Time Series

1. Bina objek masa dan tarikh dalam R

```
set.seed(123)
data1 = rnorm(12)
```

Takrifkan unit masa terhadap data

1.1 Data ialah data bulanan bermula dari Februrari 2020

```
X1 = ts(data1, start=c(2020,2), frequency=12)
X1
```

```
##              Jan         Feb         Mar         Apr         May         Jun
## 2020             -0.56047565 -0.23017749  1.55870831  0.07050839  0.12928774
## 2021  0.35981383
##              Jul         Aug         Sep         Oct         Nov         Dec
## 2020  1.71506499  0.46091621 -1.26506123 -0.68685285 -0.44566197  1.22408180
## 2021
```

1.2 Data ialah data suku tahunan bermula dari suku ketiga tahun 2020

```
X2 = ts(data1, start=c(2020,3), frequency=4)
X2
```

```
##             Qtr1        Qtr2        Qtr3        Qtr4
## 2020                         -0.56047565 -0.23017749
## 2021  1.55870831  0.07050839  0.12928774  1.71506499
## 2022  0.46091621 -1.26506123 -0.68685285 -0.44566197
## 2023  1.22408180  0.35981383
```

1.3 R mengadaptasi format masa data ISO 8601

```
date = Sys.Date()
date
```

```
## [1] "2024-12-15"
```

1.4 Bina tarikh harian bermula dari 2016-01-01 hingga 2018-12-31

```
daily_index = seq.Date(from=as.Date('2016-01-01'), to=as.Date('2018-12-31'), by='day')
head(daily_index,10)
```

```
##  [1] "2016-01-01" "2016-01-02" "2016-01-03" "2016-01-04" "2016-01-05"
##  [6] "2016-01-06" "2016-01-07" "2016-01-08" "2016-01-09" "2016-01-10"
```

1.5 Bina tarikh 3 hari selang bermula dari 2016-01-01 hingga 2018-12-31

```
daily_3day = seq.Date(from=as.Date('2016-01-01'), to=as.Date('2018-12-31'), by='3 days')
head(daily_3day,10)
```

```
##  [1] "2016-01-01" "2016-01-04" "2016-01-07" "2016-01-10" "2016-01-13"
##  [6] "2016-01-16" "2016-01-19" "2016-01-22" "2016-01-25" "2016-01-28"
```

1.6 Bina tarikh bulanan bermula dari 2016-01-01 hingga 2018-12-31

```
monthly_index = seq.Date(from=as.Date('2016-01-01'), to=as.Date('2018-12-31'), by='month')
head(monthly_index,10)
```

```
##  [1] "2016-01-01" "2016-02-01" "2016-03-01" "2016-04-01" "2016-05-01"
##  [6] "2016-06-01" "2016-07-01" "2016-08-01" "2016-09-01" "2016-10-01"
```

Exercise

```
dates_df = read.csv("E:/Master-Data-Science/Semester_1/Data_Mining/Data/dates_formats3.csv", header=T, s
head(dates_df,10)
```

```
##    Japanese_format US_format    CA_mix_format   SA_mix_format  NZ_format
## 1       20/01/2017 1/20/2017 January 20, 2017 20 January 2017 20/01/2017
## 2       21/01/2017 1/21/2017 January 21, 2017 21 January 2017 21/01/2017
## 3       22/01/2017 1/22/2017 January 22, 2017 22 January 2017 22/01/2017
## 4       23/01/2017 1/23/2017 January 23, 2017 23 January 2017 23/01/2017
## 5       24/01/2017 1/24/2017 January 24, 2017 24 January 2017 24/01/2017
## 6       25/01/2017 1/25/2017 January 25, 2017 25 January 2017 25/01/2017
## 7       26/01/2017 1/26/2017 January 26, 2017 26 January 2017 26/01/2017
## 8       27/01/2017 1/27/2017 January 27, 2017 27 January 2017 27/01/2017
## 9       28/01/2017 1/28/2017 January 28, 2017 28 January 2017 28/01/2017
## 10      29/01/2017 1/29/2017 January 29, 2017 29 January 2017 29/01/2017
```

```
str(dates_df)
```

```
## 'data.frame':    22 obs. of  5 variables:
##  $ Japanese_format: chr  "20/01/2017" "21/01/2017" "22/01/2017" "23/01/2017" ...
##  $ US_format      : chr  "1/20/2017" "1/21/2017" "1/22/2017" "1/23/2017" ...
##  $ CA_mix_format  : chr  "January 20, 2017" "January 21, 2017" "January 22, 2017" "January 23, 2017"
##  $ SA_mix_format  : chr  "20 January 2017" "21 January 2017" "22 January 2017" "23 January 2017" ...
##  $ NZ_format      : chr  "20/01/2017" "21/01/2017" "22/01/2017" "23/01/2017" ...
```

Reformat into ISO

```r
US_format_new = as.Date(dates_df$US_format, format = "%m/%d/%Y")
str(US_format_new,10)
```

US Format

```
##  Date[1:22], format: "2017-01-20" "2017-01-21" "2017-01-22" "2017-01-23" "2017-01-24" ...
```

```r
CA_mix_format_new = as.Date(dates_df$CA_mix_format, format = "%B %d, %Y")
str(CA_mix_format_new)
```

CA_Mix_Format

```
##  Date[1:22], format: "2017-01-20" "2017-01-21" "2017-01-22" "2017-01-23" "2017-01-24" ...
```

```r
Japanese_format_new = as.Date(dates_df$Japanese_format, format = "%d/%m/%Y")
str(Japanese_format_new)
```

Japanese_format

```
##  Date[1:22], format: "2017-01-20" "2017-01-21" "2017-01-22" "2017-01-23" "2017-01-24" ...
```

```r
SA_mix_format_new = as.Date(dates_df$SA_mix_format, format = "%d %B %Y")
str(SA_mix_format_new)
```

SA_mix_format

```
##  Date[1:22], format: "2017-01-20" "2017-01-21" "2017-01-22" "2017-01-23" "2017-01-24" ...
```

```
NZ_format_new = as.Date(dates_df$NZ_format, format="%d/%m/%Y")
str(NZ_format_new)
```

NZ_format

```
##  Date[1:22], format: "2017-01-20" "2017-01-21" "2017-01-22" "2017-01-23" "2017-01-24" ...
```

```
new_df = data.frame(cbind(Japanese_format_new,US_format_new,CA_mix_format_new,SA_mix_format_new, NZ_forn
head(new_df)
```

```
##   Japanese_format_new US_format_new CA_mix_format_new SA_mix_format_new
## 1               17186         17186             17186             17186
## 2               17187         17187             17187             17187
## 3               17188         17188             17188             17188
## 4               17189         17189             17189             17189
## 5               17190         17190             17190             17190
## 6               17191         17191             17191             17191
##   NZ_format_new
## 1         17186
## 2         17187
## 3         17188
## 4         17189
## 5         17190
## 6         17191
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

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

```
date_df = as.Date(as.character(new_df), format = '%Y%m%d')
date_df
```

```
## [1] NA NA NA NA NA
```

2. Kelas data siri masa dalam

2.1 Kelas TS

```
X1 = ts(data1, start=c(2020,2), frequency=12)
X1
```

```
##              Jan         Feb         Mar         Apr         May         Jun
## 2020             -0.56047565 -0.23017749  1.55870831  0.07050839  0.12928774
## 2021  0.35981383
##              Jul         Aug         Sep         Oct         Nov         Dec
## 2020  1.71506499  0.46091621 -1.26506123 -0.68685285 -0.44566197  1.22408180
## 2021
```

2.2 Kelas TimeSeries

```r
library(timeSeries)
```

```
## Loading required package: timeDate
```

```
##
## Attaching package: 'timeSeries'
```

```
## The following objects are masked from 'package:graphics':
##
##     lines, points
```

```r
data(MSFT)
class(MSFT)
```

```
## [1] "timeSeries"
## attr(,"package")
## [1] "timeSeries"
```

2.3 Kelas Zoo

```r
library(TSstudio)
library(zoo)
```

```
##
## Attaching package: 'zoo'
```

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

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

```r
data(US_indicators)
str(US_indicators)
```

```
## 'data.frame':	528 obs. of  3 variables:
##  $ Date             : Date, format: "1976-01-31" "1976-02-29" ...
##  $ Vehicle Sales    : num  885 995 1244 1191 1203 ...
##  $ Unemployment Rate: num  8.8 8.7 8.1 7.4 6.8 8 7.8 7.6 7.4 7.2 ...
```

```
Vehicle_Sale1 = zoo(x=US_indicators$`Vehicle Sales`, frequency=12)
str(Vehicle_Sale1)
```

```
## 'zooreg' series from Jan 0001 to Jan 0528
##   Data: num [1:528] 885 995 1244 1191 1203 ...
##   Index:  'yearmon' num [1:528] Jan 0001 Jan 0002 Jan 0003 Jan 0004 ...
##   Frequency: 12
```

2.4 Kelas Date

```
NZ_format_new = as.Date(dates_df$NZ_format, format="%d/%m/%Y")
str(NZ_format_new)
```

```
##  Date[1:22], format: "2017-01-20" "2017-01-21" "2017-01-22" "2017-01-23" "2017-01-24" ...
```

2.5 Kelas Xts

```
library(xts)
US_indicators2 = xts(x=US_indicators[, c('Vehicle Sales','Unemployment Rate')],
                     frequency=12, order=US_indicators$Date)
str(US_indicators2)
```

```
## An xts object on 1976-01-31 / 2019-12-31 containing:
##   Data:    double [528, 2]
##   Columns: Vehicle Sales, Unemployment Rate
##   Index:   Date [528] (TZ: "UTC")
```

2.6 Kelas POSIX

```
time_str = "2018-12-31 23:59:30"
time_POSIX = as.POSIXct(time_str)
time_POSIX
```

```
## [1] "2018-12-31 23:59:30 +08"
```

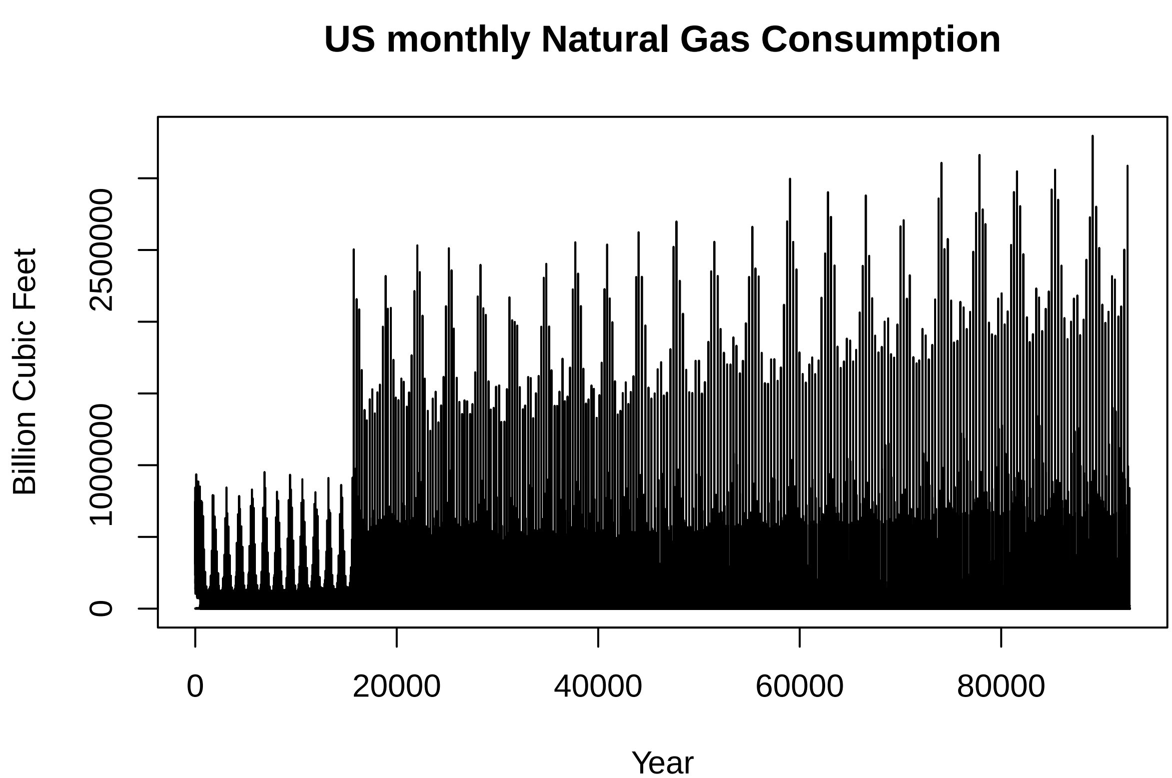
2. Kaedah Penguraian Siri Masa

2.1 Penguraian Bertambah

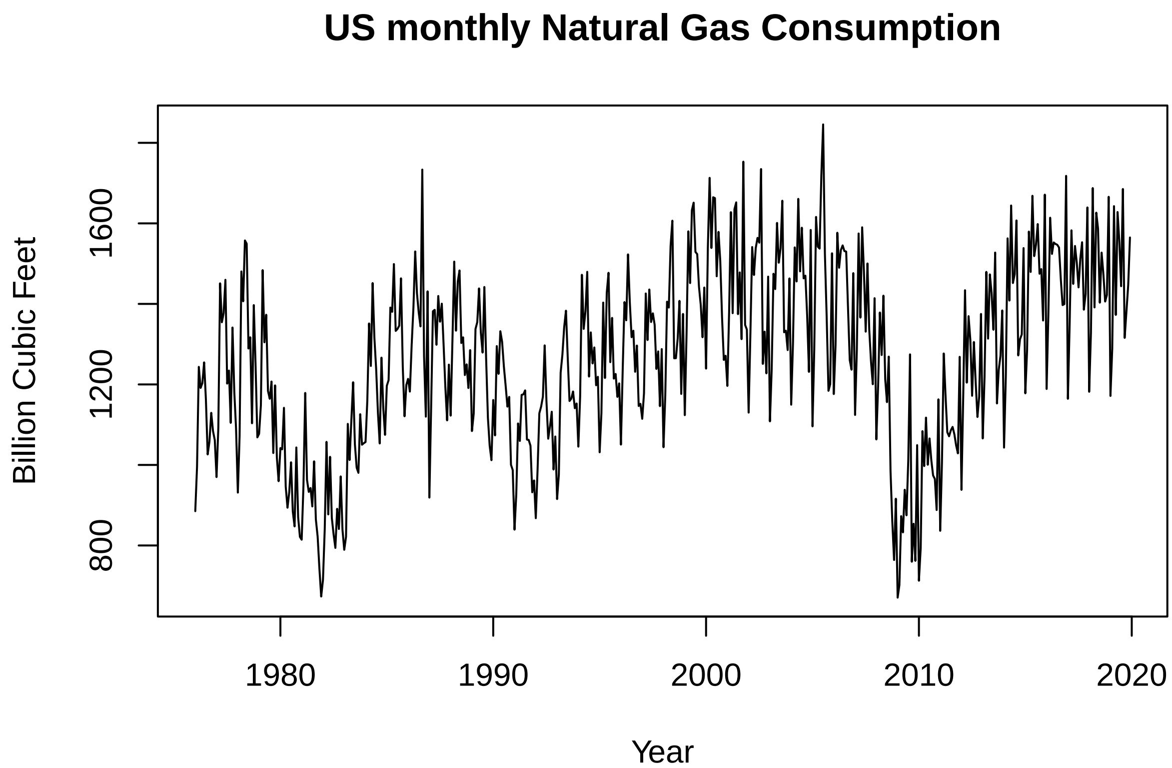
```r
library(USgas)
data("usgas")
head(usgas)
```

