on 74 degrees of freedom
Multiple R-squared:  0.6302,    Adjusted R-squared:  0.6102
F-statistic: 31.53 on 4 and 74 DF,  p-value: 2.518e-15


Response LnRM2.d :

Call:
lm(formula = LnRM2.d ~ ect1 + LnIHK.dl1 + LnRM2.dl1 + LnIPI.dl1 -
    1, data = data.mat)

Residuals:
      Min        1Q    Median        3Q       Max
-0.030921 -0.011339  0.001549  0.010970  0.044395
```

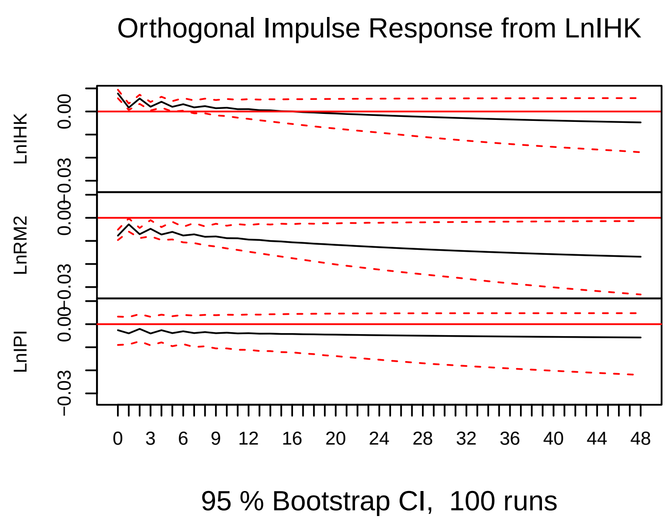
```
Coefficients:
          Estimate Std. Error t value Pr(>|t|)
ect1      -0.06400    0.02313  -2.767  0.00715 **
LnIHK.dl1  0.33519    0.19621   1.708  0.09177 .
LnRM2.dl1 -0.30552    0.13241  -2.307  0.02383 *
LnIPI.dl1  0.02765    0.05200   0.532  0.59650
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.01545 on 74 degrees of freedom
Multiple R-squared:  0.3241,    Adjusted R-squared:  0.2875
F-statistic:  8.87 on 4 and 74 DF,  p-value: 6.604e-06


Response LnIPI.d :

Call:
lm(formula = LnIPI.d ~ ect1 + LnIHK.dl1 + LnRM2.dl1 + LnIPI.dl1 -
    1, data = data.mat)

Residuals:
      Min        1Q    Median        3Q       Max
-0.094660 -0.011913  0.000434  0.021986  0.101991

Coefficients:
          Estimate Std. Error t value Pr(>|t|)
ect1      -0.01046    0.05068  -0.206  0.83700
LnIHK.dl1 -0.08499    0.42988  -0.198  0.84383
LnRM2.dl1  0.21190    0.29009   0.730  0.46742
LnIPI.dl1 -0.35319    0.11394  -3.100  0.00274 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.03385 on 74 degrees of freedom
Multiple R-squared:  0.1312,    Adjusted R-squared:  0.08427
F-statistic: 2.794 on 4 and 74 DF,  p-value: 0.03214
```

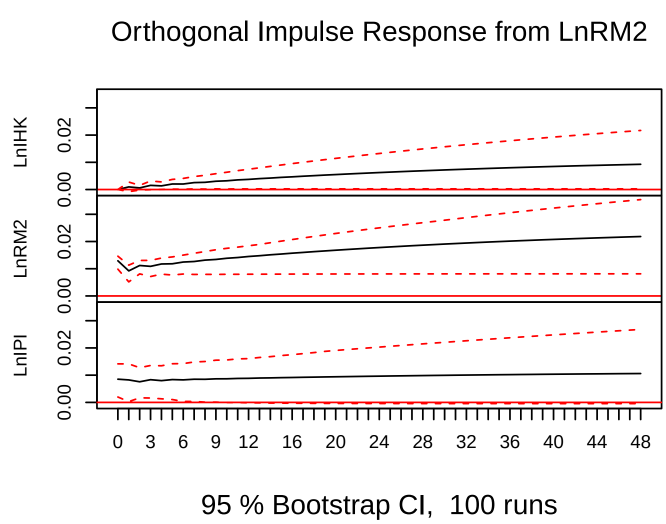
```r
modelvecm$beta
```

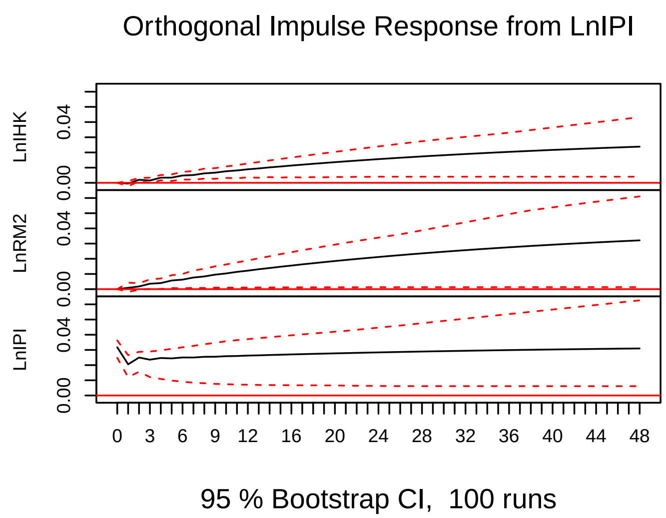
```
               ect1
LnIHK.l2  1.0000000
LnRM2.l2 -0.1153114
LnIPI.l2 -0.8107822
constant  0.4060887
```

```r
vecm = vec2var(cointest_eigen)
```

```
# IRF
ir = irf(vecm, n.ahead=48)
plot(ir)
```

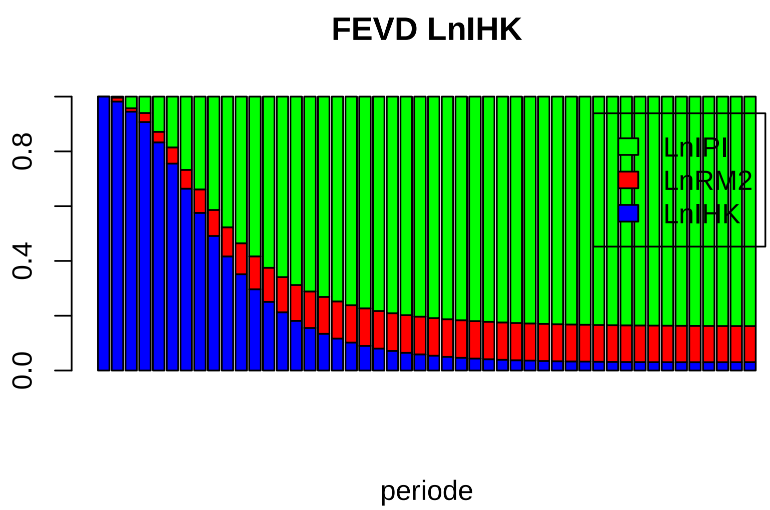




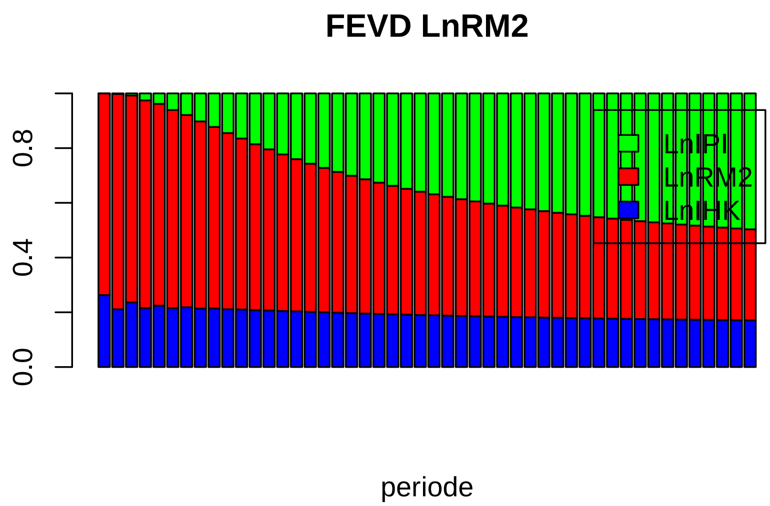



```
# FEVD
vd = fevd(vecm, n.ahead=48)
```

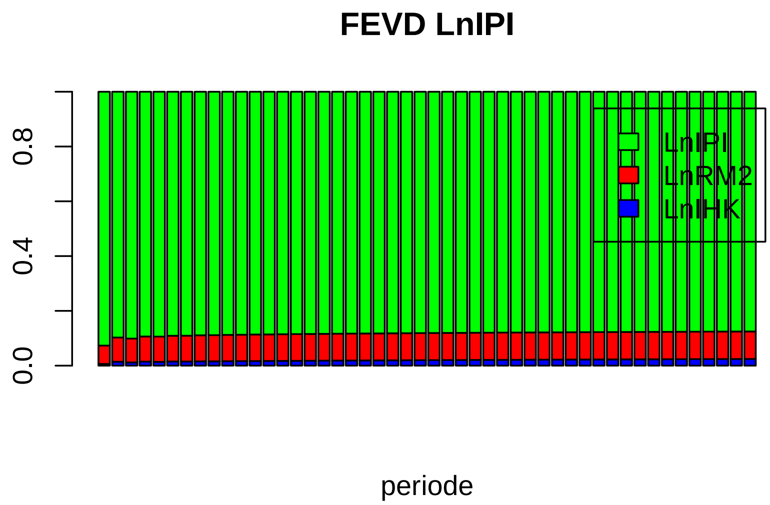
```
vd_LnIHK = as.matrix(vd$LnIHK)
barplot(t(vd_LnIHK), beside=FALSE, main="FEVD LnIHK", xlab="periode", col=c("blue","rec
```




```
vd_LnRM2 = as.matrix(vd$LnRM2)
barplot(t(vd_LnRM2), beside=FALSE, main="FEVD LnRM2", xlab="periode", col=c("blue","rec
```

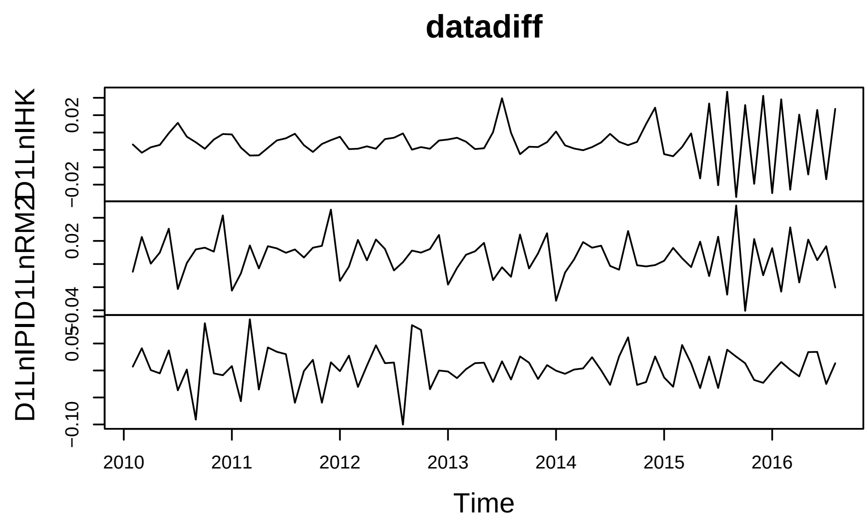



```
vd_LnIPI = as.matrix(vd$LnIPI)
barplot(t(vd_LnIPI), beside=FALSE, main="FEVD LnIPI", xlab="periode", col=c("blue","red","green")
```




```
# VAR FD, If No cointegtration
D1LnIHK = diff(tsdata[,"LnIHK"])
D1LnRM2 = diff(tsdata[,"LnRM2"])
D1LnIPI = diff(tsdata[,"LnIPI"])
datadiff = cbind(D1LnIHK, D1LnRM2, D1LnIPI)
```

```
plot(datadiff)
```




```
VARselect(datadiff, lag.max = 10)
```

```
$selection
AIC(n)  HQ(n)  SC(n) FPE(n)
    10      3      1      3

$criteria
                   1             2             3             4             5
AIC(n) -2.488624e+01 -2.508974e+01 -2.522710e+01 -2.513616e+01 -2.494060e+01
HQ(n)  -2.473209e+01 -2.481999e+01 -2.484173e+01 -2.463518e+01 -2.432401e+01
SC(n)  -2.449770e+01 -2.440980e+01 -2.425574e+01 -2.387340e+01 -2.338644e+01
FPE(n)  1.556733e-11  1.272259e-11  1.113498e-11  1.228686e-11  1.512269e-11
                   6             7             8             9            10
AIC(n) -2.490228e+01 -2.492141e+01 -2.513204e+01 -2.515552e+01 -2.527308e+01
HQ(n)  -2.417008e+01 -2.407360e+01 -2.416862e+01 -2.407649e+01 -2.407844e+01
SC(n)  -2.305671e+01 -2.278444e+01 -2.270366e+01 -2.243574e+01 -2.226189e+01
FPE(n)  1.599877e-11  1.609812e-11  1.349591e-11  1.379267e-11  1.300117e-11
```

```
varfd = VAR(datadiff, p=3, type="both")
summary(varfd)
```

```
VAR Estimation Results:
=========================
Endogenous variables: D1LnIHK, D1LnRM2, D1LnIPI
```

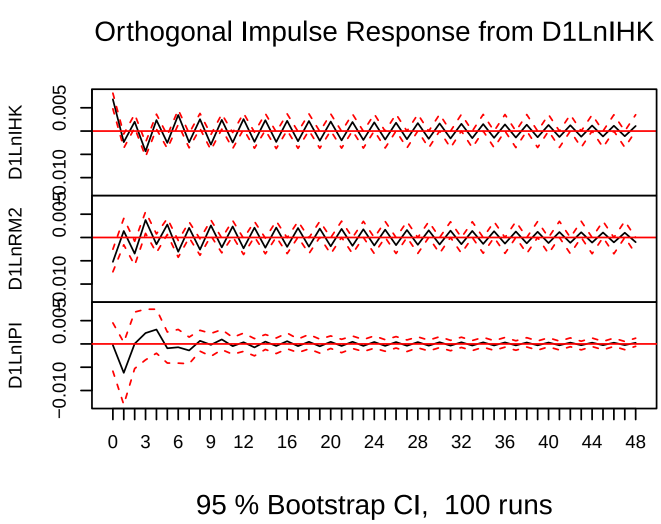
```
Deterministic variables: both
Sample size: 76
Log Likelihood: 669.253
Roots of the characteristic polynomial:
0.9777 0.7641 0.7641 0.6842 0.6842 0.6017 0.6017 0.5599 0.1487
Call:
VAR(y = datadiff, p = 3, type = "both")


Estimation results for equation D1LnIHK:
=========================================
D1LnIHK = D1LnIHK.l1 + D1LnRM2.l1 + D1LnIPI.l1 + D1LnIHK.l2 + D1LnRM2.l2 + D1LnIPI.l2 + D1LnIHK.l

             Estimate Std. Error t value Pr(>|t|)
D1LnIHK.l1 -2.695e-01  1.136e-01  -2.372  0.02065 *
D1LnRM2.l1  1.048e-01  6.640e-02   1.578  0.11951
D1LnIPI.l1 -2.316e-02  2.636e-02  -0.879  0.38286
D1LnIHK.l2  2.542e-01  1.214e-01   2.095  0.04010 *
D1LnRM2.l2  1.258e-01  6.760e-02   1.861  0.06733 .
D1LnIPI.l2  7.308e-03  2.795e-02   0.261  0.79454
D1LnIHK.l3 -3.575e-01  1.219e-01  -2.933  0.00463 **
D1LnRM2.l3  1.015e-01  6.679e-02   1.520  0.13348
D1LnIPI.l3  3.637e-02  2.675e-02   1.360  0.17860
const       3.701e-03  2.760e-03   1.341  0.18469
trend       3.603e-06  3.837e-05   0.094  0.92549
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


Residual standard error: 0.006803 on 65 degrees of freedom
Multiple R-Squared: 0.7325, Adjusted R-squared: 0.6913
F-statistic: 17.79 on 10 and 65 DF,  p-value: 4.614e-15


Estimation results for equation D1LnRM2:
=========================================
D1LnRM2 = D1LnIHK.l1 + D1LnRM2.l1 + D1LnIPI.l1 + D1LnIHK.l2 + D1LnRM2.l2 + D1LnIPI.l2 + D1LnIHK.l