```
##         date                 process state state_abb      y
## 1 1973-01-01  Commercial Consumption  U.S.      U.S. 392315
## 2 1973-01-01 Residential Consumption  U.S.      U.S. 843900
## 3 1973-02-01  Commercial Consumption  U.S.      U.S. 394281
## 4 1973-02-01 Residential Consumption  U.S.      U.S. 747331
## 5 1973-03-01  Commercial Consumption  U.S.      U.S. 310799
## 6 1973-03-01 Residential Consumption  U.S.      U.S. 648504
```

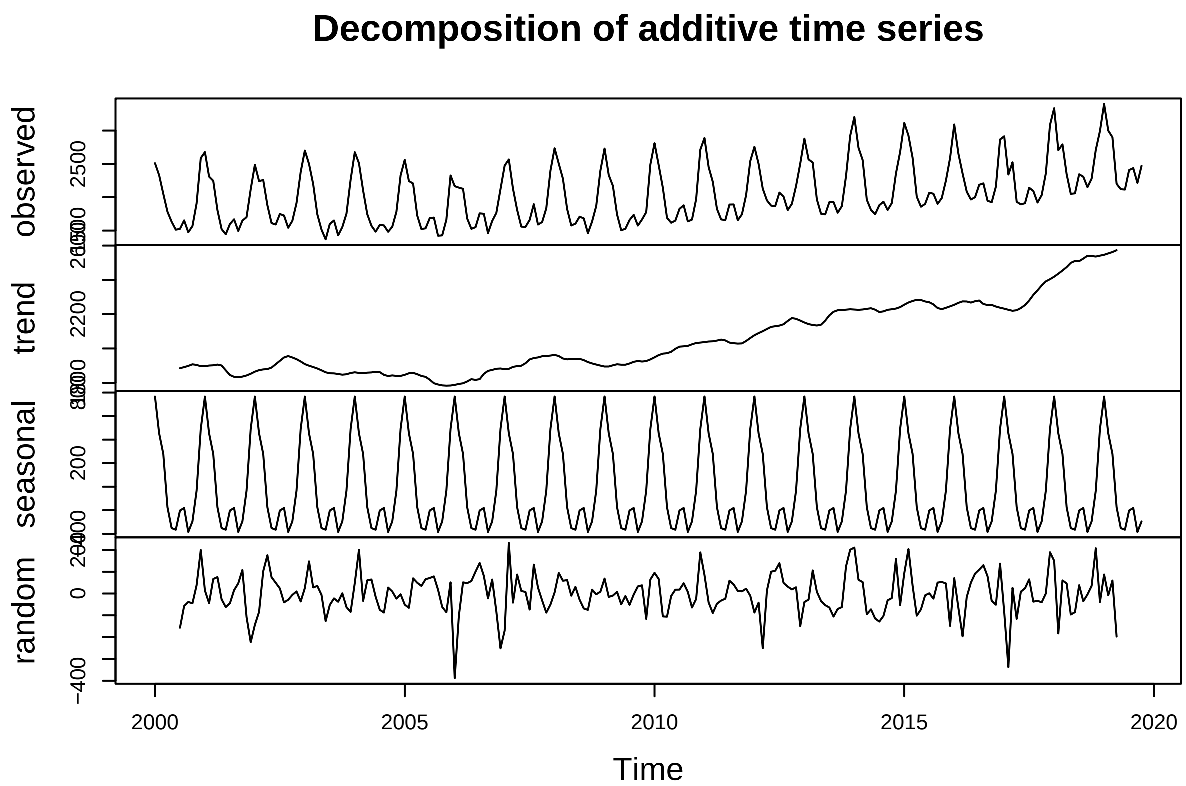
```r
ts.plot(usgas,
        main="US monthly Natural Gas Consumption",
        ylab="Billion Cubic Feet",
        xlab='Year')
```




```r
library(TSstudio)
data(USVSales)
ts.plot(USVSales,
        main="US monthly Natural Gas Consumption",
        ylab="Billion Cubic Feet",
        xlab='Year')
```




```
usgas.decompose = decompose(USgas)
plot(usgas.decompose)
```



boleh ekstrak data setiap komponen untuk analisis lanjut

```
names(usgas.decompose)
```

```
## [1] "x"        "seasonal" "trend"    "random"   "figure"   "type"
```

Komponen Trend

```
head(usgas.decompose$trend)
```

```
## [1] NA NA NA NA NA NA
```

Komponen Bermusim

```
head(usgas.decompose$seasonal)
```

```
## [1]  766.3961  453.1952  278.1884 -174.3087 -351.8246 -365.9909
```

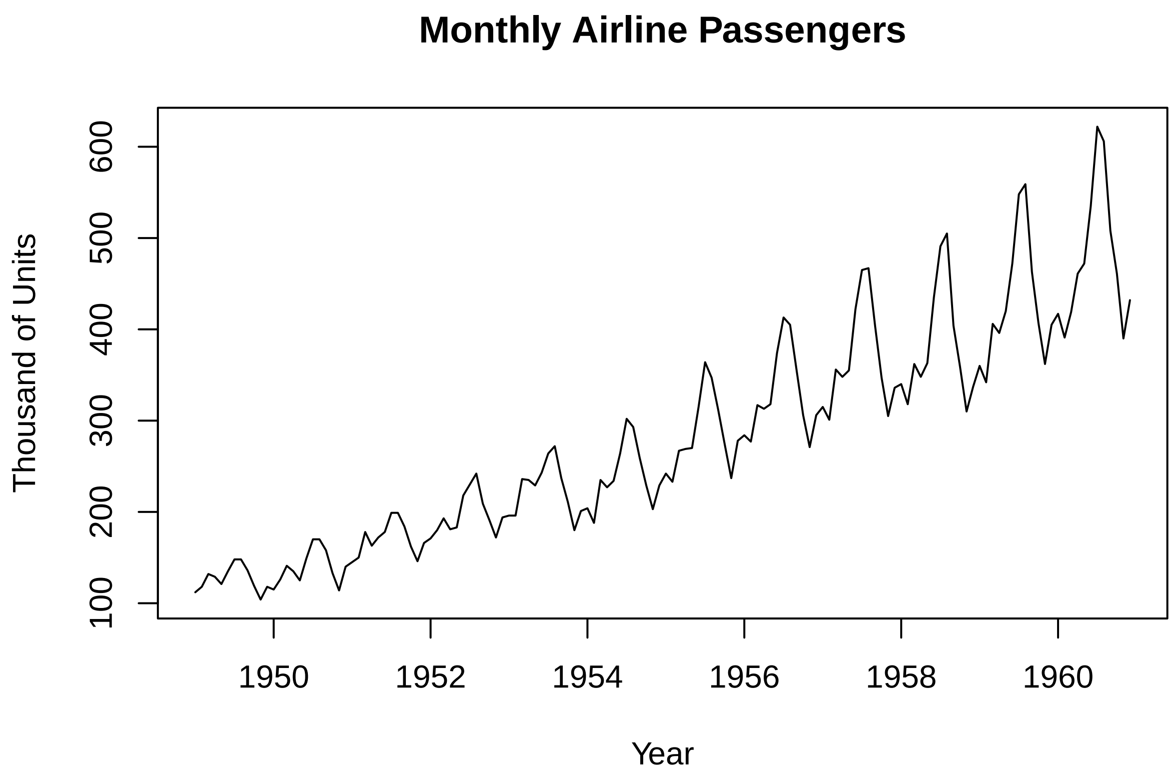
Komponen Rawak

```r
head(usgas.decompose$random)
```

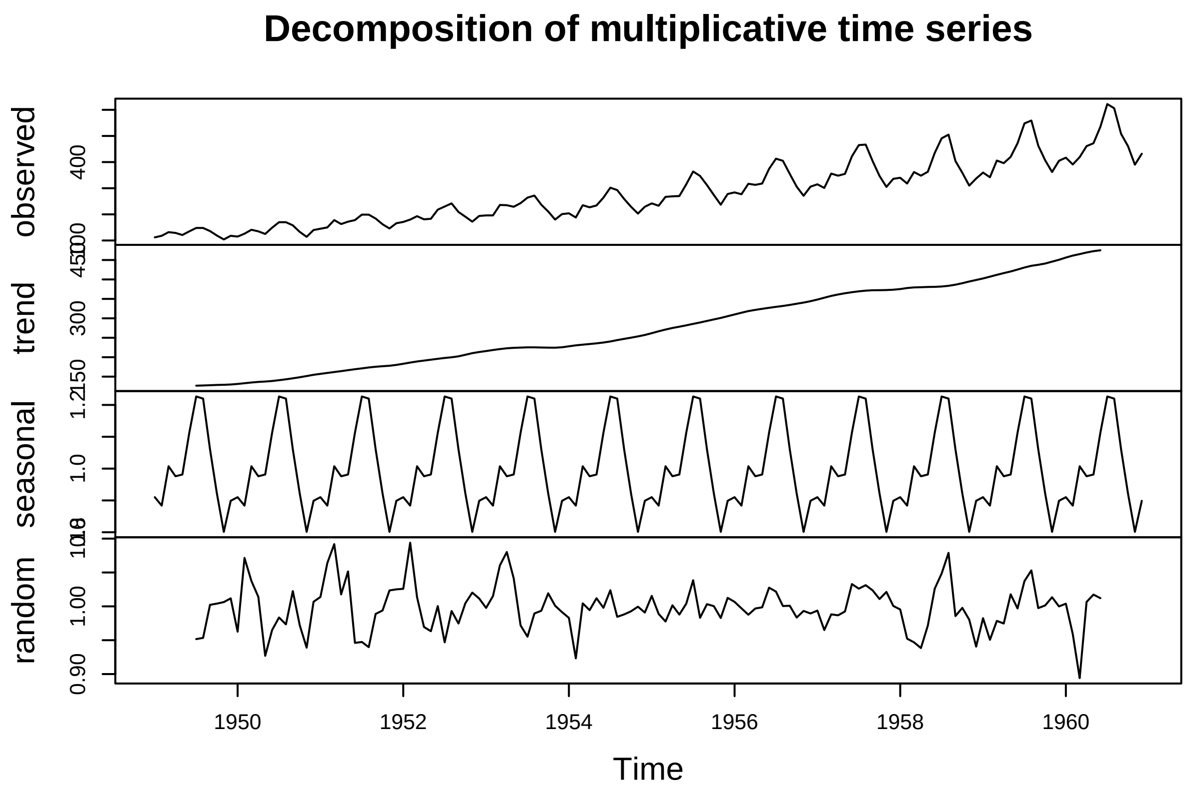
```
## [1] NA NA NA NA NA NA
```

2.2 Penguraian Berganda

```r
data(AirPassengers)
ts.plot(AirPassengers,
        main="Monthly Airline Passengers",
        ylab="Thousand of Units",
        xlab='Year')
```




```r
AirP_decompose = decompose(AirPassengers, type='multiplicative')
plot(AirP_decompose)
```



Komponen Trend

```
head(AirP_decompose$trend,10)
```

```
##  [1]       NA       NA       NA       NA       NA       NA 126.7917 127.2500
##  [9] 127.9583 128.5833
```

Komponen Bermusim

```
head(AirP_decompose$seasonal,10)
```

```
##  [1] 0.9102304 0.8836253 1.0073663 0.9759060 0.9813780 1.1127758 1.2265555
##  [8] 1.2199110 1.0604919 0.9217572
```

Komponen Rawak

```
head(AirP_decompose$random,10)
```

```
##  [1]        NA        NA        NA        NA        NA        NA 0.9516643
##  [8] 0.9534014 1.0022198 1.0040278
```

2.2.3. ARIMA Modelling

(1) Model Autoregresif peringkat *p*, AR(*p*):

$$y_t = \delta + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t,$$

- y_t bergantung kepada p nilai-nilai cerapan yang lepas.

(2) Model Purata Bergerak peringkat *q*, MA(*q*):

$$y_t = \delta + \varepsilon_t - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \cdots - \theta_q \varepsilon_{t-q},$$

- y_t bergantung kepada nilai-nilai q sebutan reja-reja yang lepas.

Kepegunan Siri Masa

$$1.\ \sum(y_t) = u_t,\ untuk\ semua\ t$$

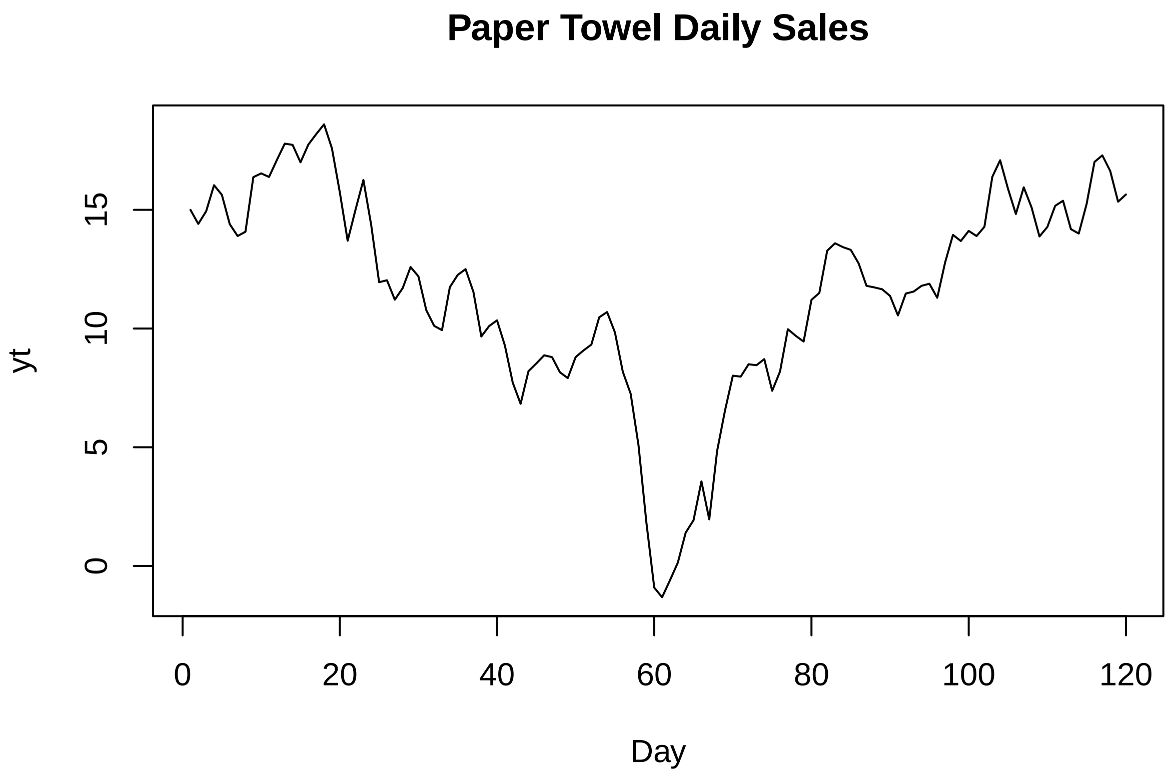
$$2.\ Var(y_t) = \sum[(y_t - u_y)^2]\ \ =\ \ \sigma^2$$

$$Cov(y_t, y_{t-k})\ \ =\ \ \gamma_k,\ untuk\ semua\ t$$

```
data = read.csv("E:/Master-Data-Science/Semester_1/Data_Mining/Data/towel.csv", header=T)
yt=ts(data)
head(yt,10)
```

```
##             y
##  [1,] 15.0000
##  [2,] 14.4064
##  [3,] 14.9383
##  [4,] 16.0374
##  [5,] 15.6320
##  [6,] 14.3975
##  [7,] 13.8959
##  [8,] 14.0765
##  [9,] 16.3750
## [10,] 16.5342
```

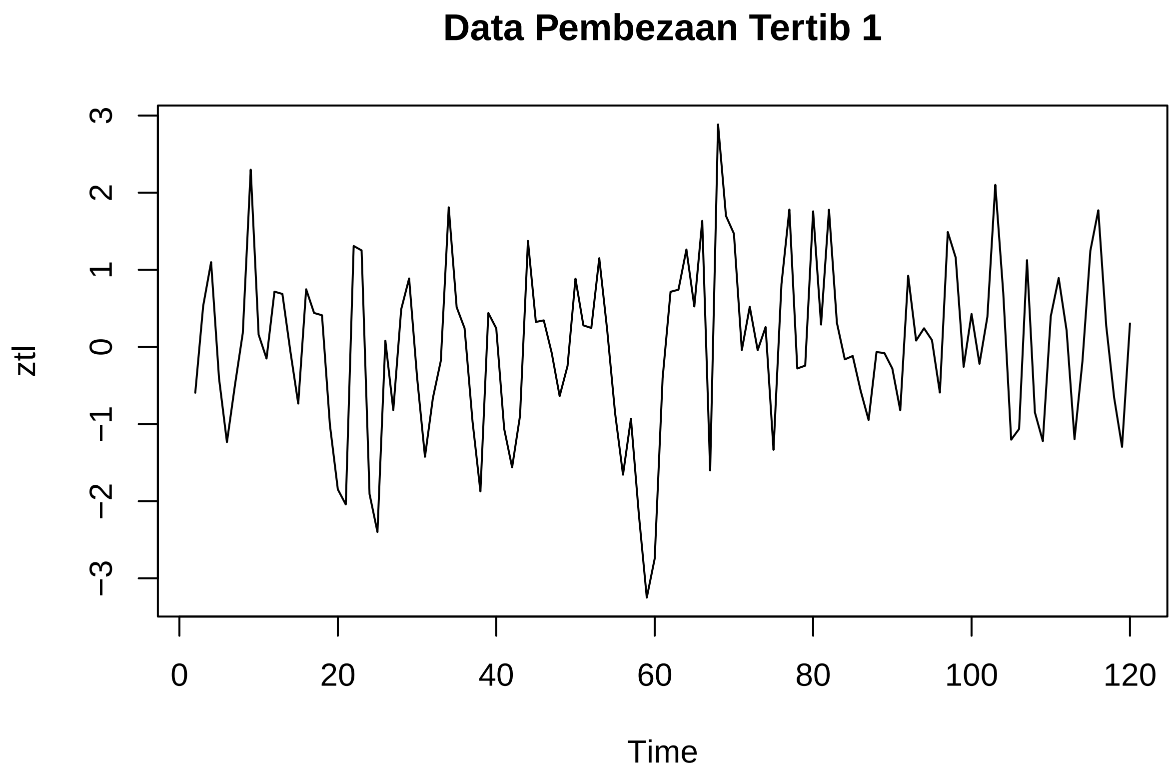
```
ts.plot(yt,
        main="Paper Towel Daily Sales",
        ylab="yt",
        xlab='Day')
```



Data tak pegun, keputusan tak tepat

Jalankan pembezaan terhadap data unntuk jadikan data menghampiri sifat kepegunan

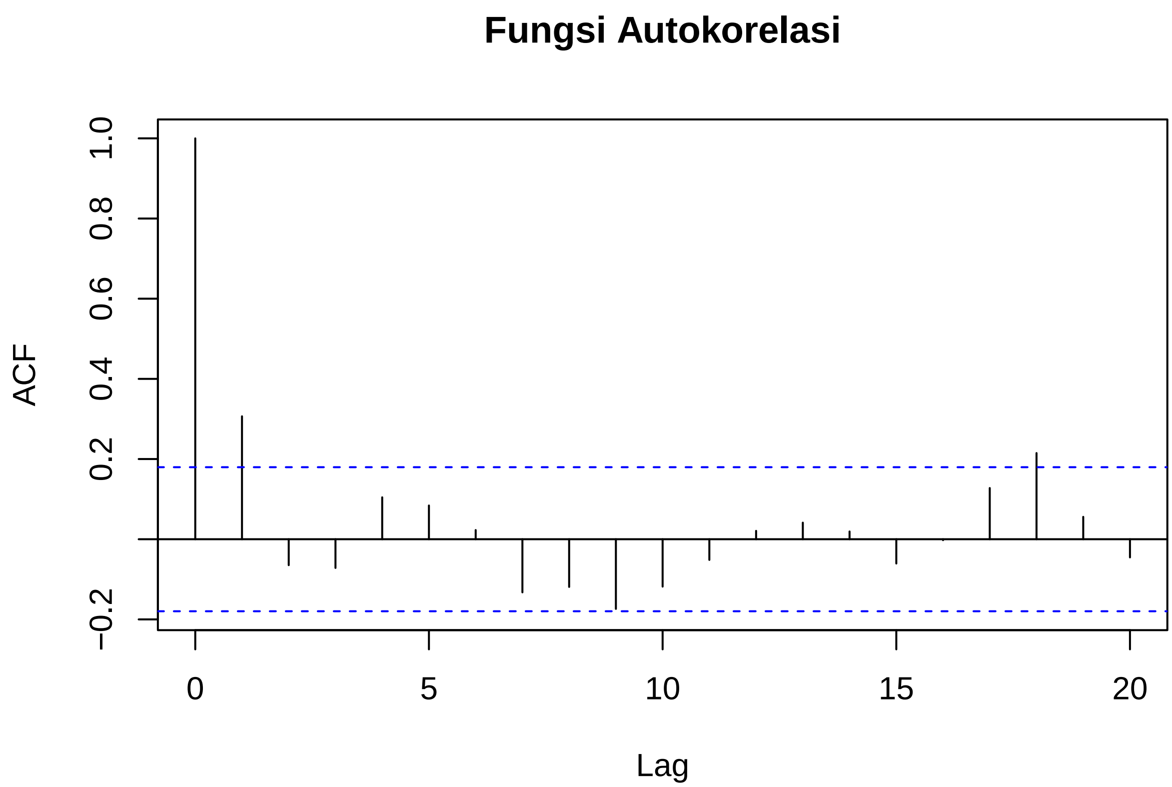
```
ztl = diff(yt, differences=1)
ts.plot(ztl,
        main="Data Pembezaan Tertib 1",
        ylab="ztl")
```



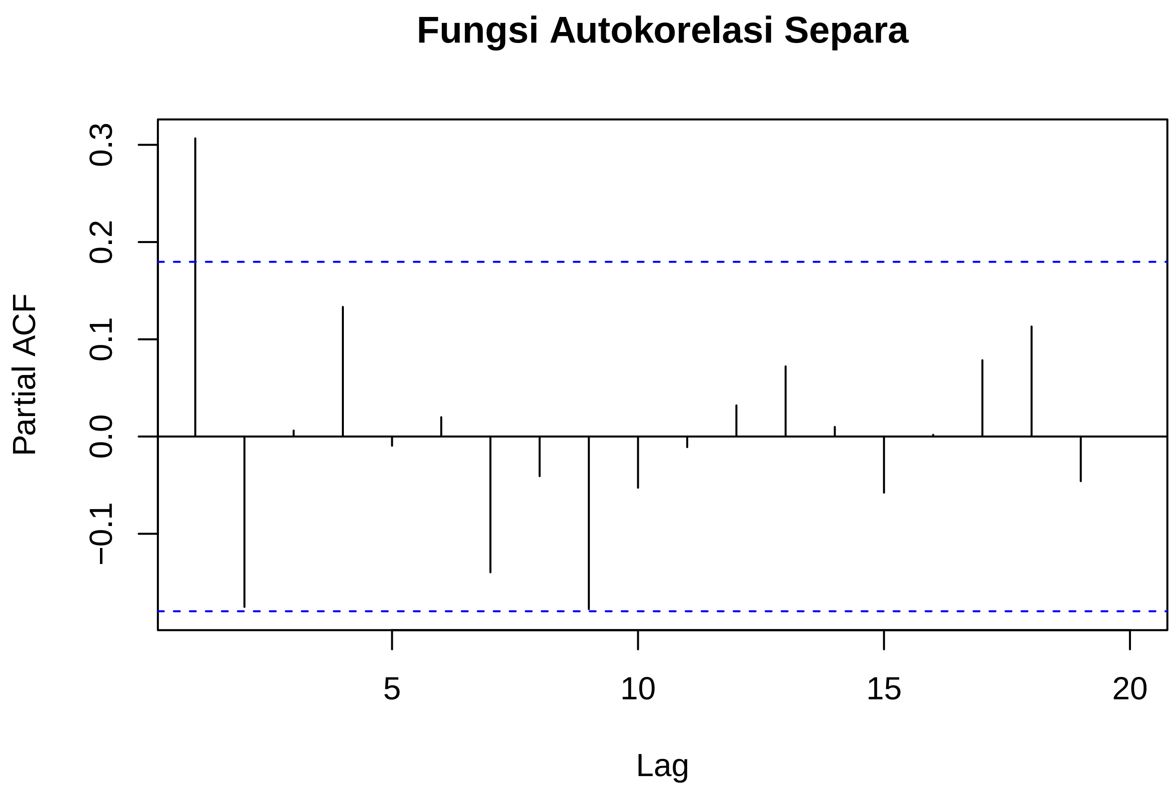
Data pembezaan peringkat-1 adalah pegun

Penentuan model ARIMA (p,i,q) Plotkan fungsi ACF dan PACF