             Estimate Std. Error t value Pr(>|t|)
D1LnIHK.l1 -1.023e-01  2.305e-01  -0.444 0.658605
D1LnRM2.l1 -4.005e-01  1.347e-01  -2.973 0.004138 **
D1LnIPI.l1 -4.924e-03  5.348e-02  -0.092 0.926915
D1LnIHK.l2 -7.711e-01  2.463e-01  -3.131 0.002611 **
D1LnRM2.l2 -3.981e-01  1.372e-01  -2.902 0.005054 **
D1LnIPI.l2  4.497e-02  5.671e-02   0.793 0.430615
D1LnIHK.l3  2.169e-01  2.473e-01   0.877 0.383719
```

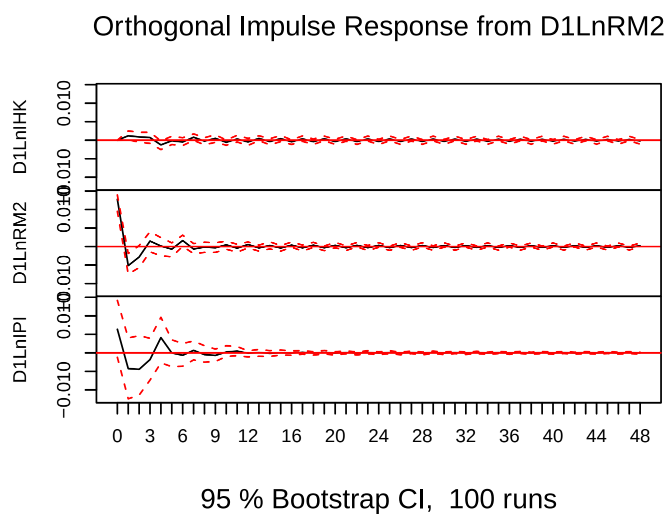
```
D1LnRM2.l3 -1.699e-02  1.355e-01  -0.125 0.900617
D1LnIPI.l3 -4.905e-02  5.427e-02  -0.904 0.369493
const       2.144e-02  5.601e-03   3.829 0.000292 ***
trend      -1.726e-04  7.786e-05  -2.217 0.030141 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


Residual standard error: 0.0138 on 65 degrees of freedom
Multiple R-Squared: 0.4561, Adjusted R-squared: 0.3724
F-statistic:  5.45 on 10 and 65 DF,  p-value: 7.948e-06


Estimation results for equation D1LnIPI:
========================================
D1LnIPI = D1LnIHK.l1 + D1LnRM2.l1 + D1LnIPI.l1 + D1LnIHK.l2 + D1LnRM2.l2 + D1LnIPI.l2 +

             Estimate Std. Error t value Pr(>|t|)
D1LnIHK.l1 -9.657e-01  5.265e-01  -1.834  0.07121 .
D1LnRM2.l1 -4.073e-02  3.078e-01  -0.132  0.89511
D1LnIPI.l1 -5.834e-01  1.221e-01  -4.776 1.06e-05 ***
D1LnIHK.l2 -1.065e+00  5.625e-01  -1.893  0.06284 .
D1LnRM2.l2 -2.528e-01  3.133e-01  -0.807  0.42264
D1LnIPI.l2 -4.311e-01  1.295e-01  -3.328  0.00144 **
D1LnIHK.l3 -3.630e-01  5.648e-01  -0.643  0.52271
D1LnRM2.l3 -3.290e-01  3.096e-01  -1.063  0.29188
D1LnIPI.l3 -2.070e-01  1.240e-01  -1.670  0.09981 .
const       2.268e-02  1.279e-02   1.773  0.08089 .
trend       4.216e-06  1.778e-04   0.024  0.98116
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


Residual standard error: 0.03153 on 65 degrees of freedom
Multiple R-Squared: 0.3186, Adjusted R-squared: 0.2138
F-statistic: 3.039 on 10 and 65 DF,  p-value: 0.003239



Covariance matrix of residuals:
          D1LnIHK    D1LnRM2    D1LnIPI
D1LnIHK  4.628e-05 -3.554e-05 -1.799e-06
D1LnRM2 -3.554e-05  1.906e-04  8.303e-05
D1LnIPI -1.799e-06  8.303e-05  9.941e-04

Correlation matrix of residuals:
```

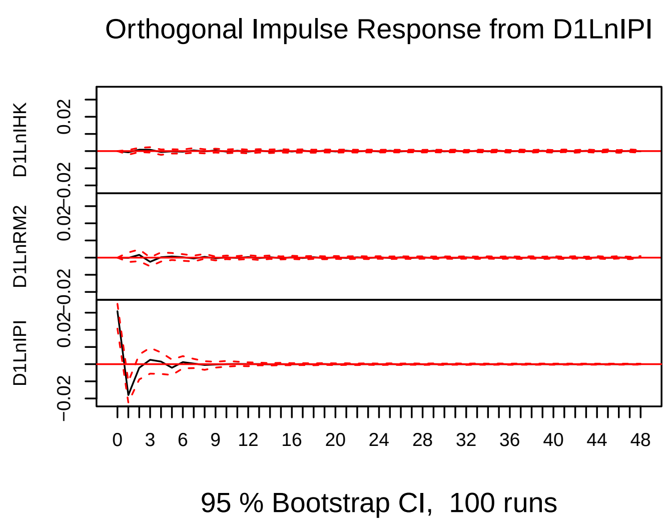
```
          D1LnIHK D1LnRM2   D1LnIPI
D1LnIHK  1.000000 -0.3785 -0.008387
D1LnRM2 -0.378457  1.0000  0.190775
D1LnIPI -0.008387  0.1908  1.000000
```

```
# Stablity VAR
# plot(stability(varfd))
```

```
# IRF
impres = irf(varfd, n.ahead=48)
plot(impres)
```







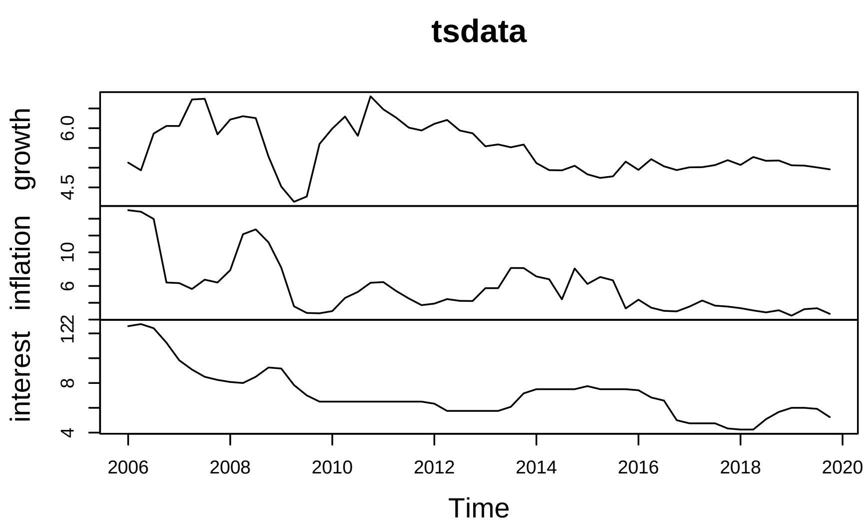
4.2 SVAR

```
library(readxl)
data1 <- read_excel("Data/Bab 4/SVAR.xlsx")
head(data1)
```

```
# A tibble: 6 x 4
```

```
  quarter growth inflation interest
  <chr>    <dbl>     <dbl>    <dbl>
1 2006q1    5.13     15.0      12.6
2 2006q2    4.93     14.8      12.8
3 2006q3    5.86     14.0      12.4
4 2006q4    6.06      6.41     11.2
5 2007q1    6.06      6.34      9.83
6 2007q2    6.73      5.64      9.08
```

```
tsdata = data1[,c(-1)]
tsdata = ts(tsdata, start=c(2006,1), frequency=4)
plot(tsdata)
```




```
library(aTSA)
adf.test(tsdata[,"growth"])
```

```
Augmented Dickey-Fuller Test
alternative: stationary

Type 1: no drift no trend
     lag    ADF p.value
[1,]   0 -0.315   0.550
[2,]   1 -0.265   0.564
[3,]   2 -0.578   0.469
[4,]   3 -0.583   0.467
Type 2: with drift no trend
     lag   ADF p.value
[1,]   0 -2.34  0.1978
```

```
[2,]   1 -2.68  0.0874
[3,]   2 -2.63  0.0955
[4,]   3 -2.25  0.2320
Type 3: with drift and trend
     lag   ADF p.value
[1,]   0 -3.06  0.1453
[2,]   1 -3.74  0.0296
[3,]   2 -3.40  0.0647
[4,]   3 -3.02  0.1627
----
Note: in fact, p.value = 0.01 means p.value <= 0.01
```

```
adf.test(tsdata[,"inflation"])
```

```
Augmented Dickey-Fuller Test
alternative: stationary

Type 1: no drift no trend
     lag   ADF p.value
[1,]   0 -2.34  0.0212
[2,]   1 -2.37  0.0200
[3,]   2 -2.31  0.0227
[4,]   3 -1.14  0.2675
Type 2: with drift no trend
     lag   ADF p.value
[1,]   0 -3.26  0.0236
[2,]   1 -3.82  0.0100
[3,]   2 -4.50  0.0100
[4,]   3 -2.37  0.1887
Type 3: with drift and trend
     lag   ADF p.value
[1,]   0 -3.46  0.0552
[2,]   1 -4.32  0.0100
[3,]   2 -5.67  0.0100
[4,]   3 -3.86  0.0221
----
Note: in fact, p.value = 0.01 means p.value <= 0.01
```

```
adf.test(tsdata[,"interest"])
```

```
Augmented Dickey-Fuller Test
alternative: stationary

Type 1: no drift no trend
     lag   ADF p.value
[1,]   0 -2.72  0.0100
[2,]   1 -2.02  0.0441
```

```
[3,]   2 -1.93  0.0532
[4,]   3 -1.67  0.0913
Type 2: with drift no trend
     lag   ADF p.value
[1,]   0 -3.06  0.0391
[2,]   1 -3.52  0.0126
[3,]   2 -3.33  0.0206
[4,]   3 -2.54  0.1232
Type 3: with drift and trend
     lag   ADF p.value
[1,]   0 -2.49  0.3714
[2,]   1 -4.07  0.0133
[3,]   2 -3.82  0.0240
[4,]   3 -2.90  0.2080
----
Note: in fact, p.value = 0.01 means p.value <= 0.01
```

```r
#cLag Optimum
library(vars)
VARselect(tsdata, lag.max = 10)
```

```
$selection
AIC(n)  HQ(n)  SC(n) FPE(n)
     2      1      1      2

$criteria
                 1           2           3           4           5           6
AIC(n) -3.70894477 -3.80592549 -3.69612636 -3.74782971 -3.74823364 -3.50174099
HQ(n)  -3.53024390 -3.49319896 -3.24937417 -3.16705186 -3.03343013 -2.65291182
SC(n)  -3.23190789 -2.97111094 -2.50353415 -2.19745983 -1.84008610 -1.23581578
FPE(n)  0.02453576  0.02239812  0.02535002  0.02471383  0.02580004  0.03528312
                 7           8           9          10
AIC(n) -3.37833948 -3.26151640 -3.71876734 -3.74360021
HQ(n)  -2.39548466 -2.14463592 -2.46786120 -2.35866842
SC(n)  -0.75463661 -0.28003587 -0.37950914 -0.04656435
FPE(n)  0.04400099  0.05681734  0.04372386  0.05614527
```

```r
# VAR Estimation
var.est1 = VAR(tsdata, p = 2, type = "none")
summary(var.est1)
```

```
VAR Estimation Results:
=========================
Endogenous variables: growth, inflation, interest
Deterministic variables: none
Sample size: 54
```

```
Log Likelihood: -128.363
Roots of the characteristic polynomial:
0.9909 0.706 0.706 0.4297 0.1321 0.1321
Call:
VAR(y = tsdata, p = 2, type = "none")


Estimation results for equation growth:
=======================================
growth = growth.l1 + inflation.l1 + interest.l1 + growth.l2 + inflation.l2 + interest.l

              Estimate Std. Error t value Pr(>|t|)
growth.l1     0.833907   0.134741   6.189 1.29e-07 ***
inflation.l1  0.001106   0.033450   0.033   0.9738
interest.l1   0.044345   0.143583   0.309   0.7588
growth.l2     0.035982   0.128753   0.279   0.7811
inflation.l2 -0.095836   0.038141  -2.513   0.0154 *
interest.l2   0.133608   0.114623   1.166   0.2495
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


Residual standard error: 0.3558 on 48 degrees of freedom
Multiple R-Squared: 0.9963, Adjusted R-squared: 0.9959
F-statistic:  2164 on 6 and 48 DF,  p-value: < 2.2e-16


Estimation results for equation inflation:
==========================================
inflation = growth.l1 + inflation.l1 + interest.l1 + growth.l2 + inflation.l2 + interes

              Estimate Std. Error t value Pr(>|t|)
growth.l1     0.232652   0.602805   0.386    0.701
inflation.l1  0.840256   0.149650   5.615 9.66e-07 ***
interest.l1  -0.061073   0.642360  -0.095    0.925
growth.l2     0.115551   0.576015   0.201    0.842
inflation.l2 -0.135143   0.170636  -0.792    0.432
interest.l2   0.002248   0.512801   0.004    0.997
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


Residual standard error: 1.592 on 48 degrees of freedom
Multiple R-Squared: 0.9381, Adjusted R-squared: 0.9304
F-statistic: 121.3 on 6 and 48 DF,  p-value: < 2.2e-16
```

```
Estimation results for equation interest:
=========================================
interest = growth.l1 + inflation.l1 + interest.l1 + growth.l2 + inflation.l2 + interest.l2

             Estimate Std. Error t value Pr(>|t|)    
growth.l1     0.20986    0.13119   1.600  0.11622    
inflation.l1  0.11010    0.03257   3.381  0.00144 ** 
interest.l1   1.12480    0.13979   8.046 1.88e-10 ***
growth.l2    -0.06797    0.12536  -0.542  0.59017    
inflation.l2 -0.03737    0.03713  -1.006  0.31930    
interest.l2  -0.30853    0.11160  -2.765  0.00806 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


Residual standard error: 0.3464 on 48 degrees of freedom
Multiple R-Squared: 0.9979, Adjusted R-squared: 0.9976 
F-statistic:  3734 on 6 and 48 DF,  p-value: < 2.2e-16 



Covariance matrix of residuals:
           growth inflation interest
growth    0.12655   0.08828  0.01781
inflation 0.08828   2.53321  0.18157
interest  0.01781   0.18157  0.11999

Correlation matrix of residuals:
          growth inflation interest
growth    1.0000    0.1559   0.1445
inflation 0.1559    1.0000   0.3293
interest  0.1445    0.3293   1.0000
```

```r
# Matriks A for SVAR AB-model
a.mat = diag(3)
diag(a.mat) = NA
a.mat[2,1] = NA
a.mat[3,1] = NA
a.mat[3,2] = NA
a.mat
```

```
     [,1] [,2] [,3]
[1,]   NA    0    0
[2,]   NA   NA    0
[3,]   NA   NA   NA
```

```
# Matriks B for SVAR AB-model
b.mat = diag(3)
diag(b.mat) = NA
b.mat
```

```
     [,1] [,2] [,3]
[1,]   NA    0    0
[2,]    0   NA    0
[3,]    0    0   NA
```

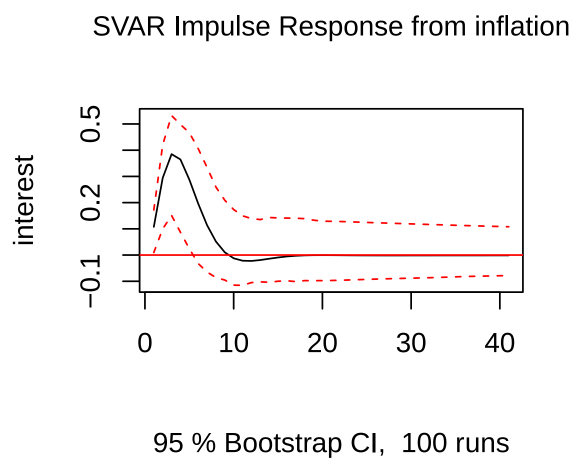
```
# Est SVAR
svar1 = SVAR(var.est1, Amat = a.mat, Bmat = b.mat, max.iter = 10000, hessian = TRUE)
```

```
Warning in SVAR(var.est1, Amat = a.mat, Bmat = b.mat, max.iter = 10000, : The
AB-model is just identified. No test possible.
```

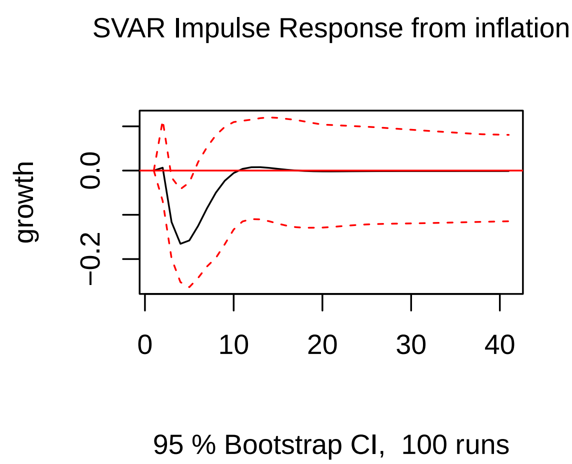
```
svar1
```

```
SVAR Estimation Results:
========================


Estimated A matrix:
            growth inflation interest
growth     1.00000   0.00000        0
inflation -0.69543   1.00000        0
interest  -0.09344  -0.06834        1

Estimated B matrix:
          growth inflation interest
growth    0.3558     0.000   0.0000
inflation 0.0000     1.573   0.0000
interest  0.0000     0.000   0.3255
```

```
# IRF
inf.int = irf(svar1, response = "interest", impulse = "inflation", n.ahead = 40)
plot(inf.int)
```

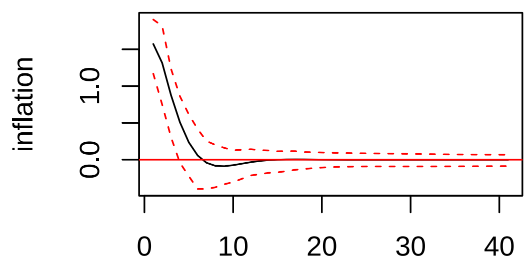



```
inf.gdp = irf(svar1, response = "growth", impulse = "inflation", n.ahead = 40)
plot(inf.gdp)
```




```
inf.inf = irf(svar1, response = "inflation", impulse = "inflation", n.ahead = 40)
plot(inf.inf)
```

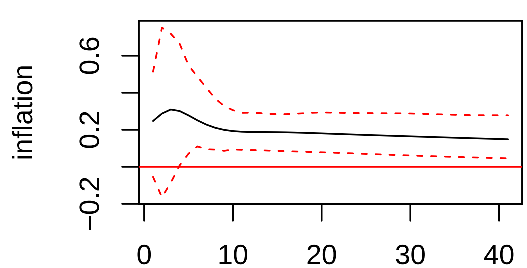
SVAR Impulse Response from inflation



95 % Bootstrap CI, 100 runs

```
gdp.inf = irf(svar1, response = "inflation", impulse = "growth", n.ahead = 40)
plot(gdp.inf)
```

SVAR Impulse Response from growth



95 % Bootstrap CI, 100 runs

```
#FEVD
vd = fevd(svar1, n.ahead=40)
```

```r
library(ARDL)
```

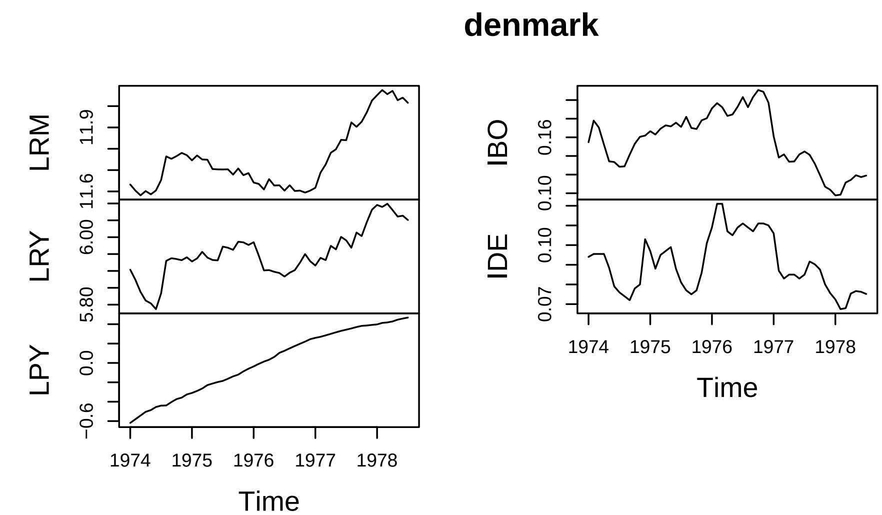
```
Warning: package 'ARDL' was built under R version 4.4.3
```

```
To cite the ARDL package in publications:

Use this reference to refer to the validity of the ARDL package.

  Natsiopoulos, Kleanthis, and Tzeremes, Nickolaos G. (2022). ARDL
  bounds test for cointegration: Replicating the Pesaran et al. (2001)
  results for the UK earnings equation using R. Journal of Applied
  Econometrics, 37(5), 1079-1090. https://doi.org/10.1002/jae.2919

Use this reference to cite this specific version of the ARDL package.

  Kleanthis Natsiopoulos and Nickolaos Tzeremes (2023). ARDL: ARDL, ECM
  and Bounds-Test for Cointegration. R package version 0.2.4.
  https://CRAN.R-project.org/package=ARDL
```

```r
# data sample
data(denmark)
denmark <- data.frame(denmark)
attach(denmark)
str(denmark)
```

```
'data.frame':	55 obs. of  5 variables:
 $ LRM: num  11.6 11.6 11.6 11.6 11.6 ...
 $ LRY: num  5.9 5.87 5.84 5.81 5.8 ...
 $ LPY: num  -0.619 -0.581 -0.543 -0.505 -0.486 ...
 $ IBO: num  0.155 0.178 0.171 0.152 0.134 ...
 $ IDE: num  0.094 0.0955 0.0955 0.0955 0.0885 0.079 0.076 0.074 0.072 0.078 ...
```

```r
denmark = ts(denmark, start=c(1974,1), frequency=12)
plot(denmark)
```




```r
library(tseries)
```

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


Attaching package: 'tseries'

The following objects are masked from 'package:aTSA':

    adf.test, kpss.test, pp.test
```

```r
pp.test(LRM) #Non-Stationary
```

```

    Phillips-Perron Unit Root Test

data:  LRM
Dickey-Fuller Z(alpha) = -3.2568, Truncation lag parameter = 3, p-value
= 0.9205
alternative hypothesis: stationary
```

```r
pp.test(LRY) #Non-Stationary
```

```

    Phillips-Perron Unit Root Test

data:  LRY
```

```
Dickey-Fuller Z(alpha) = -11.467, Truncation lag parameter = 3, p-value
= 0.428
alternative hypothesis: stationary
```

```
pp.test(IBO) #Non-Stationary
```

```

    Phillips-Perron Unit Root Test

data:  IBO
Dickey-Fuller Z(alpha) = -5.5494, Truncation lag parameter = 3, p-value
= 0.7882
alternative hypothesis: stationary
```

```
pp.test(IDE) #Non-Stationary
```

```

    Phillips-Perron Unit Root Test

data:  IDE
Dickey-Fuller Z(alpha) = -9.0346, Truncation lag parameter = 3, p-value
= 0.5761
alternative hypothesis: stationary
```

```
pp.test(diff(LRM)) #Stationary
```

```
Warning in pp.test(diff(LRM)): p-value smaller than printed p-value

    Phillips-Perron Unit Root Test

data:  diff(LRM)
Dickey-Fuller Z(alpha) = -59.819, Truncation lag parameter = 3, p-value
= 0.01
alternative hypothesis: stationary
```

```
pp.test(diff(LRY)) #Stationary
```

```
Warning in pp.test(diff(LRY)): p-value smaller than printed p-value

    Phillips-Perron Unit Root Test

data:  diff(LRY)
Dickey-Fuller Z(alpha) = -42.472, Truncation lag parameter = 3, p-value
= 0.01
alternative hypothesis: stationary
```

```
pp.test(diff(IBO)) #Stationary
```

```
Warning in pp.test(diff(IBO)): p-value smaller than printed p-value


    Phillips-Perron Unit Root Test

data:  diff(IBO)
Dickey-Fuller Z(alpha) = -38.898, Truncation lag parameter = 3, p-value
= 0.01
alternative hypothesis: stationary
```

```
pp.test(diff(IDE)) #Stationary
```

```
Warning in pp.test(diff(IDE)): p-value smaller than printed p-value


    Phillips-Perron Unit Root Test

data:  diff(IDE)
Dickey-Fuller Z(alpha) = -35.668, Truncation lag parameter = 3, p-value
= 0.01
alternative hypothesis: stationary
```

```
# ARDL Auto Search Optimum Lag
models <- auto_ardl(LRM ~ LRY + IBO + IDE, data = denmark, max_order = 5)
# The top 20 models according to the AIC
models$top_orders
```

```
   LRM LRY IBO IDE       AIC
1    3   1   3   2 -251.0259
2    3   1   3   3 -250.1144
3    2   2   0   0 -249.6266
4    3   2   3   2 -249.1087
5    3   2   3   3 -248.1858
6    2   2   0   1 -247.7786
7    2   1   0   0 -247.5643
8    2   2   1   1 -246.6885
9    3   3   3   3 -246.3061
10   2   2   1   2 -246.2709
11   2   1   1   1 -245.8736
12   2   2   2   2 -245.7722
13   1   1   0   0 -245.6620
14   2   1   2   2 -245.1712
15   3   1   2   2 -245.0996
16   1   0   0   0 -244.4317
17   1   1   0   1 -243.7702
18   5   5   5   5 -243.3120
19   4   1   3   2 -243.0728
20   4   1   3   3 -242.4378
```

```r
# The best model was found to be the ARDL(3,1,3,2)
ardl_3132 <- models$best_model
ardl_3132$order
```

```
LRM LRY IBO IDE
  3   1   3   2
```

```r
summary(ardl_3132)
```

```

Time series regression with "ts" data:
Start = 1974(4), End = 1978(7)

Call:
dynlm::dynlm(formula = full_formula, data = data, start = start,
    end = end)

Residuals:
      Min        1Q    Median        3Q       Max
-0.029939 -0.008856 -0.002562  0.008190  0.072577

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   2.6202     0.5678   4.615 4.19e-05 ***
L(LRM, 1)     0.3192     0.1367   2.336 0.024735 *
L(LRM, 2)     0.5326     0.1324   4.024 0.000255 ***
L(LRM, 3)    -0.2687     0.1021  -2.631 0.012143 *
LRY           0.6728     0.1312   5.129 8.32e-06 ***
L(LRY, 1)    -0.2574     0.1472  -1.749 0.088146 .
IBO          -1.0785     0.3217  -3.353 0.001790 **
L(IBO, 1)    -0.1062     0.5858  -0.181 0.857081
L(IBO, 2)     0.2877     0.5691   0.505 0.616067
L(IBO, 3)    -0.9947     0.3925  -2.534 0.015401 *
IDE           0.1255     0.5545   0.226 0.822161
L(IDE, 1)    -0.3280     0.7213  -0.455 0.651847
L(IDE, 2)     1.4079     0.5520   2.550 0.014803 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.0191 on 39 degrees of freedom
Multiple R-squared:  0.988, Adjusted R-squared:  0.9843
F-statistic: 266.8 on 12 and 39 DF,  p-value: < 2.2e-16
```

```r
library(lmtest)
bgtest(ardl_3132) # Autocorrelation Test
```

```
    Breusch-Godfrey test for serial correlation of order up to 1

data:  ardl_3132
LM test = 1.1192, df = 1, p-value = 0.2901
```

```
bptest(ardl_3132) # Heteroscedasticity Test
```

```

    studentized Breusch-Pagan test

data:  ardl_3132
BP = 4.4815, df = 12, p-value = 0.9731
```

```
# Cointegration Test
fbounds <- bounds_f_test(ardl_3132, case = 2, alpha = 0.05)
fbounds$tab
```

```
  statistic Lower-bound I(0) Upper-bound I(1) alpha     p.value
F  5.116768          2.77498          3.65953  0.05 0.004417563
```

```
# ARDL-ECM
uecm_3132 <- uecm(LRM ~ LRY + IBO + IDE, data = denmark, order = c(3,1,3,2))
summary(uecm_3132)
```

```

Time series regression with "ts" data:
Start = 1974(4), End = 1978(7)

Call:
dynlm::dynlm(formula = full_formula, data = data, start = start,
    end = end)

Residuals:
      Min        1Q    Median        3Q       Max
-0.029939 -0.008856 -0.002562  0.008190  0.072577

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   2.62019    0.56777   4.615 4.19e-05 ***
L(LRM, 1)    -0.41685    0.09166  -4.548 5.15e-05 ***
L(LRY, 1)     0.41538    0.11761   3.532  0.00108 **
L(IBO, 1)    -1.89172    0.39111  -4.837 2.09e-05 ***
L(IDE, 1)     1.20534    0.44690   2.697  0.01028 *
d(L(LRM, 1)) -0.26394    0.10192  -2.590  0.01343 *
d(L(LRM, 2))  0.26867    0.10213   2.631  0.01214 *
d(LRY)        0.67280    0.13116   5.129 8.32e-06 ***
d(IBO)       -1.07852    0.32170  -3.353  0.00179 **
d(L(IBO, 1))  0.70701    0.46874   1.508  0.13953
```

```
d(L(IBO, 2))  0.99468    0.39251   2.534  0.01540 *
d(IDE)        0.12546    0.55445   0.226  0.82216
d(L(IDE, 1)) -1.40786    0.55204  -2.550  0.01480 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.0191 on 39 degrees of freedom
Multiple R-squared:  0.7458,    Adjusted R-squared:  0.6676
F-statistic: 9.537 on 12 and 39 DF,  p-value: 3.001e-08
```

```
# ARDL-ECM 2
recm_3132 <- recm(uecm_3132, case = 2)
summary(recm_3132)
```

```

Time series regression with "zooreg" data:
Start = Apr 1974, End = Jul 1978

Call:
dynlm::dynlm(formula = full_formula, data = data, start = start,
    end = end)

Residuals:
      Min        1Q    Median        3Q       Max
-0.029939 -0.008856 -0.002562  0.008190  0.072577

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
d(L(LRM, 1)) -0.26394    0.09008  -2.930 0.005405 **
d(L(LRM, 2))  0.26867    0.09127   2.944 0.005214 **
d(LRY)        0.67280    0.11591   5.805 7.03e-07 ***
d(IBO)       -1.07852    0.30025  -3.592 0.000837 ***
d(L(IBO, 1))  0.70701    0.44359   1.594 0.118300
d(L(IBO, 2))  0.99468    0.36491   2.726 0.009242 **
d(IDE)        0.12546    0.48290   0.260 0.796248
d(L(IDE, 1)) -1.40786    0.48867  -2.881 0.006160 **
ect          -0.41685    0.07849  -5.311 3.63e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.01819 on 43 degrees of freedom
  (0 observations deleted due to missingness)
Multiple R-squared:  0.7613,    Adjusted R-squared:  0.7113
F-statistic: 15.24 on 9 and 43 DF,  p-value: 9.545e-11
```

```
# Short Run Coefficients
multipliers(ardl_3132, type = "sr")
```

```
         Term   Estimate Std. Error    t value     Pr(>|t|)
1 (Intercept)  2.6201916  0.5677679  4.6148990 4.186867e-05
2         LRY  0.6727993  0.1311638  5.1294603 8.317401e-06
3         IBO -1.0785180  0.3217011 -3.3525465 1.790030e-03
4         IDE  0.1254643  0.5544522  0.2262852 8.221614e-01
```

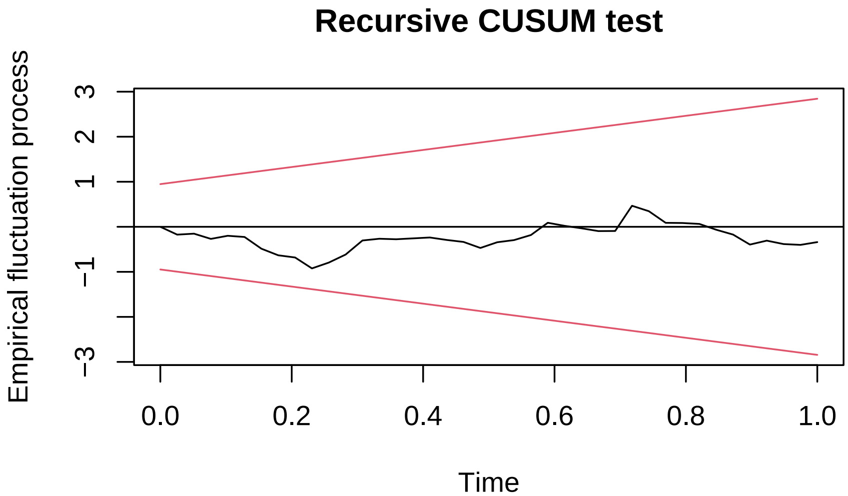
```
# Long Run Coefficients
multipliers(ardl_3132, type = "lr")
```

```
         Term   Estimate Std. Error   t value     Pr(>|t|)
1 (Intercept)  6.2856579  0.7719160  8.142930 6.107445e-10
2         LRY  0.9964676  0.1239310  8.040503 8.358472e-10
3         IBO -4.5381160  0.5202961 -8.722180 1.058619e-10
4         IDE  2.8915201  0.9950853  2.905801 6.009239e-03
```

```
library(lmtest) # for resettest()
library(strucchange) # for efp(), and sctest()
resettest(uecm_3132, type = c("regressor"))
```

```
    RESET test

data:  uecm_3132
RESET = 1.2408, df1 = 24, df2 = 15, p-value = 0.3383
```

```
# Stability
uecm_3132_lm_names <- to_lm(uecm_3132, fix_names = TRUE)
fluctuation <- efp(uecm_3132_lm_names$full_formula,
                   data = uecm_3132_lm_names$model)
plot(fluctuation)
```