```
acf(ztl,main="Fungsi Autokorelasi")
```




```r
pacf(ztl,main="Fungsi Autokorelasi Separa")
```



Berdasarkan plot ACF dan PACF, didapati ACF terpangkas pada tertib 1 dan PACF menurun terhadap masa.

Model yang mungkin sesuai ialah ARIMA(0,1,1)

```
#model = arima(yt,order=C(0,1,1))
#summary(model)
```

```
library(forecast)
```

Dapatkan model ARIMA yang sesuai secara automatik

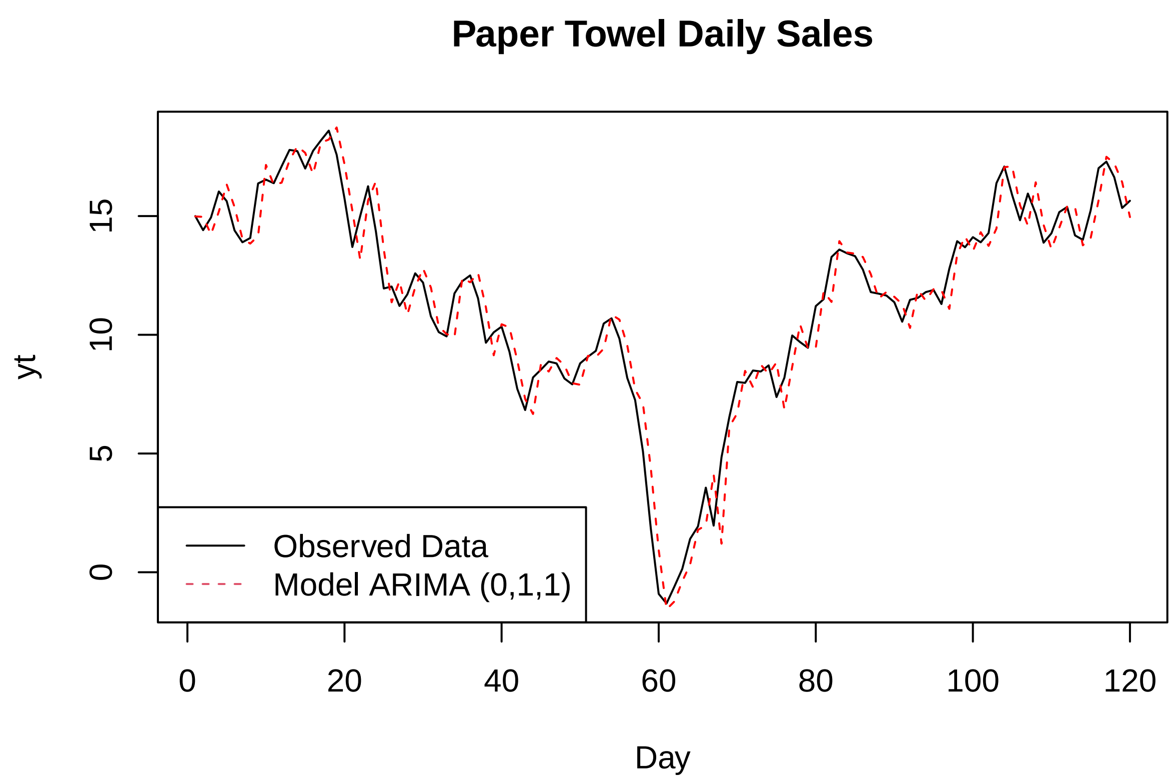
```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
model = auto.arima(yt)
model
```

```
## Series: yt
## ARIMA(0,1,1)
##
## Coefficients:
```

```
##          ma1
##       0.3518
## s.e.  0.0800
##
## sigma^2 = 1.08:  log likelihood = -172.99
## AIC=349.98   AICc=350.08   BIC=355.53
```

```
ts.plot(yt,
        main="Paper Towel Daily Sales",
       ylab="yt",
        xlab='Day');lines(fitted(model), col='red', lty=2);legend("bottomleft", c("Observed Data","Model
```

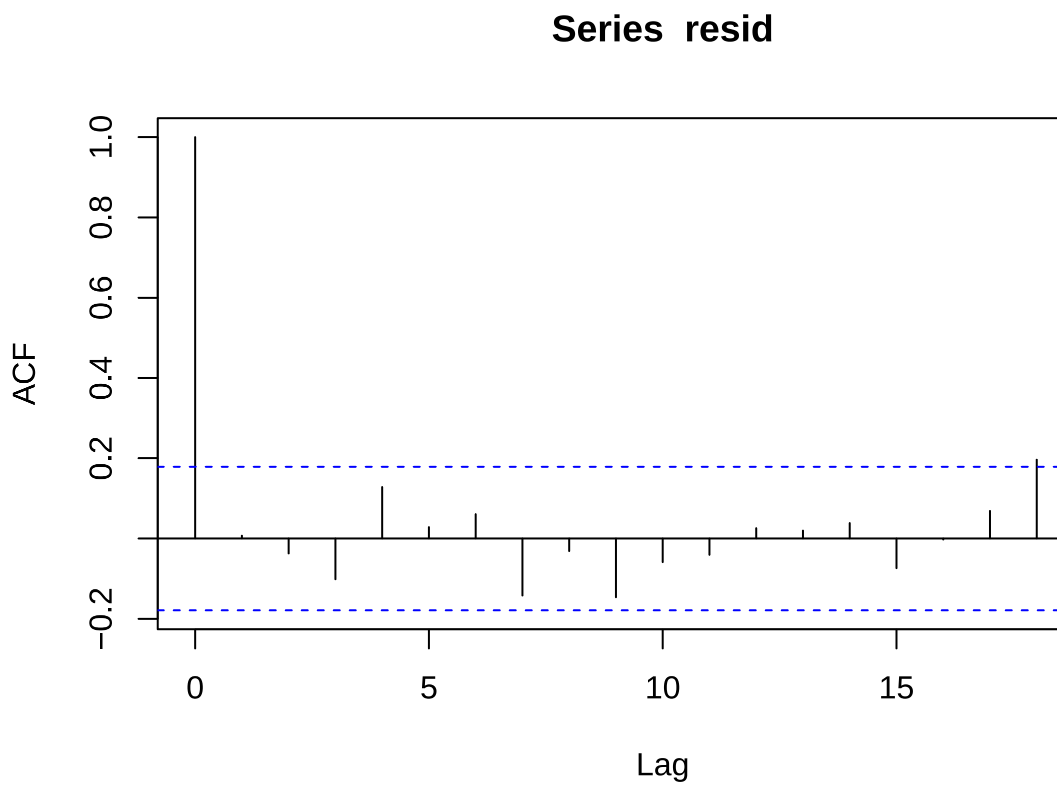


Analisis Reja [*Residuals*] (Diagnostic Model)

1. Reja adalah tak berkorelasi.
2. Reja tertabur secara normal.
3. Varians bagi reja adalah malar terhadap masa.

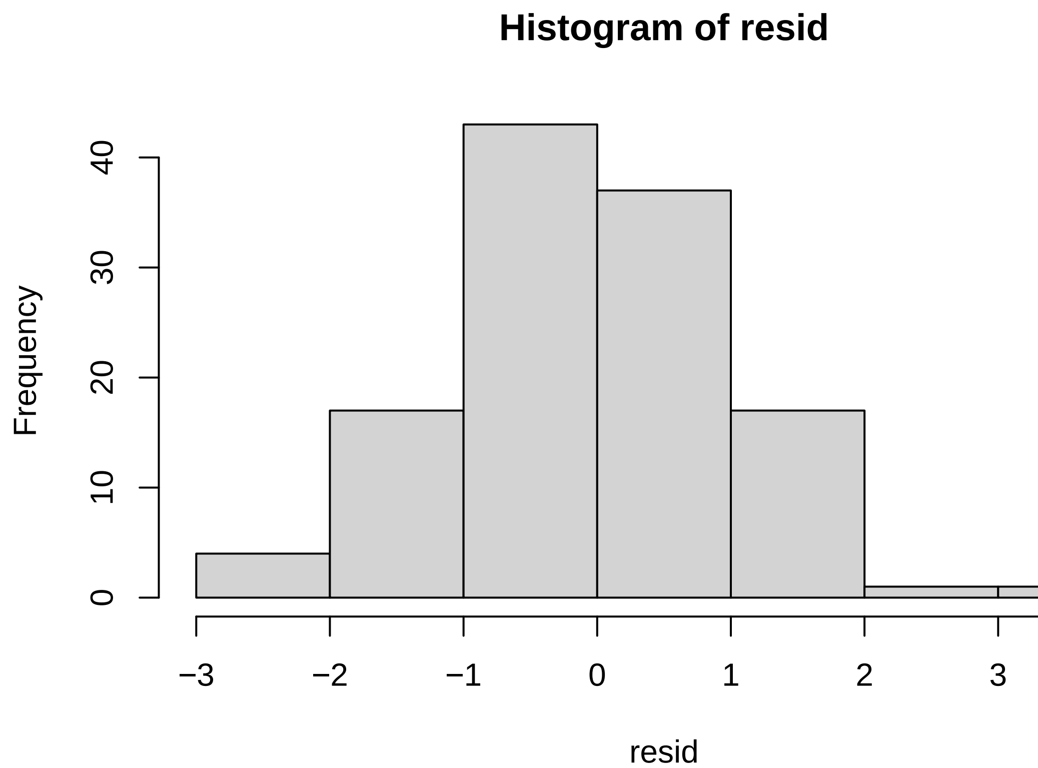
```
f.value = forecast(model, h=5)
resid = f.value$residuals
```

```r
acf(resid)
```



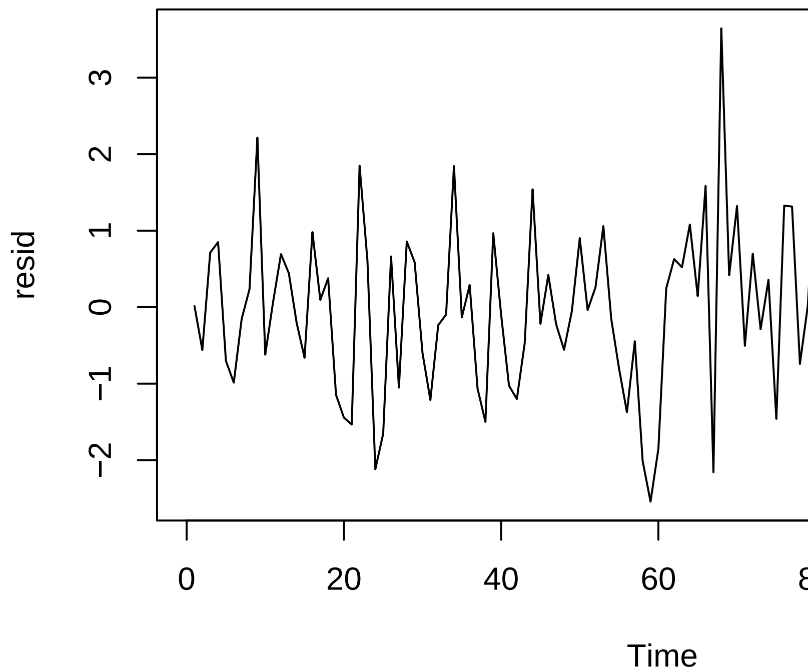
Reja adalah tak berkolerasi

```r
hist(resid)
```



Reja tertabur secara normal.

```r
plot.ts(resid)
```



Varians bagi reja adalah malar terhadap masa.

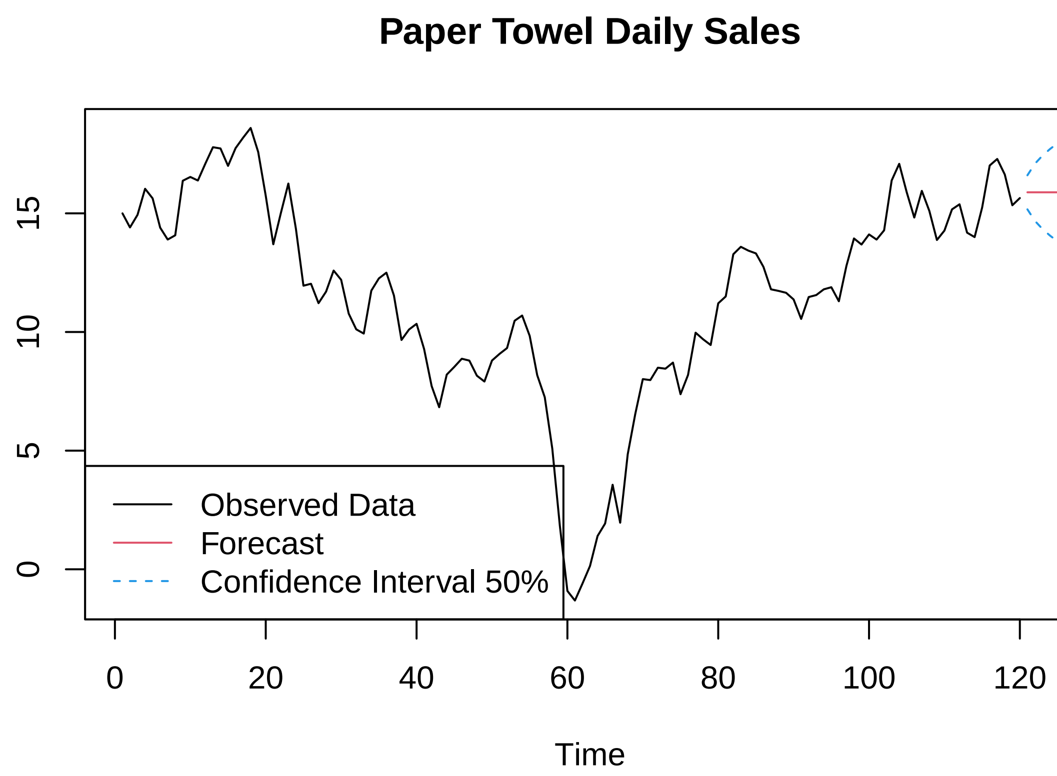
```
fore = predict(model, n.ahead=5)
fore
```

Peramalan berdasarkan model

```
## $pred
## Time Series:
## Start = 121
## End = 125
## Frequency = 1
## [1] 15.88729 15.88729 15.88729 15.88729 15.88729
##
## $se
## Time Series:
## Start = 121
## End = 125
## Frequency = 1
## [1] 1.039152 1.747345 2.242006 2.645745 2.995554
```

```
U = fore$pred+0.69*fore$se
L = fore$pred-0.69*fore$se
```

```r
ts.plot(yt, fore$pred, U, L,
        main="Paper Towel Daily Sales",
        col = c(1,2,4,4), lty=c(1,1,2,2));legend("bottomleft", c("Observed Data", "Forecast", "Confiden
```



50% selang keyakinan

Latihan

1. Import economic_data.csv.