Recursive CUSUM test
Empirical fluctuation process
3
2
1
−1
−3
0.0
0.2
0.4
0.6
0.8
1.0
Time

Chapter 5

Panel Data Regression

```r
library(readxl)
datapanel = read_excel("Data/Bab 5/Data Panel.xlsx")
head(datapanel)
```

```
# A tibble: 6 x 6
  province  year realgdp population investment   hdi
  <chr>    <dbl>   <dbl>      <dbl>      <dbl> <dbl>
1 Aceh      2010 101545.    4523100       82.3  67.1
2 Aceh      2011 104874.    4619000      463.   67.4
3 Aceh      2012 108915.    4715100     1726.   67.8
4 Aceh      2013 111756.    4811100     4785.   68.3
5 Aceh      2014 113488.    4906800     5497.   68.8
6 Aceh      2015 112672.    5002000     4485.   69.4
```

5.1 Static Panel Data

```r
library(plm)
model1 = log(realgdp) ~ log(population) + log(investment) + log(hdi)
```

```r
# time vs individual effect
pFtest(model1, data = datapanel, effect = "time")
```

```

    F test for time effects

data:  model1
F = 3.1305, df1 = 6, df2 = 221, p-value = 0.005777
alternative hypothesis: significant effects
```

```r
pFtest(model1, data = datapanel, effect = "individual")
```

```

	F test for individual effects

data:  model1
F = 824.45, df1 = 32, df2 = 195, p-value < 2.2e-16
alternative hypothesis: significant effects
```

```r
pFtest(model1, data = datapanel, effect = "twoways")
```

```

	F test for twoways effects

data:  model1
F = 812.75, df1 = 38, df2 = 189, p-value < 2.2e-16
alternative hypothesis: significant effects
```

5.1.1 Pooled OLS

```r
POLS <- plm(model1, data = datapanel,
                      index = c("province", "year"),
                      effect = "twoways", model = "pooling")
summary(POLS)
```

```
Pooling Model

Call:
plm(formula = model1, data = datapanel, effect = "twoways", model = "pooling",
    index = c("province", "year"))

Balanced Panel: n = 33, T = 7, N = 231

Residuals:
     Min.   1st Qu.    Median   3rd Qu.      Max.
-0.802950 -0.288399 -0.071922  0.204733  1.211596

Coefficients:
                  Estimate Std. Error t-value  Pr(>|t|)
(Intercept)     -14.998414   1.789923 -8.3794 5.578e-15 ***
log(population)   0.761682   0.034219 22.2592 < 2.2e-16 ***
log(investment)   0.200796   0.018761 10.7031 < 2.2e-16 ***
log(hdi)          3.187128   0.432136  7.3753 3.050e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Total Sum of Squares:    316.63
Residual Sum of Squares: 37.477
R-Squared:      0.88164
Adj. R-Squared: 0.88008
F-statistic: 563.624 on 3 and 227 DF, p-value: < 2.22e-16
```

5.1.2 Fixed Effects Model

```r
FEM <- plm(model1, data = datapanel,
                      index = c("province", "year"),
                      effect = "twoways", model = "within")

summary(FEM)
```

```
Twoways effects Within Model

Call:
plm(formula = model1, data = datapanel, effect = "twoways", model = "within",
    index = c("province", "year"))

Balanced Panel: n = 33, T = 7, N = 231

Residuals:
      Min.    1st Qu.     Median    3rd Qu.       Max.
-0.1065933 -0.0108718  0.0010591  0.0107126  0.1302578

Coefficients:
                  Estimate Std. Error t-value Pr(>|t|)
log(population) -0.4403249  0.2194092 -2.0069  0.04619 *
log(investment) -0.0046652  0.0039298 -1.1871  0.23666
log(hdi)         1.4472500  0.7092888  2.0404  0.04270 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    0.24452
Residual Sum of Squares: 0.22795
R-Squared:      0.067783
Adj. R-Squared: -0.13444
F-statistic: 4.58081 on 3 and 189 DF, p-value: 0.0040297
```

```r
# FEM vs. Pooled OLS
pFtest(FEM, POLS)
```

```
    F test for twoways effects
```

```
data:  model1
F = 812.75, df1 = 38, df2 = 189, p-value < 2.2e-16
alternative hypothesis: significant effects
```

5.1.3 Random Effects Model

```
REM <- plm(model1, data = datapanel,
                      index = c("province", "year"),
                      effect = "twoways", model = "random")
summary(REM)
```

```
Twoways effects Random Effect Model
   (Swamy-Arora's transformation)

Call:
plm(formula = model1, data = datapanel, effect = "twoways", model = "random",
    index = c("province", "year"))

Balanced Panel: n = 33, T = 7, N = 231

Effects:
                   var  std.dev share
idiosyncratic 0.001206 0.034728 0.008
individual    0.141041 0.375555 0.992
time          0.000000 0.000000 0.000
theta: 0.9651 (id) 0 (time) 0 (total)

Residuals:
      Min.    1st Qu.     Median    3rd Qu.       Max.
-0.1099833 -0.0205049  0.0012902  0.0165381  0.1635731

Coefficients:
                   Estimate  Std. Error z-value Pr(>|z|)
(Intercept)     -2.1127e+01  1.0047e+00 -21.029   <2e-16 ***
log(population)  8.9072e-01  7.1716e-02  12.420   <2e-16 ***
log(investment)  9.0107e-04  4.1329e-03   0.218   0.8274
log(hdi)         4.5687e+00  2.2971e-01  19.889   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    3.599
Residual Sum of Squares: 0.35333
R-Squared:      0.90183
Adj. R-Squared: 0.90053
Chisq: 2085.22 on 3 DF, p-value: < 2.22e-16
```

5.1.4 Hausman Test

```
phtest(FEM,REM)
```

```

	Hausman Test

data:  model1
chisq = 46.609, df = 3, p-value = 4.208e-10
alternative hypothesis: one model is inconsistent
```

5.1.5 Model Diagnostics

```
# Multicolinearity
library(car)
```

```
Loading required package: carData
```

```
vif(POLS)
```

```
log(population) log(investment)        log(hdi)
       1.596289        1.641147        1.119590
```

```
cor(datapanel[,4:6])
```

```
           population investment       hdi
population  1.0000000  0.7052198 0.1212988
investment  0.7052198  1.0000000 0.3222325
hdi         0.1212988  0.3222325 1.0000000
```

```
library(lmtest)
```

```
Loading required package: zoo


Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric
```

```
# Heteroscedasticity
bptest(FEM)
```

```

	studentized Breusch-Pagan test

data:  FEM
BP = 32.396, df = 3, p-value = 4.319e-07
```

```
# Autocorrelation
pbgtest(FEM)
```

```
	Breusch-Godfrey/Wooldridge test for serial correlation in panel models

data:  model1
chisq = 84.76, df = 7, p-value = 1.468e-15
alternative hypothesis: serial correlation in idiosyncratic errors
```

```
# Cluster Robust Standard Error
library(sandwich)
coeftest(FEM, vcovHC(FEM, type = "sss", cluster = "group"))
```

```

t test of coefficients:

                  Estimate Std. Error t value Pr(>|t|)
log(population) -0.4403249  0.4093013 -1.0758   0.2834
log(investment) -0.0046652  0.0056843 -0.8207   0.4128
log(hdi)         1.4472500  1.0807171  1.3392   0.1821
```

```
# HAC Robust Standard Error
coeftest(FEM, vcovHC(FEM, method="arellano"))
```

```

t test of coefficients:

                  Estimate Std. Error t value Pr(>|t|)
log(population) -0.4403249  0.4012959 -1.0973   0.2739
log(investment) -0.0046652  0.0055732 -0.8371   0.4036
log(hdi)         1.4472500  1.0595795  1.3659   0.1736
```

5.2 Dynamic Panel Data

```
head(datapanel)
```

```
# A tibble: 6 x 6
  province  year realgdp population investment   hdi
  <chr>    <dbl>   <dbl>      <dbl>      <dbl> <dbl>
1 Aceh      2010 101545.    4523100       82.3  67.1
2 Aceh      2011 104874.    4619000      463.   67.4
3 Aceh      2012 108915.    4715100     1726.   67.8
4 Aceh      2013 111756.    4811100     4785.   68.3
5 Aceh      2014 113488.    4906800     5497.   68.8
6 Aceh      2015 112672.    5002000     4485.   69.4
```

```
# lag(log(realgdp), 2:7) = Instrument
modeldyn1 = log(realgdp) ~ lag(log(realgdp)) + log(population) + log(investment) + log(hdi) | lag
# Dynamic OLS and FEM
modeldyn2 = log(realgdp) ~ lag(log(realgdp)) + log(population) + log(investment) + log(hdi)
```

5.2.1 First Difference GMM

```
fd.gmm = pgmm(modeldyn1, data = datapanel)
```

```
Warning in pgmm(modeldyn1, data = datapanel): the second-step matrix is
singular, a general inverse is used
```

```
summary(fd.gmm)
```

```
Warning in vcovHC.pgmm(object): a general inverse is used

Twoways effects One-step model Difference GMM

Call:
pgmm(formula = modeldyn1, data = datapanel)

Balanced Panel: n = 33, T = 7, N = 231

Number of Observations Used: 165
Residuals:
      Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
-0.0926516 -0.0071905 -0.0006685  0.0000000  0.0051944  0.1291078

Coefficients:
                    Estimate Std. Error z-value  Pr(>|z|)
lag(log(realgdp))  0.7312958  0.2057587  3.5541 0.0003792 ***
log(population)   -0.1750407  0.1798151 -0.9734 0.3303305
log(investment)    0.0013969  0.0022465  0.6218 0.5340580
log(hdi)           2.1197608  1.3035304  1.6262 0.1039137
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Sargan test: chisq(14) = 25.4339 (p-value = 0.030518)
Autocorrelation test (1): normal = -1.836971 (p-value = 0.066214)
Autocorrelation test (2): normal = 1.555262 (p-value = 0.11988)
Wald test for coefficients: chisq(4) = 95.86464 (p-value = < 2.22e-16)
Wald test for time dummies: chisq(5) = 8.364699 (p-value = 0.13725)
```

5.2.2 System GMM

```
sys.gmm = pgmm(modeldyn1, data = datapanel, transformation="ld")
```

```
Warning in pgmm(modeldyn1, data = datapanel, transformation = "ld"): the
second-step matrix is singular, a general inverse is used
```

```
summary(sys.gmm)
```

```
Warning in vcovHC.pgmm(object): a general inverse is used
```

```
Twoways effects One-step model System GMM

Call:
pgmm(formula = modeldyn1, data = datapanel, transformation = "ld")

Balanced Panel: n = 33, T = 7, N = 231

Number of Observations Used: 363
Residuals:
      Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
-0.1198240 -0.0083907  0.0001403  0.0000000  0.0065018  0.1402918

Coefficients:
                   Estimate Std. Error z-value  Pr(>|z|)
lag(log(realgdp)) 0.9686425  0.0107842 89.8208 < 2.2e-16 ***
log(population)   0.0207419  0.0078302  2.6490  0.008074 **
log(investment)   0.0070513  0.0029150  2.4190  0.015565 *
log(hdi)          0.1144280  0.0582058  1.9659  0.049308 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Sargan test: chisq(22) = 23.67029 (p-value = 0.36475)
Autocorrelation test (1): normal = -1.663193 (p-value = 0.096274)
Autocorrelation test (2): normal = 1.368565 (p-value = 0.17114)
Wald test for coefficients: chisq(4) = 314880.1 (p-value = < 2.22e-16)
Wald test for time dummies: chisq(5) = 29.57621 (p-value = 1.7869e-05)
```

5.2.3 Model Diagnostics

```
# FEM
FEMdyn = plm(modeldyn2, data = datapanel, index=c("province","year"), model="within")
summary(FEMdyn)
```

```
Oneway (individual) effect Within Model

Call:
```

```
plm(formula = modeldyn2, data = datapanel, model = "within",
    index = c("province", "year"))

Balanced Panel: n = 33, T = 6, N = 198

Residuals:
       Min.     1st Qu.      Median     3rd Qu.        Max.
-5.8397e-02 -6.5483e-03  1.4102e-05  5.9536e-03  1.1759e-01

Coefficients:
                    Estimate Std. Error t-value  Pr(>|t|)
lag(log(realgdp))  0.7670629  0.0397433 19.3004 < 2.2e-16 ***
log(population)   -0.2488432  0.1100829 -2.2605   0.02513 *
log(investment)   -0.0014210  0.0024243 -0.5861   0.55862
log(hdi)           1.8786968  0.2913007  6.4493 1.253e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    1.9728
Residual Sum of Squares: 0.050149
R-Squared:      0.97458
Adj. R-Squared: 0.96889
F-statistic: 1543.1 on 4 and 161 DF, p-value: < 2.22e-16
```

```r
# OLS
OLSdyn = plm(modeldyn2, data = datapanel, index=c("province","year"), model="pooling")
summary(OLSdyn)
```

```
Pooling Model

Call:
plm(formula = modeldyn2, data = datapanel, model = "pooling",
    index = c("province", "year"))

Balanced Panel: n = 33, T = 6, N = 198

Residuals:
       Min.     1st Qu.      Median     3rd Qu.        Max.
-0.10036444 -0.00975944 -0.00044271  0.00946369  0.12897120

Coefficients:
                    Estimate Std. Error  t-value Pr(>|t|)
(Intercept)       -0.1070843  0.1261711  -0.8487  0.39709
lag(log(realgdp))  0.9868710  0.0041424 238.2382  < 2e-16 ***
log(population)    0.0088980  0.0036806   2.4175  0.01656 *
log(investment)    0.0016145  0.0016016   1.0080  0.31470
```

```
log(hdi)           0.0397307  0.0297085   1.3374  0.18268
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    269.87
Residual Sum of Squares: 0.09922
R-Squared:      0.99963
Adj. R-Squared: 0.99962
F-statistic: 131187 on 4 and 193 DF, p-value: < 2.22e-16
```

FDGMM = 0.731 SysGMM = 0.968 FEM = 0.767 OLS = 0.986

FEM < GMM < OLS Best Model: System GMM

```
summary(sys.gmm)
```

```
Warning in vcovHC.pgmm(object): a general inverse is used


Twoways effects One-step model System GMM

Call:
pgmm(formula = modeldyn1, data = datapanel, transformation = "ld")

Balanced Panel: n = 33, T = 7, N = 231

Number of Observations Used: 363
Residuals:
      Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
-0.1198240 -0.0083907  0.0001403  0.0000000  0.0065018  0.1402918

Coefficients:
                   Estimate Std. Error z-value  Pr(>|z|)
lag(log(realgdp)) 0.9686425  0.0107842 89.8208 < 2.2e-16 ***
log(population)   0.0207419  0.0078302  2.6490  0.008074 **
log(investment)   0.0070513  0.0029150  2.4190  0.015565 *
log(hdi)          0.1144280  0.0582058  1.9659  0.049308 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Sargan test: chisq(22) = 23.67029 (p-value = 0.36475)
Autocorrelation test (1): normal = -1.663193 (p-value = 0.096274)
Autocorrelation test (2): normal = 1.368565 (p-value = 0.17114)
Wald test for coefficients: chisq(4) = 314880.1 (p-value = < 2.22e-16)
Wald test for time dummies: chisq(5) = 29.57621 (p-value = 1.7869e-05)
```

5.2.4 Speed of Adjustment

```
alpha1 = sys.gmm$coef[1]
1-alpha1
```

```
lag(log(realgdp))
       0.03135749
```

5.2.5 Half Time

```
log(0.5)/log(sys.gmm$coef[1])
```

```
lag(log(realgdp))
         21.75626
```

5.2.6 Short Run and Long Run Coefficients

```
sys.gmm$coefficients[2] # Short Run Poppulation
```

```
log(population)
     0.02074192
```

```
sys.gmm$coefficients[2] / (1-alpha1) # Long Run Poppulation
```

```
log(population)
      0.6614661
```

Chapter 6

Spatial Regression

6.1 Library

```r
# install.packages("spdep")
# install.packages("spatialreg")
# install.packages("RColorBrewer")
# install.packages("splm")
# install.packages("sf")
# install.packages("ggplot2")
library(spdep)
```

```
Warning: package 'spdep' was built under R version 4.4.3
```

```
Loading required package: spData
```

```
Warning: package 'spData' was built under R version 4.4.3
```

```
To access larger datasets in this package, install the spDataLarge
package with: `install.packages('spDataLarge',
repos='https://nowosad.github.io/drat/', type='source')`
```

```
Loading required package: sf
```

```
Warning: package 'sf' was built under R version 4.4.3
```

```
Linking to GEOS 3.13.0, GDAL 3.10.1, PROJ 9.5.1; sf_use_s2() is TRUE
```

```r
library(spatialreg)
```

```
Warning: package 'spatialreg' was built under R version 4.4.3
```

```
Loading required package: Matrix
```

```
Attaching package: 'spatialreg'

The following objects are masked from 'package:spdep':

    get.ClusterOption, get.coresOption, get.mcOption,
    get.VerboseOption, get.ZeroPolicyOption, set.ClusterOption,
    set.coresOption, set.mcOption, set.VerboseOption,
    set.ZeroPolicyOption
```

```
library(RColorBrewer)
library(splm)
```

```
Warning: package 'splm' was built under R version 4.4.3
```

```
library(sf)
library(ggplot2)
```

```
Warning: package 'ggplot2' was built under R version 4.4.3
```

6.2 Cross-Section

```
library(readxl)
provinsi <- read_excel("Data/Bab6/provinsi Indonesia.xlsx")
head(provinsi)
```

```
# A tibble: 6 x 5
  province    pdrb investment infra revenue
  <chr>      <dbl>      <dbl> <dbl>   <dbl>
1 Aceh     129093.      4485.  0.37  11694.
2 Sumut    571722.     21477.  0.52   8481.
3 Sumbar   179952.      2340.  0.52   4052.
4 Riau     652762.     18957.  0.28   6911.
5 Jambi    155066.      5026.  0.26   3130.
6 Sumsel   331766.     19853.  0.2    5990.
```

6.2.1 OLS Model

```
model1 = log(pdrb) ~ log(investment) + log(infra) + log(revenue)
ols = lm(model1, data=provinsi)
summary(ols)
```

```
Call:
lm(formula = model1, data = provinsi)

Residuals:
     Min       1Q   Median       3Q      Max
```

```
-0.82306 -0.33812  0.00604  0.32399  0.83581

Coefficients:
                Estimate Std. Error t value Pr(>|t|)    
(Intercept)      3.95883    0.85379   4.637 6.48e-05 ***
log(investment)  0.42187    0.08375   5.037 2.10e-05 ***
log(infra)       0.24988    0.07807   3.201  0.00323 ** 
log(revenue)     0.53807    0.15207   3.538  0.00133 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4094 on 30 degrees of freedom
Multiple R-squared:  0.8883,	Adjusted R-squared:  0.8771 
F-statistic: 79.53 on 3 and 30 DF,  p-value: 2.226e-14
```

6.2.2 Weight Matrix

```r
migrasi <- read_excel("Data/Bab6/matriks migrasi.xlsx", sheet = 2, col_names = FALSE)
```

```
New names:
* `` -> `...1`
* `` -> `...2`
* `` -> `...3`
* `` -> `...4`
* `` -> `...5`
* `` -> `...6`
* `` -> `...7`
* `` -> `...8`
* `` -> `...9`
* `` -> `...10`
* `` -> `...11`
* `` -> `...12`
* `` -> `...13`
* `` -> `...14`
* `` -> `...15`
* `` -> `...16`
* `` -> `...17`
* `` -> `...18`
* `` -> `...19`
* `` -> `...20`
* `` -> `...21`
* `` -> `...22`
* `` -> `...23`
* `` -> `...24`
* `` -> `...25`
```

```
* `` -> `...26`
* `` -> `...27`
* `` -> `...28`
* `` -> `...29`
* `` -> `...30`
* `` -> `...31`
* `` -> `...32`
* `` -> `...33`
* `` -> `...34`
```

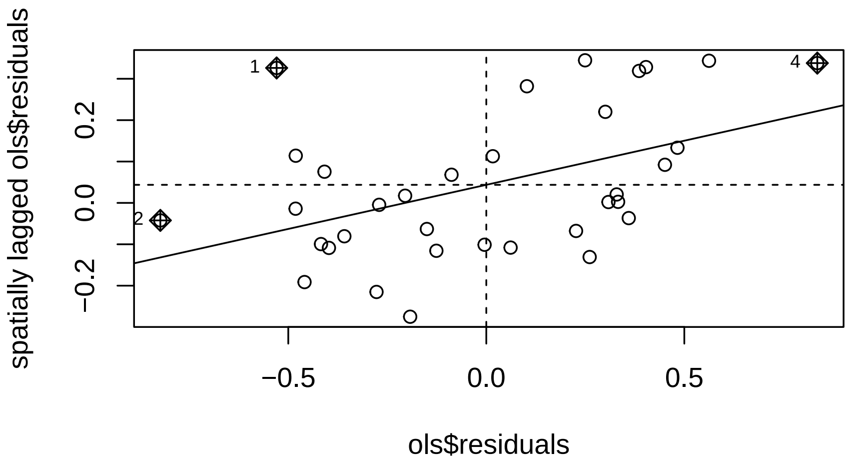
```
migrasi = as.matrix(migrasi)
W.migrasi = mat2listw(migrasi)
```

```
Warning in mat2listw(migrasi): style is M (missing); style should be set to a
valid value
```

```
moran.lm = lm.morantest(ols, W.migrasi)
moran.lm
```

```

    Global Moran I for regression residuals

data:
model: lm(formula = model1, data = provinsi)
weights: W.migrasi

Moran I statistic standard deviate = 3.4669, p-value = 0.0002632
alternative hypothesis: greater
sample estimates:
Observed Moran I      Expectation         Variance
     0.213239417     -0.048308437      0.005691365
```

```
moran.plot(ols$residuals, W.migrasi)
```



6.2.4 LM Test

```
LM = lm.LMtests(ols, W.migrasi, test="all")
```

```
Please update scripts to use lm.RStests in place of lm.LMtests

Warning in lm.RStests(model = model, listw = listw, zero.policy = zero.policy,
: Spatial weights matrix not row standardized
```

```
LM
```

```
	Rao's score (a.k.a Lagrange multiplier) diagnostics for spatial
	dependence

data:
model: lm(formula = model1, data = provinsi)
test weights: listw

RSerr = 5.4456, df = 1, p-value = 0.01962


	Rao's score (a.k.a Lagrange multiplier) diagnostics for spatial
	dependence

data:
model: lm(formula = model1, data = provinsi)
```

```
test weights: listw

RSlag = 3.2163, df = 1, p-value = 0.07291


    Rao's score (a.k.a Lagrange multiplier) diagnostics for spatial
    dependence

data:
model: lm(formula = model1, data = provinsi)
test weights: listw

adjRSerr = 3.0702, df = 1, p-value = 0.07974


    Rao's score (a.k.a Lagrange multiplier) diagnostics for spatial
    dependence

data:
model: lm(formula = model1, data = provinsi)
test weights: listw

adjRSlag = 0.84087, df = 1, p-value = 0.3591


    Rao's score (a.k.a Lagrange multiplier) diagnostics for spatial
    dependence

data:
model: lm(formula = model1, data = provinsi)
test weights: listw

SARMA = 6.2865, df = 2, p-value = 0.04314
```

6.2.5 SAR Model

```r
sar.provinsi = lagsarlm(model1, data=provinsi, W.migrasi)
summary(sar.provinsi)
```

```
Call:lagsarlm(formula = model1, data = provinsi, listw = W.migrasi)

Residuals:
      Min        1Q    Median        3Q       Max
-0.679482 -0.291161 -0.083437  0.336403  0.808845
```

```
Type: lag
Coefficients: (asymptotic standard errors)
                Estimate Std. Error z value  Pr(>|z|)
(Intercept)     1.744150   1.437574  1.2133 0.2250310
log(investment) 0.380784   0.079364  4.7979 1.603e-06
log(infra)      0.207431   0.073675  2.8155 0.0048701
log(revenue)    0.529174   0.136397  3.8797 0.0001046

Rho: 0.21114, LR test value: 3.0033, p-value: 0.083096
Asymptotic standard error: 0.12002
    z-value: 1.7592, p-value: 0.07854
Wald statistic: 3.0949, p-value: 0.07854

Log likelihood: -14.24739 for lag model
ML residual variance (sigma squared): 0.13454, (sigma: 0.36679)
Number of observations: 34
Number of parameters estimated: 6
AIC: 40.495, (AIC for lm: 41.498)
LM test for residual autocorrelation
test value: 2.5558, p-value: 0.10989
```

6.2.6 Impacts (Spillover)

```
impacts(sar.provinsi, listw=W.migrasi)
```

```
Impact measures (lag, exact):
                   Direct   Indirect     Total
log(investment) 0.3831961 0.09950271 0.4826988
log(infra)      0.2087457 0.05420401 0.2629497
log(revenue)    0.5325268 0.13827873 0.6708055
```

6.2.7 SEM Model

```
sem.provinsi = errorsarlm(model1, data=provinsi, W.migrasi)
summary(sem.provinsi)
```

```
Call:errorsarlm(formula = model1, data = provinsi, listw = W.migrasi)

Residuals:
      Min        1Q    Median        3Q       Max
-0.770148 -0.273480 -0.020662  0.325690  0.647156

Type: error
```

```
Coefficients: (asymptotic standard errors)
                Estimate Std. Error z value  Pr(>|z|)
(Intercept)     3.831947   0.772493  4.9605 7.031e-07
log(investment) 0.410266   0.072801  5.6354 1.746e-08
log(infra)      0.209178   0.071237  2.9364  0.003321
log(revenue)    0.550807   0.132071  4.1705 3.039e-05

Lambda: 0.60794, LR test value: 5.1345, p-value: 0.023455
Asymptotic standard error: 0.19219
    z-value: 3.1632, p-value: 0.0015604
Wald statistic: 10.006, p-value: 0.0015604

Log likelihood: -13.18178 for error model
ML residual variance (sigma squared): 0.11956, (sigma: 0.34577)
Number of observations: 34
Number of parameters estimated: 6
AIC: 38.364, (AIC for lm: 41.498)
```

6.3 Spatial Panel

```
library(readxl)
paneljateng <- read_excel("Data/Bab6/panel jateng.xlsx")
head(paneljateng)
```

```
# A tibble: 6 x 8
  Region        Tahun      PDRB     AK     PAD      UMK   IPM NO
  <chr>         <dbl>     <dbl>  <dbl>   <dbl>    <dbl> <dbl> <chr>
1 Kab. Cilacap   2011 78156819. 797518 173141.  718667.  64.7 01
2 Kab. Cilacap   2012 79702238. 716465 196673.  773000   65.7 01
3 Kab. Cilacap   2013 81022670. 729059 278508.  887667   66.8 01
4 Kab. Cilacap   2014 83392999. 736247 373907. 1016667.  67.2 01
5 Kab. Cilacap   2015 88777805. 715819 409846. 1195667.  67.8 01
6 Kab. Banyumas  2011 24538596. 761034 193263.  750000   67.4 02
```

6.3.1 Static Panel Regression

```
library(plm)
modelpanel = log(PDRB) ~ log(AK) + log(PAD) + log(UMK) + log(IPM)
fem1 = plm(modelpanel, data=paneljateng, index=c("Region", "Tahun"), model="within")
rem1 = plm(modelpanel, data=paneljateng, index=c("Region", "Tahun"), model="random")
phtest(fem1, rem1)
```

```
    Hausman Test
```

```
data:  modelpanel
chisq = 37.156, df = 4, p-value = 1.673e-07
alternative hypothesis: one model is inconsistent
```

```r
library(lmtest)
```

```
Loading required package: zoo


Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric
```

```r
bptest(fem1)
```

```

    studentized Breusch-Pagan test

data:  fem1
BP = 5.8081, df = 4, p-value = 0.2139
```

```r
pbgtest(fem1)
```

```

    Breusch-Godfrey/Wooldridge test for serial correlation in panel models

data:  modelpanel
chisq = 51.619, df = 5, p-value = 6.458e-10
alternative hypothesis: serial correlation in idiosyncratic errors
```

6.3.2 Depndency Test

```r
pcdtest(fem1, test="lm")
```

```

    Breusch-Pagan LM test for cross-sectional dependence in panels

data:  log(PDRB) ~ log(AK) + log(PAD) + log(UMK) + log(IPM)
chisq = 1268.3, df = 595, p-value < 2.2e-16
alternative hypothesis: cross-sectional dependence
```

```r
pcdtest(fem1, test="cd")
```

```

    Pesaran CD test for cross-sectional dependence in panels

data:  log(PDRB) ~ log(AK) + log(PAD) + log(UMK) + log(IPM)
```

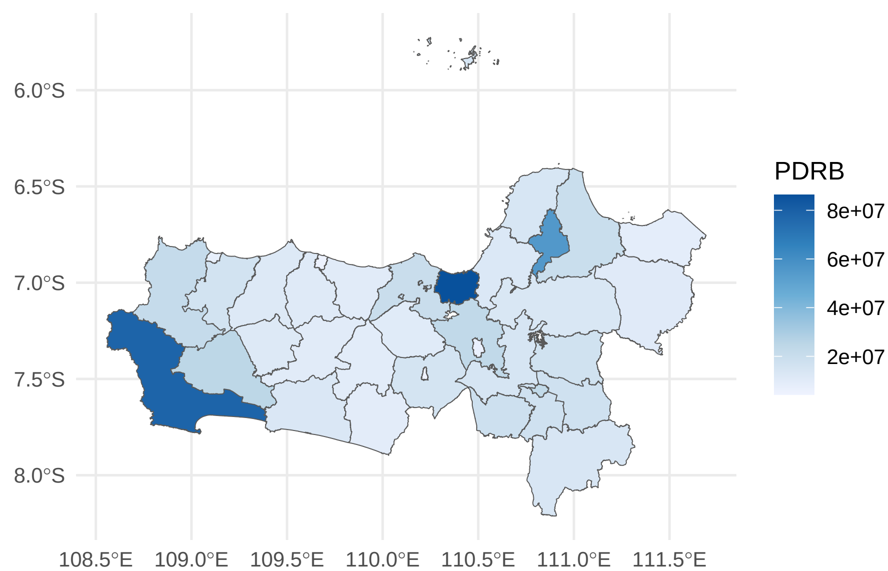
```
z = 12.724, p-value < 2.2e-16
alternative hypothesis: cross-sectional dependence
```

6.3.3 Maps Visualization

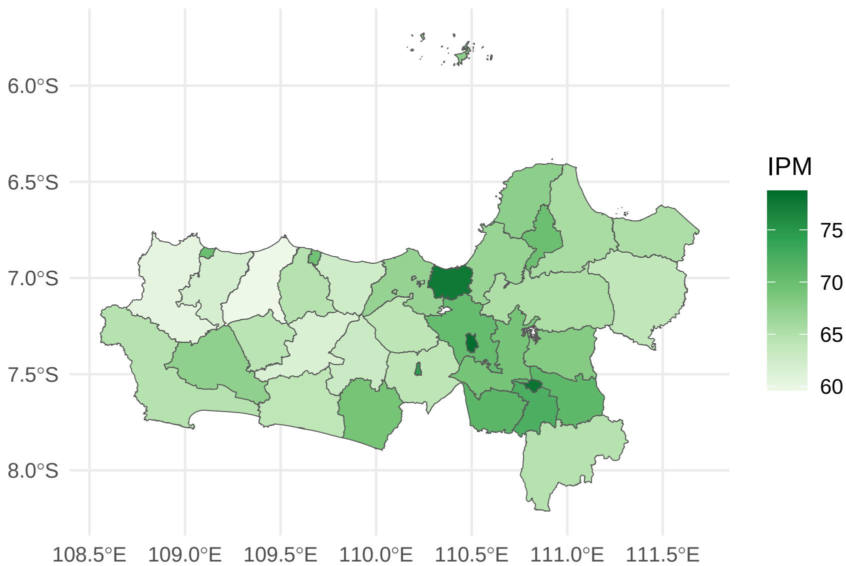
```
jateng.map = st_read('Data/Bab6/peta jateng/Jawa_Tengah.shp')
```

```
Reading layer `Jawa_Tengah' from data source
  `D:\Dokumentasi\econometricsbook\Data\Bab6\peta jateng\Jawa_Tengah.shp'
  using driver `ESRI Shapefile'
Simple feature collection with 35 features and 4 fields
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: 108.5559 ymin: -8.211962 xmax: 111.6914 ymax: -5.725698
Geodetic CRS:  WGS 84
```

```
jateng2011 = subset(paneljateng,(Tahun==2011))
jateng2011 = merge(jateng.map, jateng2011, by.x="KABKOTNO", by.y="NO")
jateng2011 <- st_make_valid(jateng2011)
ggplot(jateng2011) +
  geom_sf(aes(fill = PDRB)) +
  scale_fill_gradientn(colours = brewer.pal(5, "Blues"),
                       values = scales::rescale(seq(min(jateng2011$PDRB),
                                                    max(jateng2011$PDRB)*1.01,
                                                    length = 6))) +
  theme_minimal() +
  labs(fill = "PDRB")
```



```
ggplot(jateng2011) +
  geom_sf(aes(fill = IPM)) +
  scale_fill_gradientn(colours = brewer.pal(5, "Greens"),
                       values = scales::rescale(seq(min(jateng2011$IPM),
                                                    max(jateng2011$IPM)*1.01,
                                                    length = 6))) +
  theme_minimal() +
  labs(fill = "IPM")
```