```r
econ = read.csv("E:/Master-Data-Science/Semester_1/Data_Mining/Data/economic_data.csv", header = T, sep
head(econ,10)
```

```
##    Time Economic_Data.x
## 1     1        2.697622
## 2     2        8.509367
## 3     3       19.293542
## 4     4       11.012796
## 5     5        8.146439
## 6     6       11.575325
## 7     7        0.804581
## 8     8      -10.985560
## 9     9       -8.934264
## 10   10       -5.888564
```

2. Takrifkan data kepada format siri masa iaitu ianya adalah data bulanan bermula Januari 2000.

```
econ_time = ts(econ$Economic_Data.x, start=c(2000,1), frequency=12)
econ_time
```

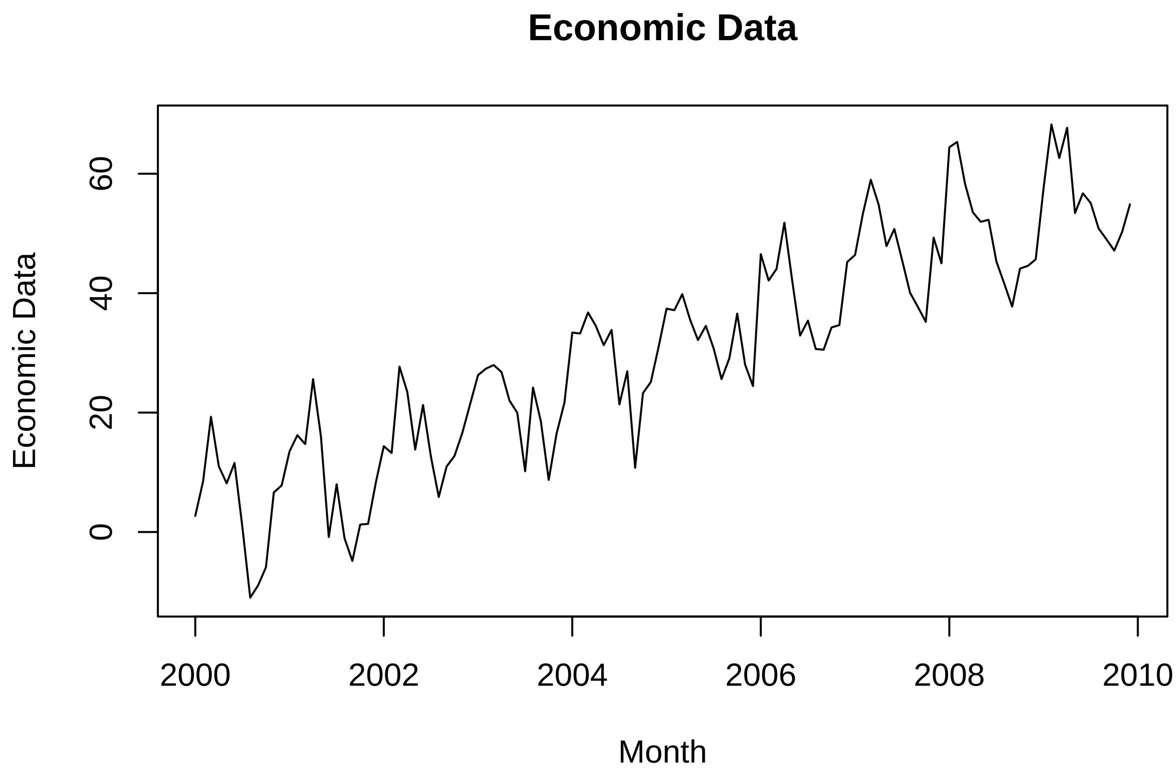
```
##             Jan         Feb         Mar         Apr         May         Jun
## 2000  2.6976218   8.5093666  19.2935416  11.0127960   8.1464387  11.5753249
## 2001 13.5038572  16.2136676  14.7207943  25.5948197  15.9892524  -0.8330858
## 2002 14.3748037  13.2267875  27.6889352  23.4271196  13.8093153  21.2690746
## 2003 26.2695883  27.3506955  27.9701867  26.7578990  22.0264651  19.9604136
## 2004 33.3998256  33.2434087  36.7665926  34.5175203  31.2856477  33.8430114
## 2005 37.3981974  37.1486368  39.8339631  35.5673771  32.1410439  34.5176432
## 2006 46.5286926  42.1142502  44.0599569  51.7881109  42.0761350  32.8964114
## 2007 46.3975672  53.3191639  58.9841951  54.8361615  47.8703421  50.7440381
## 2008 64.4366650  65.3233072  58.3214982  53.5281495  51.9479672  52.2844185
## 2009 57.5988674  68.2552371  62.6232652  67.7000757  53.4105865  56.7221902
##             Jul         Aug         Sep         Oct         Nov         Dec
## 2000  0.8045810 -10.9855602  -8.9342643  -5.8885639   6.6204090   7.7990691
## 2001  8.0067795  -1.0242111  -4.8391185   1.2498714   1.3699778   8.3555439
## 2002 12.6323211   5.8643885  10.9756283  12.7304134  16.6079054  21.4432013
## 2003 10.1730182  24.1845258  18.5398100   8.7242030  16.4855758  21.6667232
## 2004 21.3711451  26.9220990  10.7562360  23.2628147  25.1192712  31.0797078
## 2005 30.7410489  25.6047671  29.1113373  36.5901694  28.0448442  24.4541556
## 2006 35.4065174  30.6452891  30.5288209  34.2661480  34.6466998  45.2218827
## 2007 45.4675193  40.0817308  37.6936587  35.2002156  49.3032622  44.9987021
## 2008 45.2665406  41.6020330  37.7419072  44.1146073  44.5754777  45.6602903
## 2009 55.0970360  50.8455128  49.0283810  47.1362159  50.2514783  54.8793560
```

```
str(econ_time)
```

```
##  Time-Series [1:120] from 2000 to 2010: 2.7 8.51 19.29 11.01 8.15 ...
```

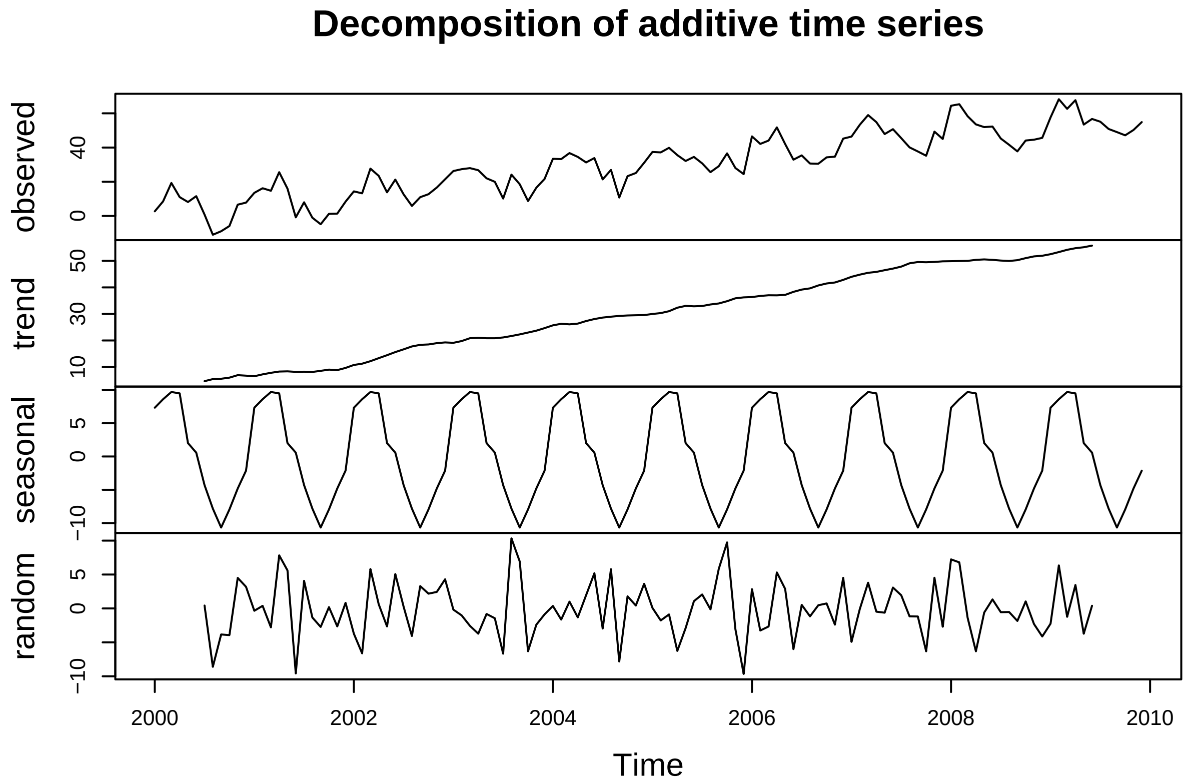
3. Plotkan siri masa tersebut.

```
ts.plot(econ_time,
        main="Economic Data",
        ylab="Economic Data",
        xlab='Month')
```



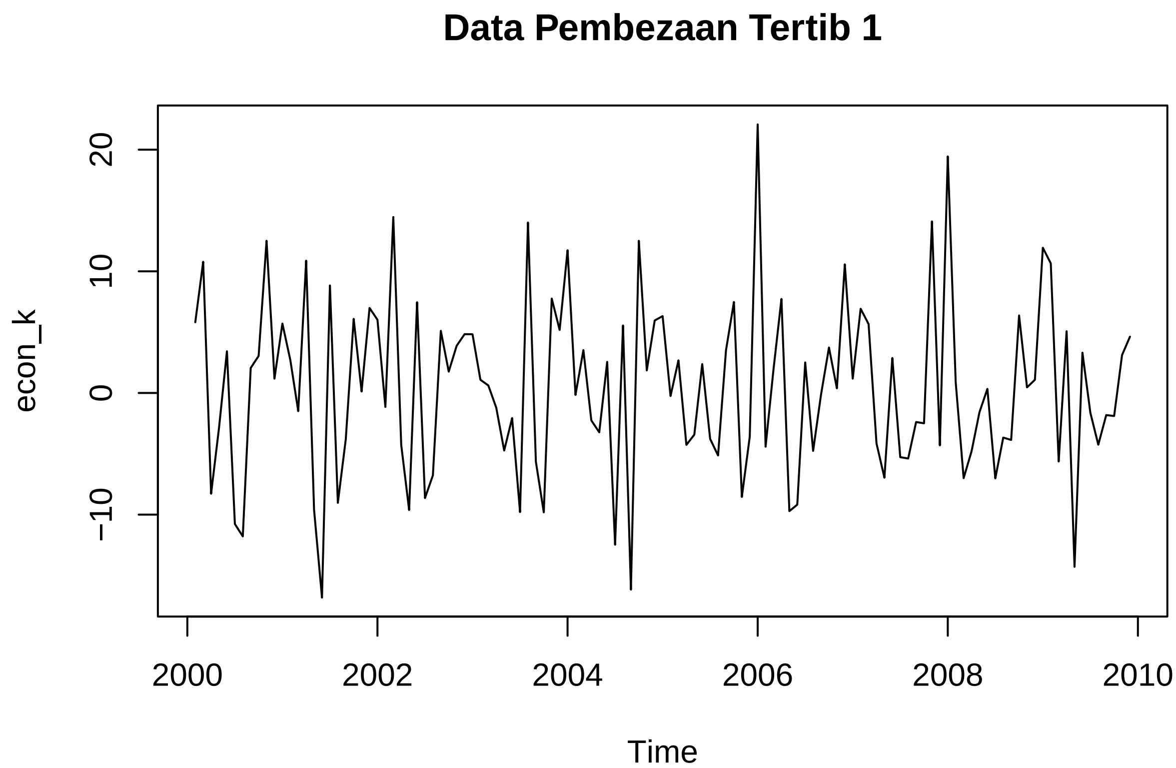
Komponen Trend

```
econ.decompose = decompose(econ_time)
plot(econ.decompose)
```



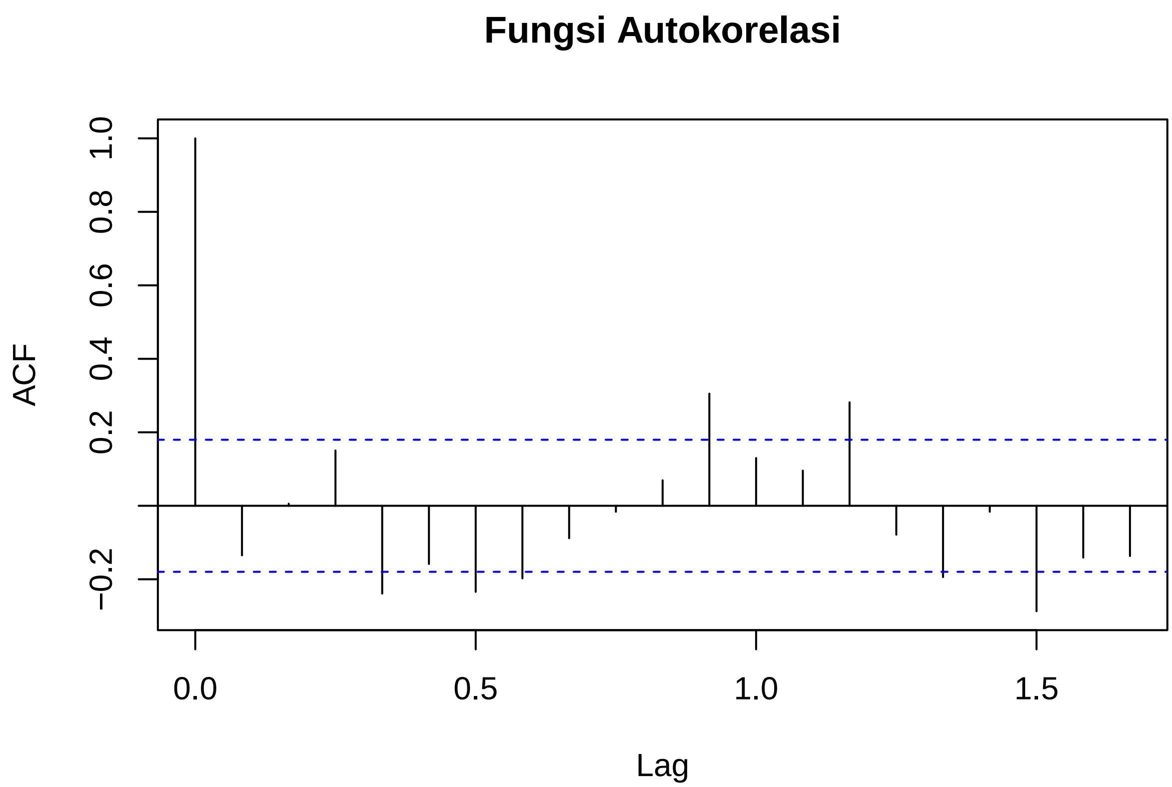
4. Kenalpasti dan suaikan model ARIMA yang sesuai terhadap data.

```
econ_k = diff(econ_time, differences=1)
ts.plot(econ_k,
        main="Data Pembezaan Tertib 1",
        ylab="econ_k")
```

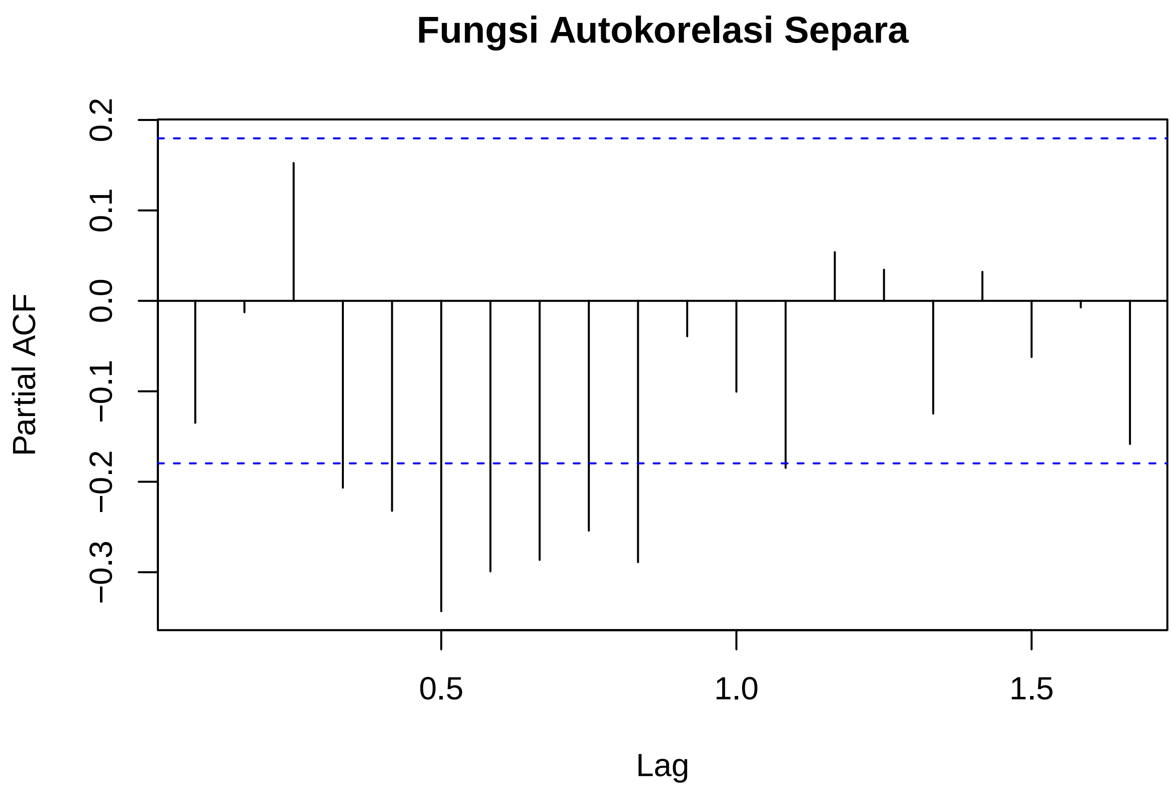


Plotkan fungsi ACF dan PACF

```
acf(econ_k,main="Fungsi Autokorelasi")
```




```
pacf(econ_k,main="Fungsi Autokorelasi Separa")
```




```r
auto.arima(econ_time)
```

```
## Series: econ_time
## ARIMA(0,0,0)(0,1,2)[12] with drift
##
## Coefficients:
##          sma1    sma2   drift
##       -1.0488  0.1765  0.4944
## s.e.   0.1606  0.1148  0.0109
##
## sigma^2 = 21.76:  log likelihood = -327.72
## AIC=663.44   AICc=663.82   BIC=674.16
```

```r
model2 = auto.arima(econ_time)
summary(model2)
```

```
## Series: econ_time
## ARIMA(0,0,0)(0,1,2)[12] with drift
##
## Coefficients:
##          sma1    sma2   drift
##       -1.0488  0.1765  0.4944
## s.e.   0.1606  0.1148  0.0109
##
## sigma^2 = 21.76:  log likelihood = -327.72
```

```
## AIC=663.44   AICc=663.82   BIC=674.16
##
## Training set error measures:
##                        ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -0.01724821 4.363728 3.269357 5.841319 32.55449 0.4613457
##                     ACF1
## Training set 0.04087961
```

5. Jalankan peramalan terhadap data untuk 24 bulan seterusnya.

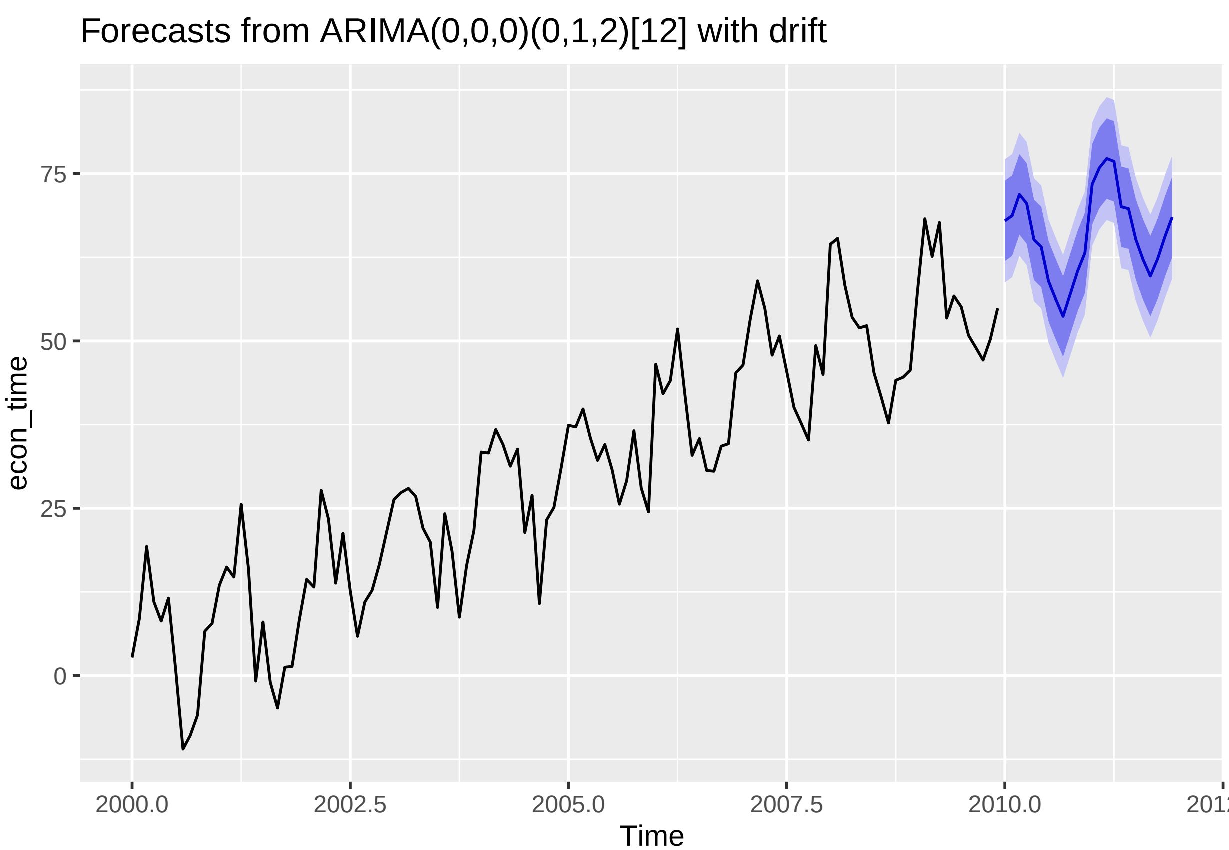
```r
econ_pred = forecast(model2, h=24)
str(econ_pred)
```

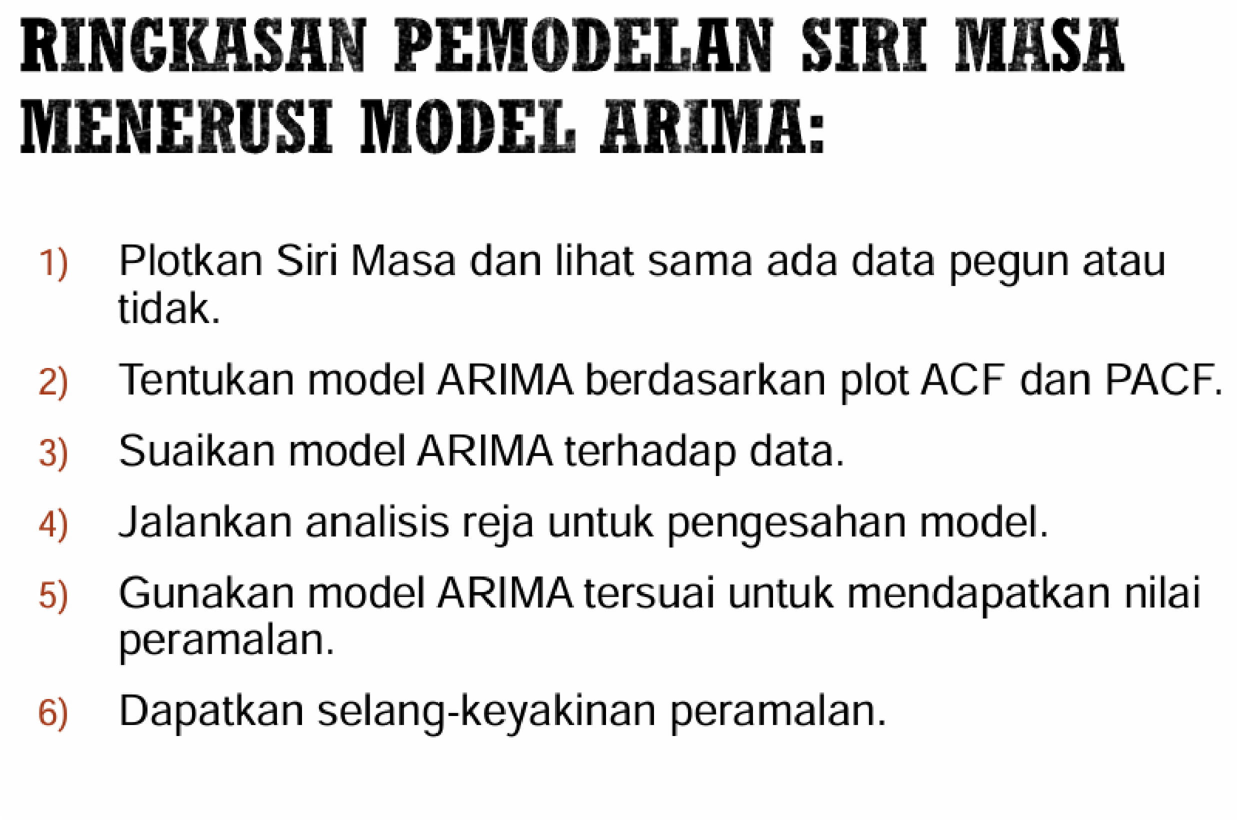
```
## List of 10
##  $ method   : chr "ARIMA(0,0,0)(0,1,2)[12] with drift"
##  $ model    :List of 19
##   ..$ coef     : Named num [1:3] -1.049 0.176 0.494
##   .. ..- attr(*, "names")= chr [1:3] "sma1" "sma2" "drift"
##   ..$ sigma2   : num 21.8
##   ..$ var.coef : num [1:3, 1:3] 2.58e-02 -4.53e-03 -8.09e-05 -4.53e-03 1.32e-02 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:3] "sma1" "sma2" "drift"
##   .. .. ..$ : chr [1:3] "sma1" "sma2" "drift"
##   ..$ mask     : logi [1:3] TRUE TRUE TRUE
##   ..$ loglik   : num -328
##   ..$ aic      : num 663
##   ..$ arma     : int [1:7] 0 0 0 2 12 0 1
##   ..$ residuals: Time-Series [1:120] from 2000 to 2010: 0.0022 0.00752 0.01781 0.00904 0.00567 ...
##   ..$ call     : language auto.arima(y = econ_time, x = list(x = c(2.697621767, 8.50936659, 19.29354
##   ..$ series   : chr "econ_time"
##   ..$ code     : int 0
##   ..$ n.cond   : int 0
##   ..$ nobs     : int 108
##   ..$ model    :List of 10
##   .. ..$ phi  : num(0)
##   .. ..$ theta: num [1:24] 0 0 0 0 0 0 0 0 0 0 ...
##   .. ..$ Delta: num [1:12] 0 0 0 0 0 0 0 0 0 0 ...
##   .. ..$ Z    : num [1:37] 1 0 0 0 0 0 0 0 0 0 ...
##   .. ..$ a    : num [1:37] 3.29 4.41 -5.46 3.34 -3.1 ...
##   .. ..$ P    : num [1:37, 1:37] 0 0 0 0 0 0 0 0 0 0 ...
##   .. ..$ T    : num [1:37, 1:37] 0 0 0 0 0 0 0 0 0 0 ...
##   .. ..$ V    : num [1:37, 1:37] 1 0 0 0 0 0 0 0 0 0 ...
##   .. ..$ h    : num 0
##   .. ..$ Pn   : num [1:37, 1:37] 1.02 0 0 0 0 ...
##   ..$ xreg     : int [1:120, 1] 1 2 3 4 5 6 7 8 9 10 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : NULL
##   .. .. ..$ : chr "drift"
##   ..$ bic      : num 674
##   ..$ aicc     : num 664
##   ..$ x        : Time-Series [1:120] from 2000 to 2010: 2.7 8.51 19.29 11.01 8.15 ...
##   ..$ fitted   : Time-Series [1:120] from 2000 to 2010: 2.7 8.5 19.28 11 8.14 ...
```