6.3.4 Function for Spatial Panel Evalutation

```
godf.spml<-function(object, k=2, criterion=c("AIC", "BIC"),  ...){
  s<-summary(object)
  l<-s$logLik
  np<- length(coef(s))
  N<- nrow(s$model)
  if(criterion=="AIC"){
    aic<- -2*l+k*np
    names(aic)<-"AIC"
    return(aic)
  }
  if(criterion=="BIC"){
    bic<- -2*l+log(N)*np
    names(bic)<-"BIC"
    if(k!=2){
      warning("parameter <k> not used for BIC")
```

```
    }
    return(bic)
  }
}
```

6.3.5 Spatial Panel Model with Contiguity Weight Matrix

```
jateng.map <- st_make_valid(jateng.map)
listqueen = poly2nb(jateng.map, queen=TRUE)
W.queen = nb2listw(listqueen, style="W")
W.queen
```

```
Characteristics of weights list object:
Neighbour list object:
Number of regions: 35
Number of nonzero links: 148
Percentage nonzero weights: 12.08163
Average number of links: 4.228571

Weights style: W
Weights constants summary:
   n   nn S0       S1       S2
W 35 1225 35 18.64242 151.0178
```

```
# SAR Model
sar.fem.contig = spml(modelpanel, data=paneljateng, listw=W.queen, model="within", lag=
sar.rem.contig = spml(modelpanel, data=paneljateng, listw=W.queen, model="random", lag=
sphtest(sar.fem.contig, sar.rem.contig)
```

```

    Hausman test for spatial models

data:  modelpanel
chisq = 0.15855, df = 4, p-value = 0.997
alternative hypothesis: one model is inconsistent
```

```
godf.spml(sar.rem.contig, criterion="AIC")
```

```
      AIC
-703.0118
```

```
# SEM Model
sem.fem.contig = spml(modelpanel, data=paneljateng, listw=W.queen, model="within", lag=
sem.rem.contig = spml(modelpanel, data=paneljateng, listw=W.queen, model="random", lag=
sphtest(sem.fem.contig, sem.rem.contig)
```

```
    Hausman test for spatial models

data:  modelpanel
chisq = 5.4546, df = 4, p-value = 0.2438
alternative hypothesis: one model is inconsistent
```

```
godf.spml(sem.rem.contig, criterion="AIC")
```

```
      AIC
-633.5611
```

6.3.6 Spatial Panel Model with KNN Weight Matrix

```
# K-nearest neighbour with 5 neighbour
centroids <- st_centroid(jateng.map)
```

```
Warning: st_centroid assumes attributes are constant over geometries
```

```
coords <- st_coordinates(centroids)
neighbour = knearneigh(coords, k=5, longlat=T)
neighbourlist = knn2nb(neighbour)
mat.knn5 = nb2mat(neighbourlist, style="W")
W.knn5 = nb2listw(neighbourlist, style="W")
W.knn5
```

```
Characteristics of weights list object:
Neighbour list object:
Number of regions: 35
Number of nonzero links: 175
Percentage nonzero weights: 14.28571
Average number of links: 5
Non-symmetric neighbours list

Weights style: W
Weights constants summary:
   n   nn S0    S1     S2
W 35 1225 35 12.44 144.48
```

```
# SAR Model
sar.fem.5nn = spml(modelpanel, data=paneljateng, listw=W.knn5, model="within", lag=TRUE, spatial.
sar.rem.5nn = spml(modelpanel, data=paneljateng, listw=W.knn5, model="random", lag=TRUE, spatial.
sphtest(sar.fem.5nn, sar.rem.5nn)
```

```
    Hausman test for spatial models

data:  modelpanel
chisq = 0.9208, df = 4, p-value = 0.9216
```

```
alternative hypothesis: one model is inconsistent
```

```
godf.spml(sar.rem.5nn, criterion="AIC")
```

```
      AIC
-717.8954
```

```
# SEM Model
sem.fem.5nn = spml(modelpanel, data=paneljateng, listw=W.knn5, model="within", lag=FALS
sem.rem.5nn = spml(modelpanel, data=paneljateng, listw=W.knn5, model="random", lag=FALS
sphtest(sem.fem.5nn, sem.rem.5nn)
```

```

    Hausman test for spatial models

data:  modelpanel
chisq = 5.3246, df = 4, p-value = 0.2556
alternative hypothesis: one model is inconsistent
```

```
godf.spml(sem.rem.5nn, criterion="AIC")
```

```
      AIC
-636.4432
```

6.3.7 Best Model

```
summary(sar.rem.5nn)
```

```
ML panel with spatial lag, random effects

Call:
spreml(formula = formula, data = data, index = index, w = listw2mat(listw),
    w2 = listw2mat(listw2), lag = lag, errors = errors, cl = cl)

Residuals:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   11.2    12.1    12.4    12.4    12.7    14.2

Error variance parameters:
    Estimate Std. Error t-value  Pr(>|t|)
phi   4204.2     1172.1  3.5868 0.0003348 ***

Spatial autoregressive coefficient:
       Estimate Std. Error t-value  Pr(>|t|)
lambda 0.750456   0.053423  14.047 < 2.2e-16 ***

Coefficients:
             Estimate Std. Error t-value Pr(>|t|)
```

```
(Intercept) 1.2042585  0.7177210  1.6779 0.093368 .
log(AK)     0.0474276  0.0241591  1.9631 0.049630 *
log(PAD)    0.0134898  0.0063019  2.1406 0.032306 *
log(UMK)    0.0470068  0.0167872  2.8002 0.005108 **
log(IPM)    0.3633197  0.1772088  2.0502 0.040342 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

6.3.8 Impacts (Spillovr) - Only for SAR Model

```
# Direct and Indirect Effect
time = length(unique(paneljateng$Tahun))
sW.5knn = kronecker(Diagonal(time), listw2dgCMatrix(W.knn5))
set.seed(12345)
trMatc = trW(sW.5knn, type="mult")
imp = impacts(sar.rem.5nn, tr = trMatc, R = 200)
summary(imp, zstats=TRUE, short=T)
```

```
Impact measures (lag, trace):
             Direct  Indirect      Total
log(AK)  0.05723256 0.1327900 0.19002256
log(PAD) 0.01627868 0.0377695 0.05404818
log(UMK) 0.05672481 0.1316119 0.18833673
log(IPM) 0.43843072 1.0172393 1.45567006
========================================================
Simulation results ( variance matrix):
========================================================
Simulated standard errors
              Direct   Indirect      Total
log(AK)  0.029013340 0.08364084 0.10974771
log(PAD) 0.007350001 0.02062333 0.02706877
log(UMK) 0.021472770 0.08001914 0.09849884
log(IPM) 0.228933484 0.65969232 0.86740920

Simulated z-values:
           Direct Indirect    Total
log(AK)  1.994004 1.692971 1.817389
log(PAD) 2.422582 2.079346 2.242031
log(UMK) 2.675263 1.800226 2.045687
log(IPM) 1.772324 1.503129 1.610943

Simulated p-values:
         Direct    Indirect Total
log(AK)  0.0461516 0.090461 0.069158
log(PAD) 0.0154107 0.037586 0.024959
```

```
log(UMK) 0.0074671 0.071825 0.040787
log(IPM) 0.0763407 0.132806 0.107192
```

Chapter 7

Time Series Spillover - GVAR

7.1 Library

```
library(Spillover)
```

```
Loading required package: vars

Loading required package: MASS

Loading required package: strucchange

Loading required package: zoo


Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric

Loading required package: sandwich

Loading required package: urca

Loading required package: lmtest
```

```
library(vars)
library(urca)
library(splitstackshape)
library(igraph)
```

```
Attaching package: 'igraph'

The following objects are masked from 'package:stats':

    decompose, spectrum

The following object is masked from 'package:base':

    union
```

```
library(reshape)
```

7.2 Data: Diebold-Yilmaz 2012

```
data(dy2012)
head(dy2012)  # in log volatility form
```

```
        Date     Stocks      Bonds Commodities         FX
1 1999-01-25  -9.891998 -10.081905   -9.797694 -12.971578
2 1999-01-26  -9.353294 -10.090498  -11.475212 -13.237477
3 1999-01-27  -9.314619 -10.103319  -15.317140  -9.749465
4 1999-01-28  -8.997370 -10.090498  -12.044040 -10.853610
5 1999-01-29  -8.855955  -9.426092  -12.928477 -11.788281
6 1999-02-01 -10.282395  -8.936206  -12.821930 -11.308455
```

log volatility return:

$$\sigma_{it}^2 = 0.361[ln(P_{i,t}^{max}) - ln(P_{i,t-1}^{min}))]^2$$

$$\sigma_{it} = 100 * (\sqrt{252 * \sigma_{it}^2})$$

```
class(dy2012)
```

```
[1] "data.frame"
```

```
nrow(dy2012)
```

```
[1] 2771
```

7.3 VAR Model

```
PP.test(dy2012$Stocks)
```

```

    Phillips-Perron Unit Root Test
```

```
data:  dy2012$Stocks
Dickey-Fuller = -32.546, Truncation lag parameter = 9, p-value = 0.01
```

```r
PP.test(dy2012$Bonds)
```

```
    Phillips-Perron Unit Root Test

data:  dy2012$Bonds
Dickey-Fuller = -41.249, Truncation lag parameter = 9, p-value = 0.01
```

```r
PP.test(dy2012$Commodities)
```

```
    Phillips-Perron Unit Root Test

data:  dy2012$Commodities
Dickey-Fuller = -48.523, Truncation lag parameter = 9, p-value = 0.01
```

```r
PP.test(dy2012$FX)
```

```
    Phillips-Perron Unit Root Test

data:  dy2012$FX
Dickey-Fuller = -47.527, Truncation lag parameter = 9, p-value = 0.01
```

```r
# Optimum Lag
VARselect(dy2012[,-1], lag.max = 4, type = c("both"))
```

```
$selection
AIC(n)  HQ(n)  SC(n) FPE(n) 
     4      4      4      4 

$criteria
                 1          2          3          4
AIC(n) -0.11329876 -0.4490460 -0.5882320 -0.6679637
HQ(n)  -0.09473562 -0.4181074 -0.5449180 -0.6122743
SC(n)  -0.06190286 -0.3633861 -0.4683083 -0.5137760
FPE(n)  0.89288389  0.6382368  0.5553084  0.5127520
```

```r
# VAR Model
VAR_4 <- VAR(dy2012[,-1], p=4)
VAR_4
```

```
VAR Estimation Results:
========================
```

```
Estimated coefficients for equation Stocks:
===========================================
Call:
Stocks = Stocks.l1 + Bonds.l1 + Commodities.l1 + FX.l1 + Stocks.l2 + Bonds.l2 + Commodi

     Stocks.l1       Bonds.l1 Commodities.l1          FX.l1      Stocks.l2
  0.1835084630  -0.0242487169  -0.0036337209   0.0219988664   0.2741423171
      Bonds.l2 Commodities.l2          FX.l2      Stocks.l3       Bonds.l3
  0.0027926328  -0.0108044031   0.0040316592   0.2025153665   0.0360740337
Commodities.l3          FX.l3      Stocks.l4       Bonds.l4 Commodities.l4
 -0.0158017651   0.0007938866   0.1620506569   0.0333733392  -0.0029608791
         FX.l4          const
 -0.0007384703  -1.3401330918


Estimated coefficients for equation Bonds:
==========================================
Call:
Bonds = Stocks.l1 + Bonds.l1 + Commodities.l1 + FX.l1 + Stocks.l2 + Bonds.l2 + Commodit

     Stocks.l1       Bonds.l1 Commodities.l1          FX.l1      Stocks.l2
   0.068141170    0.163772247    0.052592815    0.004319034    0.025511120
      Bonds.l2 Commodities.l2          FX.l2      Stocks.l3       Bonds.l3
   0.174716290    0.024936542    0.013962476    0.041353468    0.129894730
Commodities.l3          FX.l3      Stocks.l4       Bonds.l4 Commodities.l4
  -0.043136698    0.011844827   -0.026075514    0.218597337    0.039128430
         FX.l4          const
  -0.027491905   -1.080792116


Estimated coefficients for equation Commodities:
================================================
Call:
Commodities = Stocks.l1 + Bonds.l1 + Commodities.l1 + FX.l1 + Stocks.l2 + Bonds.l2 + Co

     Stocks.l1       Bonds.l1 Commodities.l1          FX.l1      Stocks.l2
  -0.022755735    0.098201350    0.183333080    0.048298645   -0.013291588
      Bonds.l2 Commodities.l2          FX.l2      Stocks.l3       Bonds.l3
  -0.013132921    0.207410140    0.044188995   -0.049827512    0.004592325
Commodities.l3          FX.l3      Stocks.l4       Bonds.l4 Commodities.l4
   0.185198136   -0.037523992   -0.030251601    0.060251069    0.156050483
         FX.l4          const
   0.038706445   -1.543777242
```

```
Estimated coefficients for equation FX:
=======================================
Call:
FX = Stocks.l1 + Bonds.l1 + Commodities.l1 + FX.l1 + Stocks.l2 + Bonds.l2 + Commodities.l2 + FX.l

     Stocks.l1       Bonds.l1 Commodities.l1          FX.l1      Stocks.l2
    0.06467245    -0.01930672     0.02517504     0.06996402     0.03590206
      Bonds.l2 Commodities.l2          FX.l2      Stocks.l3       Bonds.l3
   -0.02073267     0.01066279     0.20847309    -0.01249224     0.03780188
Commodities.l3          FX.l3      Stocks.l4       Bonds.l4 Commodities.l4
   -0.00393800     0.18185464    -0.01552454     0.05072537     0.01667104
         FX.l4          const
    0.11443511    -2.99838336
```

7.4 Volatility Spillover DY-2012

```r
# Total Spillover Index
sp <- G.spillover(VAR_4, n.ahead = 10, standardized = F )
sp
```

```
                               Stocks     Bonds Commodities        FX
Stocks                      88.757002  7.291185   0.3453279  3.606486
Bonds                       10.213545 81.445712   2.7269737  5.613770
Commodities                  0.468118  3.695953  93.6941893  2.141740
FX                           5.691579  7.026017   1.5477592 85.734645
C. to others (spillover)    16.373241 18.013154   4.6200608 11.361996
C. to others including own 105.130243 99.458866  98.3142500 97.096641
                           C. from others
Stocks                          11.242998
Bonds                           18.554288
Commodities                      6.305811
FX                              14.265355
C. to others (spillover)        12.592113
C. to others including own     400.000000
```

The total volatility spillover appears in the lower right corner of Table, which indicates that, on average, across our entire sample, 12.6% of the volatility forecast error variance in all four markets comes from spillovers

```r
Spillover::net(sp)
```

```
Warning in Spillover::net(sp): 'Spillover::net' is deprecated.
Use 'dynamic.spillover' instead.
See help("Deprecated")

                   To      From        Net Transmitter
Stocks      16.373241 11.242998  5.1302430        TRUE
```

```
Bonds        18.013154 18.554288 -0.5411342       FALSE
Commodities   4.620061  6.305811 -1.6857500       FALSE
FX           11.361996 14.265355 -2.9033588       FALSE
```

7.5 Dynamic Spillover Index / rolling-sample total volatility spillover

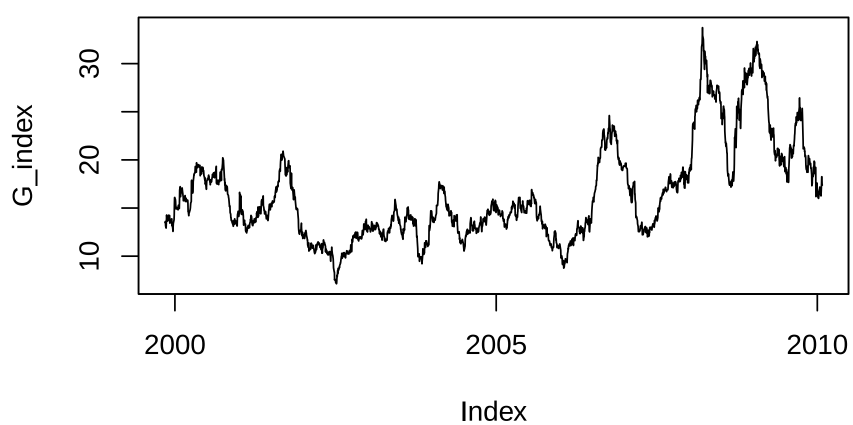
```r
# Data Setting
data(dy2012)
dy2012$Date <- as.Date(dy2012$Date, "%Y-%m-%d")
dy2012 <- as.zoo(dy2012[,-1], order.by = dy2012$Date)
class(dy2012)
```

```
[1] "zoo"
```

```r
# Generalized rolling spillover index based on a VAR(4)
G_index<- total.dynamic.spillover(dy2012, width = 200, index="generalized", p=4)
head(G_index, n=10)
```

```
1999-11-05 1999-11-08 1999-11-09 1999-11-10 1999-11-11 1999-11-12 1999-11-15
  13.50622   13.60646   13.16968   13.04980   12.95939   12.92011   14.27211
1999-11-16 1999-11-17 1999-11-18
  14.04579   13.90963   13.85581
```