```
##   ..- attr(*, "class")= chr [1:3] "forecast_ARIMA" "ARIMA" "Arima"
##  $ level    : num [1:2] 80 95
##  $ mean     : Time-Series [1:24] from 2010 to 2012: 67.9 68.7 71.9 70.5 65.1 ...
##  $ lower    : Time-Series [1:24, 1:2] from 2010 to 2012: 61.9 62.7 65.9 64.5 59.1 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:2] "80%" "95%"
##  $ upper    : Time-Series [1:24, 1:2] from 2010 to 2012: 74 74.7 77.9 76.5 71.1 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr [1:2] "80%" "95%"
##  $ x        : Time-Series [1:120] from 2000 to 2010: 2.7 8.51 19.29 11.01 8.15 ...
##  $ series   : chr "econ_time"
##  $ fitted   : Time-Series [1:120] from 2000 to 2010: 2.7 8.5 19.28 11 8.14 ...
##  $ residuals: Time-Series [1:120] from 2000 to 2010: 0.0022 0.00752 0.01781 0.00904 0.00567 ...
##  - attr(*, "class")= chr "forecast"
```

6. Plotkan peramalan bersama selang keyakinan.

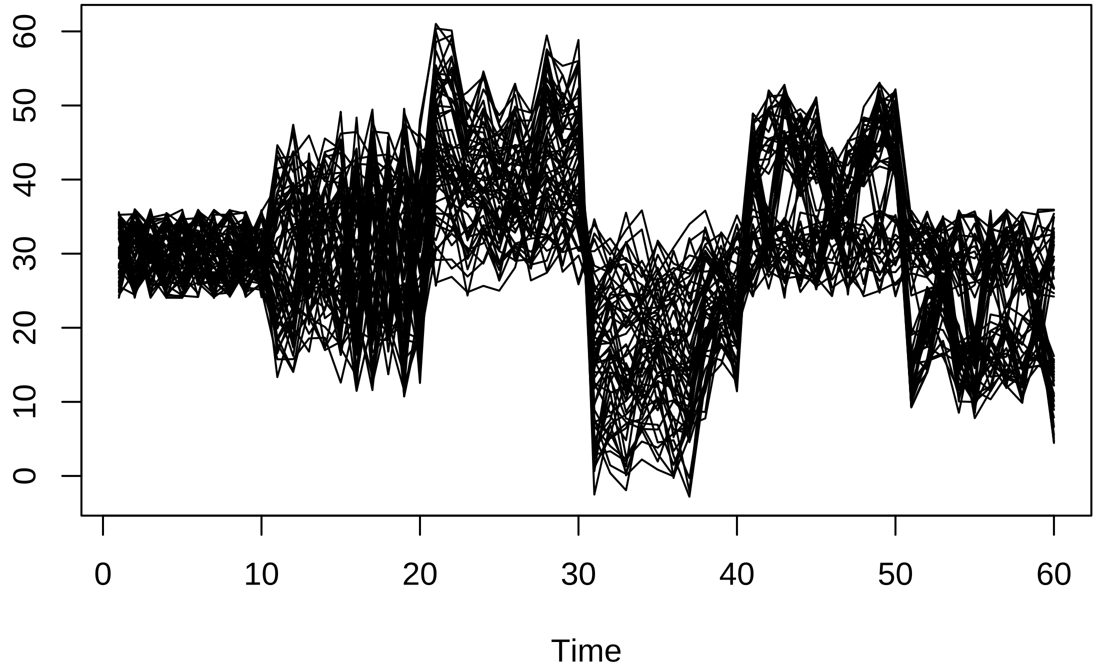
```r
autoplot(econ_pred)
```



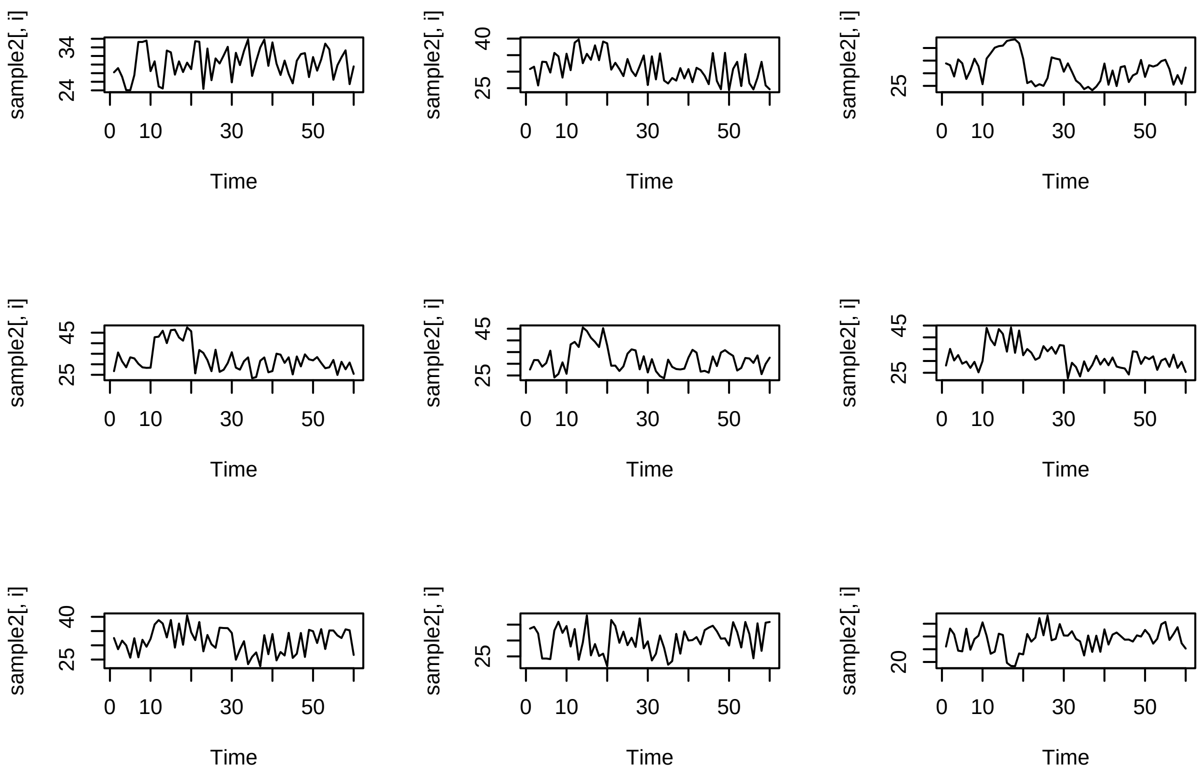


3. Pengkelompokan siri masa

```r
load("E:/Master-Data-Science/Semester_1/Data_Mining/Data/sample2.RData")
ts.plot(sample2)
```




```
par(mfrow=c(3,3))  # Set up a 3x3 grid for plots
for (i in 1:9) {   # Loop over the indices from 1 to 9
  plot.ts(sample2[,i])  # Plot the i-th column of 'sample2'
}
```



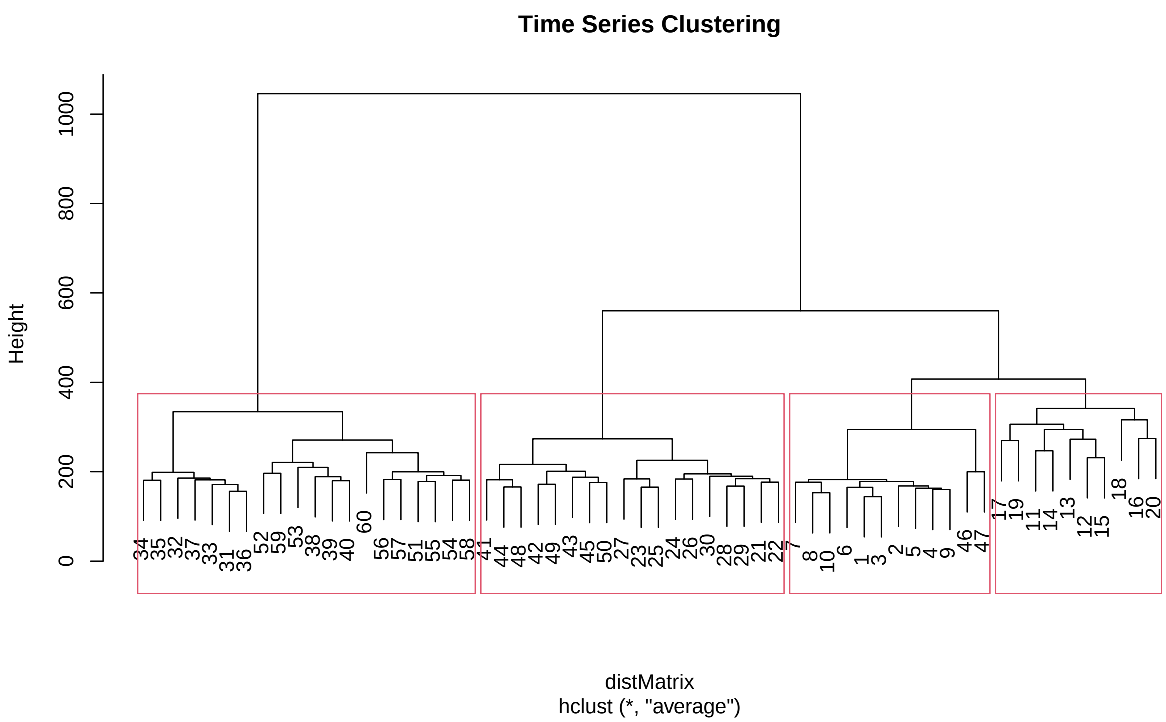
```r
library(dtw)
```