```r
plot(G_index)
```



7.6 Directional volatility spillovers

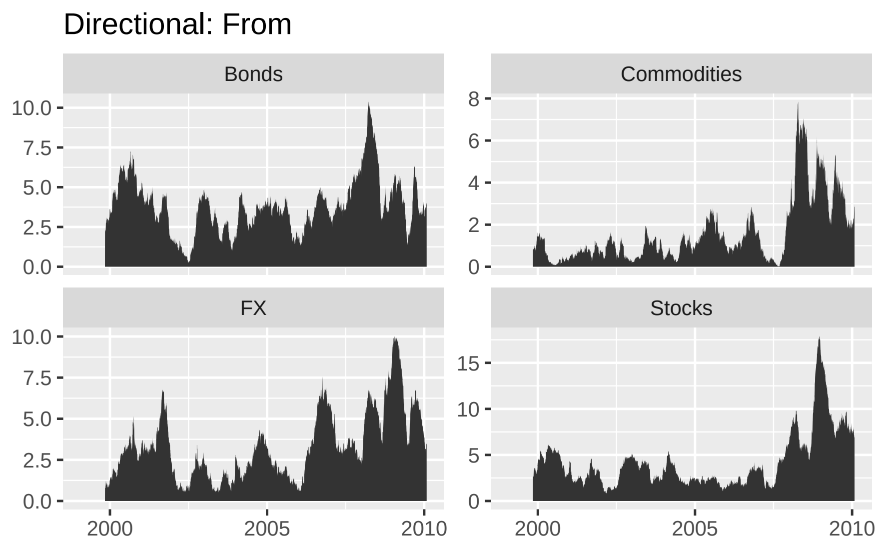
```
library(zoo)
data(dy2012) # re-import data
class(dy2012)
```

```
[1] "data.frame"
```

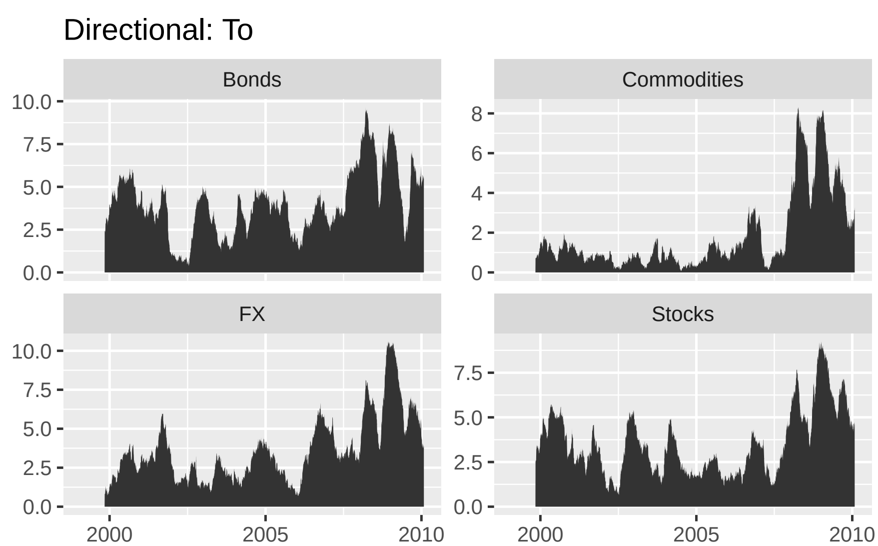
```
dy_results <- dynamic.spillover(dy2012, width=200, remove.own = FALSE)
str(dy_results)
```

```
List of 5
 $ from            :'data.frame':   2771 obs. of  5 variables:
  ..$ Date       : Factor w/ 2771 levels "1999-01-25","1999-01-26",..: 1 2 3 4 5 6 7 8 9 10 ...
  ..$ Stocks     : num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ Bonds      : num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ Commodities: num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ FX         : num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
 $ to              :'data.frame':   2771 obs. of  5 variables:
  ..$ Date       : Factor w/ 2771 levels "1999-01-25","1999-01-26",..: 1 2 3 4 5 6 7 8 9 10 ...
  ..$ Stocks     : num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ Bonds      : num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ Commodities: num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ FX         : num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
 $ net             :'data.frame':   2771 obs. of  5 variables:
  ..$ Date       : Factor w/ 2771 levels "1999-01-25","1999-01-26",..: 1 2 3 4 5 6 7 8 9 10 ...
  ..$ Stocks     : num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ Bonds      : num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ Commodities: num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ FX         : num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
 $ net_pairwise    :'data.frame':   2771 obs. of  7 variables:
  ..$ Date              : Factor w/ 2771 levels "1999-01-25","1999-01-26",..: 1 2 3 4 5 6 7 8 9 1
  ..$ Stocks-Bonds      : num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ Stocks-Commodities: num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ Stocks-FX         : num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ Bonds-Commodities : num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ Bonds-FX          : num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ Commodities-FX    : num [1:2771] 0 0 0 0 0 0 0 0 0 0 ...
 $ from_to_pairwise:'data.frame':   44336 obs. of  3 variables:
  ..$ Date     : Factor w/ 2771 levels "1999-01-25","1999-01-26",..: 1 2 3 4 5 6 7 8 9 10 ...
  ..$ variables: chr [1:44336] "From: Stocks  to: Stocks" "From: Stocks  to: Bonds" "From: Stocks
  ..$ value    : num [1:44336] 0 0 0 0 0 0 0 0 0 0 ...
 - attr(*, "class")= chr "directional.spillover"
```

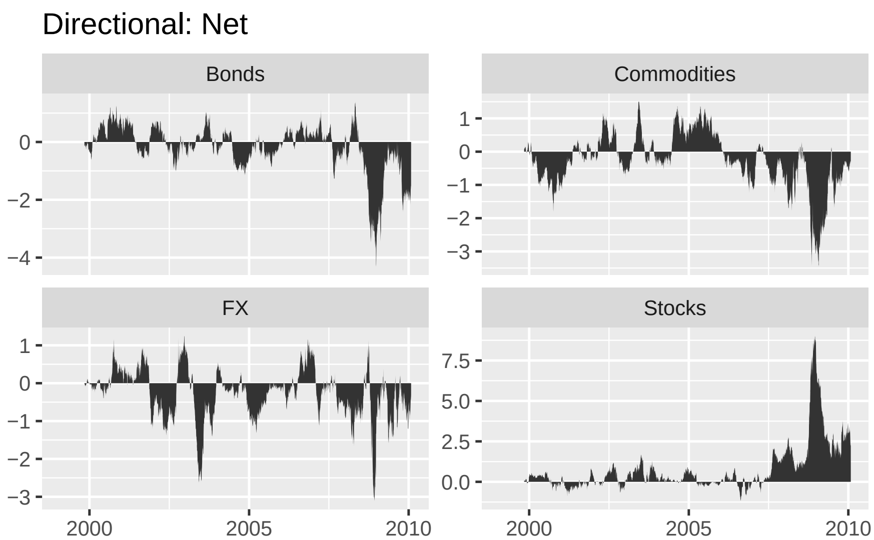
```
# Directional volatility spillovers, FROM four asset classes.
pp_from <- plotdy(dy_results, direction = "from")
```



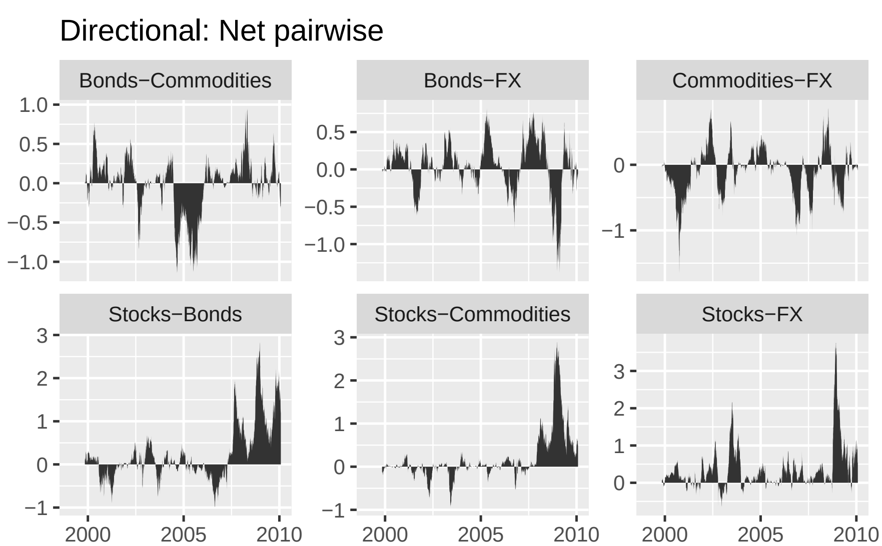

```
# Directional volatility spillovers, TO four asset classes.
pp_to <- plotdy(dy_results, direction = "to")
```



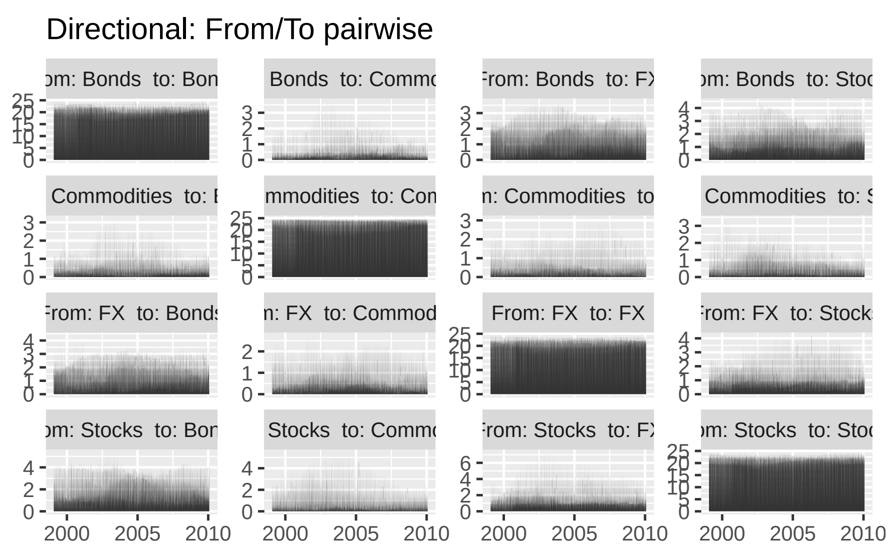

```
# Net volatility spillovers, four asset classes
pp_net <- plotdy(dy_results, direction = "net")
```




```
# Net pairwise volatility spillovers
pp_netpairwise <- plotdy(dy_results, direction = "net_pairwise")
```




```
pp_from_to_pairwise <- plotdy(dy_results, direction = "from_to_pairwise")
```



7.7 Connectedness Network

```
sp <- G.spillover(VAR_4, n.ahead = 10, standardized = F )
datanet <-  Spillover::net(sp)
```

```
Warning in Spillover::net(sp): 'Spillover::net' is deprecated.
Use 'dynamic.spillover' instead.
See help("Deprecated")
```

```
datanet
```

```
                    To      From        Net Transmitter
Stocks       16.373241 11.242998  5.1302430        TRUE
Bonds        18.013154 18.554288 -0.5411342       FALSE
Commodities   4.620061  6.305811 -1.6857500       FALSE
FX           11.361996 14.265355 -2.9033588       FALSE
```

```
# Data frame node
node_df <- data.frame(rownames(datanet), rownames(datanet),datanet$Net)
names(node_df) <- c("id","label","size")
head(node_df)
```

```
           id       label       size
1      Stocks      Stocks  5.1302430
2       Bonds       Bonds -0.5411342
3 Commodities Commodities -1.6857500
4          FX          FX -2.9033588
```

```
sp <- sp[1:4,1:4]
sp
```

```
               Stocks     Bonds Commodities        FX
Stocks      88.757002  7.291185   0.3453279  3.606486
Bonds       10.213545 81.445712   2.7269737  5.613770
Commodities  0.468118  3.695953  93.6941893  2.141740
FX           5.691579  7.026017   1.5477592 85.734645
```

```
# Data frame edge
m1 <- melt(sp)[melt(upper.tri(sp))$value,] # FROM
m2 <- melt(sp)[melt(lower.tri(sp))$value,] # TO
m1 <- m1[order(m1$X1),]
m2 <- m2[order(m2$X2),]
```

```
edge_df <- data.frame("to"=m1[,2],"from"=m1[,1], "weight" = m1$value-m2$value)
library(dplyr)
```

```
Attaching package: 'dplyr'

The following object is masked from 'package:reshape':

    rename

The following objects are masked from 'package:igraph':

    as_data_frame, groups, union

The following object is masked from 'package:MASS':

    select

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

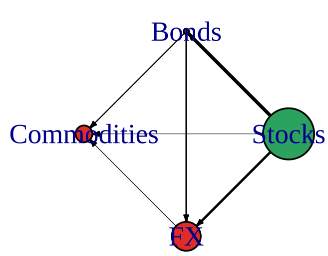
```
edge_df_positive <- edge_df %>% filter(weight >= 0)
edge_df_negative <- edge_df %>% filter(weight < 0)

edge_df_negative <- edge_df_negative %>%
  mutate(weight = -weight) %>%
  dplyr::rename(to = from, from = to)
edge_df <- bind_rows(edge_df_positive, edge_df_negative)
```

```
positive_weight <- edge_df$weight[edge_df$weight > 0]
negative_weight <- edge_df$weight[edge_df$weight < 0]
positive_size <- node_df$size[node_df$size > 0]
negative_size <- node_df$size[node_df$size < 0]

library(RColorBrewer)
Transmitter_color <- "#2ca25f"
else_color <- "#de2d26"
color_vec1 <- ifelse(edge_df$weight > 0, Transmitter_color, else_color)
color_vec2 <- ifelse(node_df$size > 0, Transmitter_color, else_color)
```

```
graph <- graph_from_data_frame(edge_df, directed = TRUE, vertices = node_df)
E(graph)$color <- "black" # Egde
V(graph)$color <- color_vec2 # Node
E(graph)$weight <- abs(edge_df$weight)
V(graph)$size <- abs(node_df$size)
```

```
E(graph)$weight <- E(graph)$weight / max(E(graph)$weight) * 2
V(graph)$size <- V(graph)$size / max(V(graph)$size) * 50
```

```
plot(graph, edge.width = E(graph)$weight, layout=layout_in_circle(graph), edge.arrow.mo
```



Chapter 8

Multivariate GARCH

8.1 DCC-GARCH

```
library(quantmod)
```

```
Loading required package: xts

Loading required package: zoo


Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric

Loading required package: TTR

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

```
# Needed Internet Connection !! alike install packages
# Stock Ticker
stocks <- c("ASII.JK","BBCA.JK")

data_list <- lapply(stocks, function(stock) {
    getSymbols(stock, src = "yahoo", from = "2018-01-01", to="2022-12-31", auto.assign = FALSE)
})
```

```
Warning: ASII.JK contains missing values. Some functions will not work if
objects contain missing values in the middle of the series. Consider using
na.omit(), na.approx(), na.fill(), etc to remove or replace them.
```

```r
# Daily Return
returns <- lapply(data_list, function(data) {
    dailyReturn(Cl(data))
})
```

```
Warning in to_period(xx, period = on.opts[[period]], ...): missing values
removed from data
```

```r
# Combine data
combined_returns <- do.call(merge, returns)
names(combined_returns) <- stocks
combined_returns <- na.omit(combined_returns)
head(combined_returns)
```

```
                ASII.JK     BBCA.JK
2018-01-01  0.000000000 0.000000000
2018-01-02 -0.012048193 0.000000000
2018-01-03 -0.018292683 0.000000000
2018-01-04  0.021739130 0.014840183
2018-01-05  0.009118541 0.001124859
2018-01-08  0.000000000 0.004494382
```

```r
library(rugarch)
```

```
Loading required package: parallel


Attaching package: 'rugarch'

The following object is masked from 'package:stats':

    sigma
```

```r
library(rmgarch)
```

```


Attaching package: 'rmgarch'

The following objects are masked from 'package:xts':

    first, last
```

```r
# GARCH Specification for a Single Asset
unispec <- ugarchspec(mean.model = list(armaOrder = c(0, 0)),
                      variance.model = list(model = "gjrGARCH",
                                            garchOrder = c(1, 1)),
                      distribution.model = "norm")

# Determine the number of assets
n_assets <- ncol(combined_returns)
```

```
# Replicate GARCH Specification for All Assets
garch_spec <- multispec(replicate(n_assets, unispec))

# DCC Model Specification
dcc_spec <- dccspec(uspec = garch_spec,
                    dccOrder = c(1, 1),
                    distribution = "mvnorm")
```

```
## Fit DCC
dcc.fit <- dccfit(dcc_spec,
                  data = combined_returns,
                  fit.control=list(scale=TRUE))
```

```
dcc.fit
```

```
*---------------------------------*
*          DCC GARCH Fit          *
*---------------------------------*

Distribution         :  mvnorm
Model                :  DCC(1,1)
No. Parameters       :  13
[VAR GARCH DCC UncQ] : [0+10+2+1]
No. Series           :  2
No. Obs.             :  1253
Log-Likelihood       :  6804.626
Av.Log-Likelihood    :  5.43

Optimal Parameters
-----------------------------------
                  Estimate  Std. Error   t value Pr(>|t|)
[ASII.JK].mu     -0.000439    0.000538  -0.81597 0.414515
[ASII.JK].omega   0.000013    0.000001  12.73770 0.000000
[ASII.JK].alpha1  0.029271    0.009480   3.08756 0.002018
[ASII.JK].beta1   0.912943    0.007720 118.25080 0.000000
[ASII.JK].gamma1  0.056569    0.023773   2.37952 0.017335
[BBCA.JK].mu      0.000635    0.000531   1.19598 0.231706
[BBCA.JK].omega   0.000015    0.000008   1.90315 0.057021
[BBCA.JK].alpha1  0.053503    0.030192   1.77212 0.076374
[BBCA.JK].beta1   0.820666    0.052704  15.57124 0.000000
[BBCA.JK].gamma1  0.127614    0.044277   2.88221 0.003949
[Joint]dcca1      0.050241    0.033824   1.48539 0.137441
[Joint]dccb1      0.766227    0.197276   3.88403 0.000103

Information Criteria
```

```
---------------------

Akaike       -10.841
Bayes        -10.787
Shibata      -10.841
Hannan-Quinn -10.821


Elapsed time : 2.838826
```

```r
# Conditional Covariances
cov <- rcov(dcc.fit)
dim(cov)
```

```
[1]    2    2 1253
```

```r
cov[,,1:4]
```

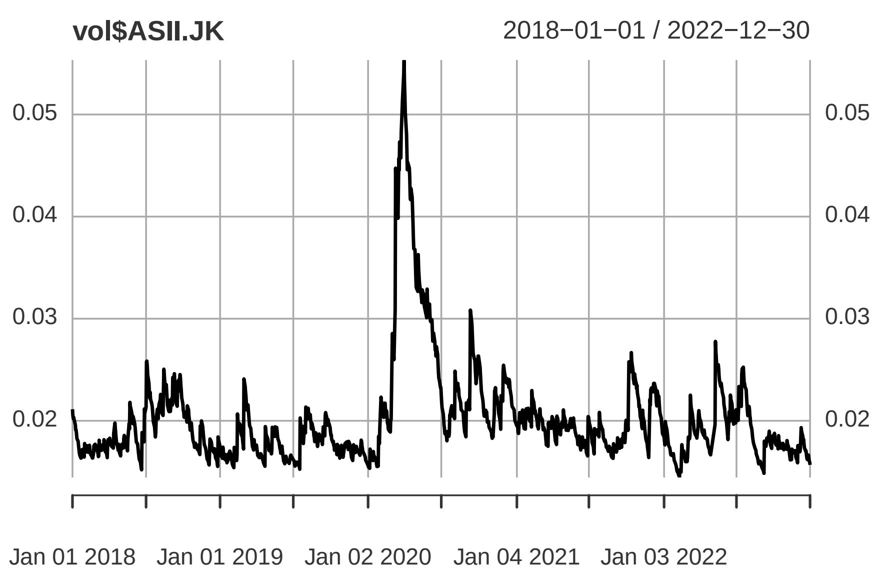
```
, , 2018-01-01

             ASII.JK      BBCA.JK
ASII.JK 0.0004465808 0.0001387231
BBCA.JK 0.0001387231 0.0002527424

, , 2018-01-02

             ASII.JK      BBCA.JK
ASII.JK 0.0004207083 0.0001246004
BBCA.JK 0.0001246004 0.0002229686

, , 2018-01-03

             ASII.JK      BBCA.JK
ASII.JK 0.0004086508 0.0001138079
BBCA.JK 0.0001138079 0.0001985343

, , 2018-01-04

             ASII.JK      BBCA.JK
ASII.JK 0.0004134356 0.0001058398
BBCA.JK 0.0001058398 0.0001784819
```

```r
# Conditional Volatilities
vol <- sigma(dcc.fit)
head(vol)
```

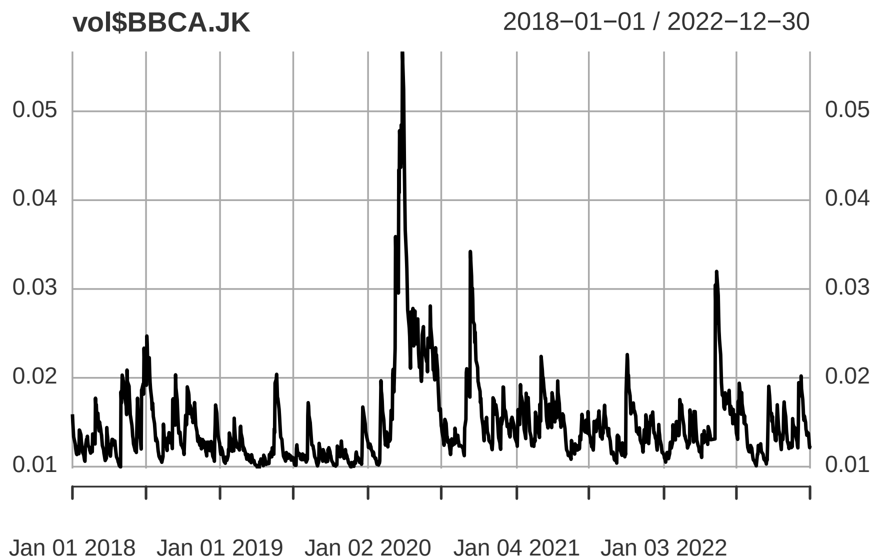
```
              ASII.JK    BBCA.JK
2018-01-01 0.02113246 0.01589787
```

```
2018-01-02 0.02051117 0.01493213
2018-01-03 0.02021511 0.01409022
2018-01-04 0.02033311 0.01335971
2018-01-05 0.02012066 0.01314337
2018-01-08 0.01962831 0.01254032
```

```
plot(vol$ASII.JK)
```



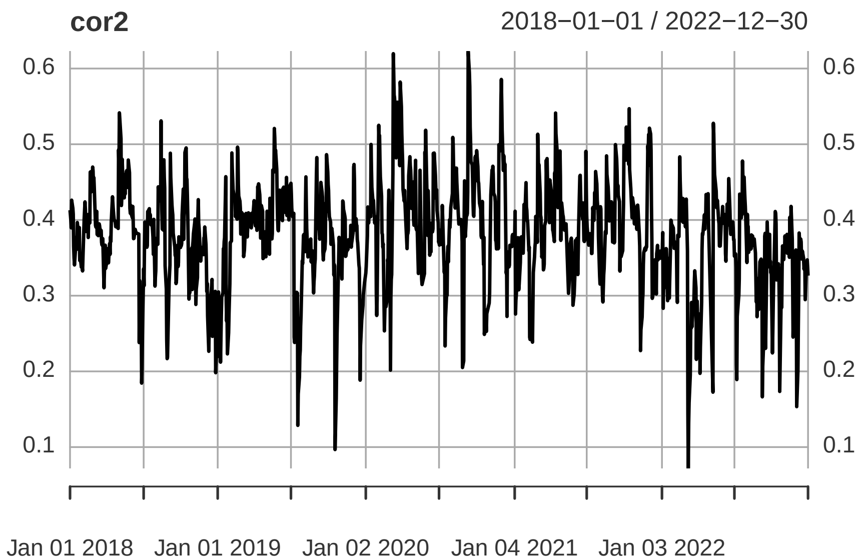

```
plot(vol$BBCA.JK)
```




```r
# Conditional Correlations
cor <- rcor(dcc.fit)
cor[,,1:4]
```

```
, , 2018-01-01

          ASII.JK   BBCA.JK
ASII.JK 1.0000000 0.4129141
BBCA.JK 0.4129141 1.0000000

, , 2018-01-02

          ASII.JK   BBCA.JK
ASII.JK 1.0000000 0.4068245
BBCA.JK 0.4068245 1.0000000

, , 2018-01-03

          ASII.JK   BBCA.JK
ASII.JK 1.0000000 0.3995568
BBCA.JK 0.3995568 1.0000000

, , 2018-01-04

          ASII.JK   BBCA.JK
ASII.JK 1.0000000 0.3896261
BBCA.JK 0.3896261 1.0000000
```

```r
date <- row.names(data.frame(cor[1,1,]))
cor2 <- xts(cor[1, 2, ], order.by = as.Date(date))
plot(cor2)
```




```r
forecast <- dccforecast(dcc.fit, n.ahead = 5)
```

```r
forecast@mforecast$H #Cov
```

```
[[1]]
, , 1

             [,1]         [,2]
[1,] 2.400993e-04 5.997351e-05
[2,] 5.997351e-05 1.426083e-04

, , 2

             [,1]         [,2]
[1,] 2.460159e-04 6.413044e-05
[2,] 6.413044e-05 1.492416e-04

, , 3

             [,1]         [,2]
[1,] 2.517579e-04 6.792145e-05
[2,] 6.792145e-05 1.554635e-04
```

```
, , 4

             [,1]         [,2]
[1,] 2.573305e-04 7.138253e-05
[2,] 7.138253e-05 1.612995e-04

, , 5

             [,1]         [,2]
[1,] 2.627387e-04 7.454746e-05
[2,] 7.454746e-05 1.667735e-04
```

```
forecast@mforecast$R #Cor
```

```
[[1]]
, , 1

          [,1]      [,2]
[1,] 1.0000000 0.3241094
[2,] 0.3241094 1.0000000

, , 2

          [,1]      [,2]
[1,] 1.0000000 0.3346861
[2,] 0.3346861 1.0000000

, , 3

          [,1]      [,2]
[1,] 1.0000000 0.3433217
[2,] 0.3433217 1.0000000

, , 4

          [,1]      [,2]
[1,] 1.0000000 0.3503723
[2,] 0.3503723 1.0000000

, , 5

          [,1]      [,2]
[1,] 1.0000000 0.3561289
[2,] 0.3561289 1.0000000
```

```
forecast@mforecast$mu
```

```
, , 1

             [,1]         [,2]
[1,] -0.0004392902 0.0006354097
[2,] -0.0004392902 0.0006354097
[3,] -0.0004392902 0.0006354097
[4,] -0.0004392902 0.0006354097
[5,] -0.0004392902 0.0006354097
```

Chapter 9

NARDL

```r
# install.packages("ardl.nardl")
library(ardl.nardl)
```

```
Warning: package 'ardl.nardl' was built under R version 4.4.3
```

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

```r
# Data
datanardl <- read.csv("Data/Bab 9/datanardl.csv")
head(datanardl)
```

```
      date price.Vietnam price.China
1 1/1/2002           115    143.7978
2 2/1/2002           105    129.4568
3 3/1/2002           100    127.9081
4 4/1/2002           117    149.8740
5 5/1/2002           103    131.0987
6 6/1/2002           113    145.1878
```

```r
# Phillips-Perron Unit Root Test
PP.test(datanardl$price.Vietnam)
```

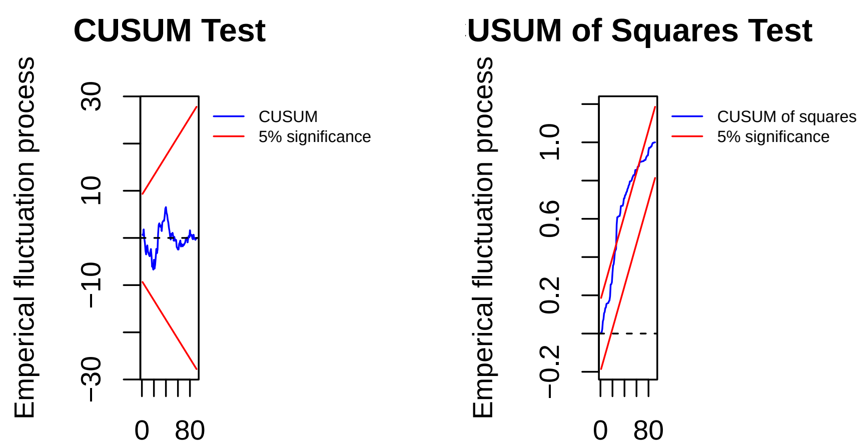
```

    Phillips-Perron Unit Root Test

data:  datanardl$price.Vietnam
Dickey-Fuller = -5.6063, Truncation lag parameter = 3, p-value = 0.01
```

```r
PP.test(datanardl$price.China)
```

```
    Phillips-Perron Unit Root Test

data:  datanardl$price.China
Dickey-Fuller = -5.0955, Truncation lag parameter = 3, p-value = 0.01
```

```
# Auto Selecet Model
model1 <- gets_nardl_uecm(x =  datanardl,
                          decomp = 'price.China',
                          dep_var = 'price.Vietnam',
                          p_order = c(5),
                          q_order = c(5),
                          graph_save = TRUE,
                          case = 3,
                          F_HC = TRUE)
```

```
Percentage of positive changes in decomp is 56 percent while negative change is 44
```




```
# Cointegratio Test
model1$cointegration$Fstat
```

```
                   observation k    fstat case lower.b upper.b
10% critical value          91 1 10.88198    3    4.04    4.78
5% critical value           91 1 10.88198    3    4.94    5.73
1% critical value           91 1 10.88198    3    6.84    7.84
```

```
# NARDL Form
summary(model1$Parsimonious_NARDL_fit)
```

```
Call:
lm(formula = price.Vietnam ~ price.Vietnam_1 + price.Vietnam_3 +
    price.Vietnam_4 + price.Vietnam_5 + price.China_pos + price.China_pos_1 +
    price.China_pos_2 + price.China_pos_4 + price.China_pos_5 +
    price.China_neg + price.China_neg_1 + price.China_neg_4 +
    price.China_neg_5, na.action = na.exclude)

Residuals:
     Min       1Q   Median       3Q      Max
-16.3047  -4.4665  -0.3095   4.0691  21.2637

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       66.70011   14.83287   4.497 2.40e-05 ***
price.Vietnam_1    0.52746    0.09462   5.575 3.52e-07 ***
price.Vietnam_3    0.14052    0.10138   1.386  0.16974
price.Vietnam_4   -0.03782    0.10453  -0.362  0.71847
price.Vietnam_5   -0.14259    0.09817  -1.452  0.15043
price.China_pos    0.08230    0.18191   0.452  0.65223
price.China_pos_1 -0.28087    0.22492  -1.249  0.21553
price.China_pos_2  0.19196    0.17400   1.103  0.27338
price.China_pos_4 -0.26694    0.19885  -1.342  0.18341
price.China_pos_5  0.25245    0.16491   1.531  0.12991
price.China_neg    0.04123    0.15263   0.270  0.78778
price.China_neg_1  0.05460    0.17787   0.307  0.75969
price.China_neg_4 -0.50597    0.17240  -2.935  0.00440 **
price.China_neg_5  0.41552    0.15347   2.708  0.00835 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7.061 on 77 degrees of freedom
Multiple R-squared:  0.5657,    Adjusted R-squared:  0.4923
F-statistic: 7.714 on 13 and 77 DF,  p-value: 1.571e-09
```

```r
# NARDL ECM Form
summary(model1$Parsimonious_ECM_fit)
```

```
Call:
lm(formula = D.price.Vietnam ~ price.Vietnam_1 + price.China_pos_1 +
    price.China_neg_1 + D.price.Vietnam_2 + D.price.Vietnam_3 +
    D.price.Vietnam_4 + D.price.China_neg_4, na.action = na.exclude)

Residuals:
     Min       1Q   Median       3Q      Max
```

```
-16.0510  -5.0380  -0.3461   3.9997  23.1158

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)         61.8968384 11.5905403   5.340 7.95e-07 ***
price.Vietnam_1     -0.5138343  0.0903867  -5.685 1.90e-07 ***
price.China_pos_1   -0.0192273  0.0771006  -0.249   0.8037
price.China_neg_1    0.0003975  0.0741449   0.005   0.9957
D.price.Vietnam_2    0.0790419  0.0929207   0.851   0.3974
D.price.Vietnam_3    0.2409279  0.0963186   2.501   0.0143 *
D.price.Vietnam_4    0.1413008  0.0922292   1.532   0.1293
D.price.China_neg_4 -0.5632653  0.1217687  -4.626 1.36e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7.093 on 83 degrees of freedom
Multiple R-squared:  0.4579,    Adjusted R-squared:  0.4122
F-statistic: 10.01 on 7 and 83 DF,  p-value: 5.192e-09
```

```
# Long Run Coefficients
model1$Longrun_relation
```

```
                       Estimate Std. Error      t value  Pr(>|t|)
price.China_pos_1 -0.0374192226  0.1505206 -0.248598606 0.8042781
price.China_neg_1  0.0007735328  0.1442840  0.005361181 0.9957351
```

```
# Long Run Asymetric Test
model1$longrun_asym
```

```
               Fstat       Pval
price.China 5.031109 0.02756084
```

```
# Short Run Asymetric Test
model1$Shortrun_asym
```

```
               Fstat         Pval
price.China 21.39708 1.359712e-05
```