```
## Loading required package: proxy

##
## Attaching package: 'proxy'

## The following object is masked from 'package:timeSeries':
##
##     as.matrix

## The following objects are masked from 'package:stats':
##
##     as.dist, dist

## The following object is masked from 'package:base':
##
##     as.matrix

## Loaded dtw v1.23-1. See ?dtw for help, citation("dtw") for use in publication.
```

```r
D.Labels = rep(1:60)
distMatrix = dist(sample2, method = 'DTW')
TSCluster = hclust(distMatrix, method = 'average')
```

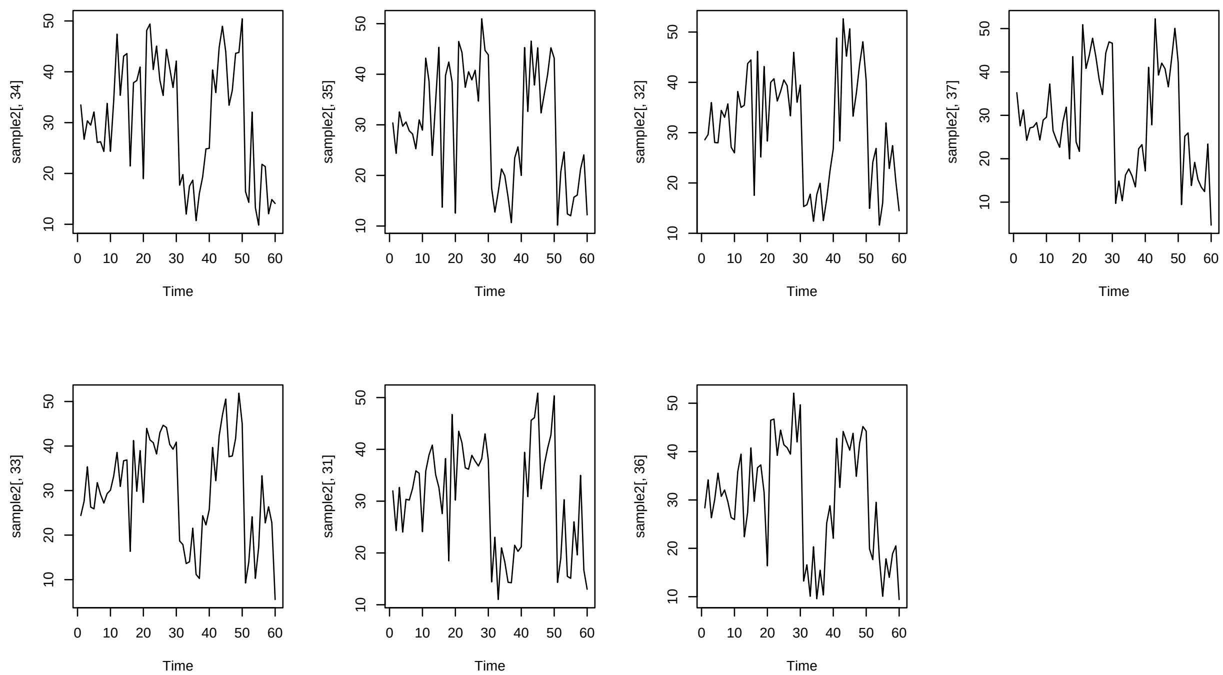
Pangkas untuk dapatkan bilangan kelompok

```
plot(TSCluster, labels = D.Labels,main = 'Time Series Clustering');rect.hclust(TSCluster, k =4)
```



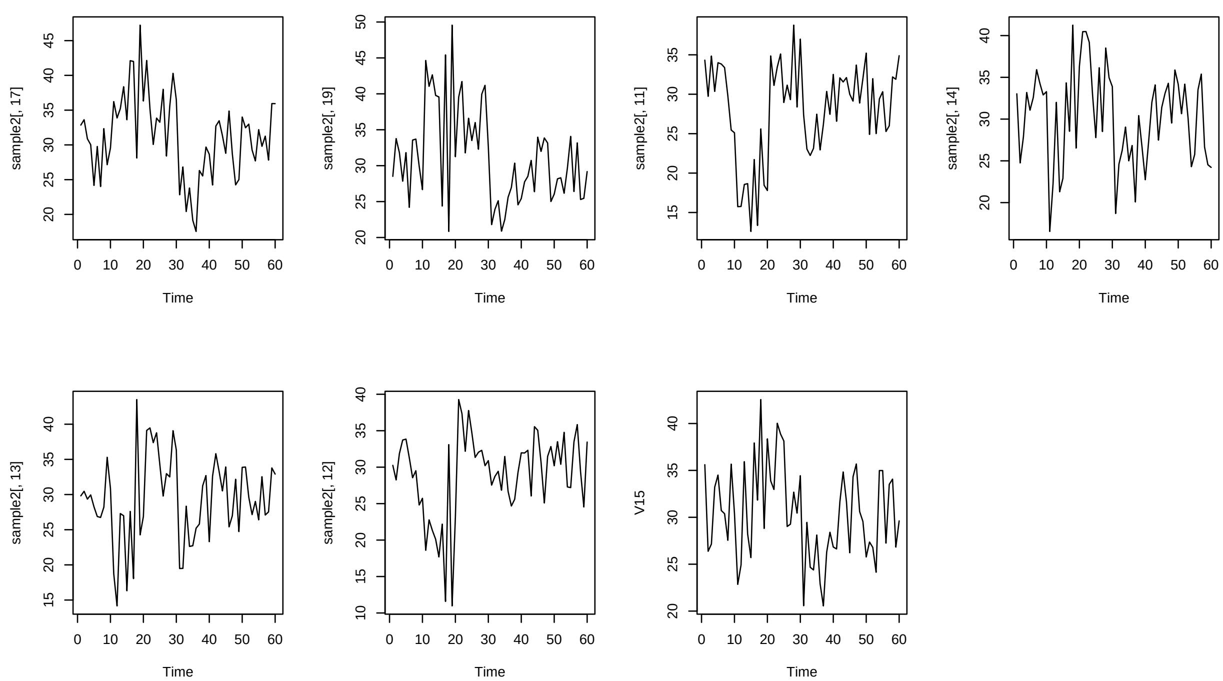
Kelompok 1

```
par(mfrow=c(2,4))
plot.ts(sample2[,34]);plot.ts(sample2[,35]);plot.ts(sample2[,32]);plot.ts(sample2[,37]);plot.ts(sample2
```



Kelompok 2

```
par(mfrow=c(2,4))
plot.ts(sample2[,17]);plot.ts(sample2[,19]);plot.ts(sample2[,11]);plot.ts(sample2[,14]);plot.ts(sample2
```



Klasifikasi Siri Masa

```r
newdata = read.csv("E:/Master-Data-Science/Semester_1/Data_Mining/Data/newdata.csv", header=T, sep=';')
head(newdata)
```

```
##        V1      V2      V3      V4      V5      V6      V7      V8      V9
## 1 28.7812 34.4632 31.3381 31.2834 28.9207 33.7596 25.3969 27.7849 35.2479
## 2 24.8923 25.7410 27.5532 32.8217 27.8789 31.5926 31.4861 35.5469 27.9516
## 3 31.3987 30.6316 26.3983 24.2905 27.8613 28.5491 24.9717 32.4358 25.2239
## 4 25.7740 30.5262 35.4209 25.6033 27.9700 25.2702 28.1320 29.4268 31.4549
## 5 27.1798 29.2498 33.6928 25.6264 24.6555 28.9446 35.7980 34.9446 24.5596
## 6 25.5067 29.7929 28.0765 34.4812 33.8000 27.6671 30.6122 25.6393 30.1171
##       V10     V11     V12     V13     V14     V15     V16     V17     V18
## 1 27.1159 32.8717 29.2171 36.0253 32.3370 34.5249 32.8717 34.1173 26.5235
## 2 31.6595 27.5415 31.1887 27.4867 31.3910 27.8110 24.4880 27.5918 35.6273
## 3 27.3068 31.8387 27.2587 28.2572 26.5819 24.0455 35.0625 31.5717 32.5614
## 4 27.3200 28.9564 28.9916 29.9578 30.2773 30.4447 24.3037 24.3140 35.0966
## 5 34.2366 27.9634 25.3216 35.4154 34.8620 25.1472 29.4686 33.1739 31.1274
## 6 26.5188 30.1524 27.8514 29.5582 32.3601 29.2064 26.1001 33.4677 33.9010
##       V19     V20     V21     V22     V23     V24     V25     V26     V27
## 1 27.6623 26.3693 25.7744 29.2700 30.7326 29.5054 33.0292 25.0400 28.9167
## 2 35.4102 31.4167 30.7447 24.1311 35.1422 30.4719 31.9874 33.6615 25.5511
## 3 31.0308 34.1202 26.9337 31.4781 35.0173 32.3851 24.3323 30.2001 31.2452
## 4 25.3679 32.0968 33.3303 25.0102 35.3155 31.6264 29.2806 34.2021 26.5077
## 5 31.3701 26.5173 28.6486 31.6565 35.9497 33.0321 24.6081 33.2025 27.4335
## 6 29.2674 34.8311 31.9815 26.4960 32.6645 27.7188 35.7385 32.8309 30.1509
##       V28     V29     V30     V31     V32     V33     V34     V35     V36
## 1 24.3437 26.1203 34.9424 25.0293 26.6311 35.6541 28.4353 29.1495 28.1584
## 2 30.4686 33.6472 25.0701 34.0765 32.5981 28.3038 26.1471 26.9414 31.5203
## 3 26.6814 31.5137 28.8778 27.3086 24.2460 26.9631 25.2919 31.6114 24.7131
## 4 32.2279 25.5265 24.8240 27.5587 28.3714 32.3667 26.9752 35.9346 35.1146
## 5 32.6355 35.8773 28.0295 33.1247 33.4129 26.9245 30.2123 29.6526 30.8644
## 6 30.5593 27.3321 27.4559 24.2361 34.7268 29.9207 27.2730 35.9963 32.3917
##       V37     V38     V39     V40     V41     V42     V43     V44     V45
## 1 26.1927 33.3182 30.9772 27.0443 35.5344 26.2353 28.9964 32.0036 31.0558
## 2 33.1089 24.1491 28.5157 25.7906 35.9519 26.5301 24.8578 25.9562 32.8357
## 3 27.4809 24.2075 26.8059 35.1253 32.6293 31.0561 26.3583 28.0861 31.4391
## 4 24.3749 27.6083 27.8433 29.8557 32.4185 26.8908 31.3209 29.3849 34.3336
## 5 24.5119 33.9931 33.3094 33.2040 31.2651 27.9072 35.1110 35.0757 33.8330
## 6 27.1390 26.4589 25.0466 35.5002 27.9961 25.8897 31.3951 30.7583 34.9652
##       V46     V47     V48     V49     V50     V51     V52     V53     V54
## 1 34.2553 28.0721 28.9402 35.4973 29.7470 31.4333 24.5556 33.7431 25.0466
## 2 28.5322 26.3458 30.6213 28.9861 29.4047 32.5577 31.0205 26.6418 28.4331
## 3 27.3057 29.6082 35.9725 34.1444 27.1717 33.6318 26.5966 25.5387 32.5434
## 4 24.7381 35.7690 31.8725 34.2054 31.1560 34.6292 28.7261 28.2979 31.5787
## 5 25.9481 29.1348 24.2875 32.3223 34.9244 27.7218 27.9601 35.7198 27.5760
## 6 28.0919 35.6706 33.4401 28.4580 31.1795 26.9458 35.8381 26.7134 25.1641
##       V55     V56     V57     V58     V59     V60 pattern100
## 1 34.9318 34.9879 32.4721 33.3759 25.4652 25.8717     Normal
## 2 33.6564 26.4244 28.4661 34.2484 32.1005 26.6910     Normal
## 3 25.5772 29.9897 31.3510 33.9002 29.5446 29.3430     Normal
## 4 34.6156 32.5492 30.9827 24.8938 27.3659 25.3069     Normal
## 5 35.3375 29.9993 34.2149 33.1276 31.1057 31.0179     Normal
```

```
## 6 27.3410 25.2093 33.4669 24.1094 33.1669 35.4907     Normal
```

```r
newdata$pattern100 = as.factor(newdata$pattern100)
library(party)
```

```
## Loading required package: grid
```

```
## Loading required package: mvtnorm
```

```
## Loading required package: modeltools
```

```
## Loading required package: stats4
```

```
## Loading required package: strucchange
```

```
## Loading required package: sandwich
```

```r
model3 = ctree(pattern100~.,newdata)
model3
```

```
##
##   Conditional inference tree with 25 terminal nodes
##
## Response:  pattern100
## Inputs:  V1, V2, V3, V4, V5, V6, V7, V8, V9, V10, V11, V12, V13, V14, V15, V16, V17, V18, V19, V20, V
## Number of observations:  600
##
## 1) V59 <= 45.6952; criterion = 1, statistic = 510.323
##   2) V59 <= 35.9324; criterion = 1, statistic = 382.48
##     3) V59 <= 23.9595; criterion = 1, statistic = 264.546
##       4) V54 <= 27.3847; criterion = 1, statistic = 149.907
##         5) V19 <= 35.1025; criterion = 1, statistic = 97.167
##           6) V16 <= 24.8534; criterion = 1, statistic = 77.304
##             7) V8 <= 31.5525; criterion = 0.961, statistic = 11.609
##               8)*  weights = 54
##             7) V8 > 31.5525
##               9)*  weights = 7
##           6) V16 > 24.8534
##             10) V19 <= 23.9112; criterion = 1, statistic = 48.619
##               11)*  weights = 21
##             10) V19 > 23.9112
##               12) V17 <= 24.8025; criterion = 1, statistic = 22.548
##                 13)*  weights = 11
##               12) V17 > 24.8025
##                 14) V20 <= 24.0984; criterion = 0.994, statistic = 15.014
##                   15)*  weights = 7
##                 14) V20 > 24.0984
##                   16) V13 <= 24.2266; criterion = 0.985, statistic = 13.343
##                     17)*  weights = 7
##                   16) V13 > 24.2266
##                     18) V57 <= 10.1064; criterion = 0.977, statistic = 12.573
```

```
##                        19)*  weights = 7
##                      18) V57 > 10.1064
##                        20)*  weights = 76
##         5) V19 > 35.1025
##           21)*  weights = 7
##       4) V54 > 27.3847
##         22)*  weights = 21
##     3) V59 > 23.9595
##       23) V4 <= 36.0264; criterion = 1, statistic = 112.745
##         24) V51 <= 22.3131; criterion = 1, statistic = 56.717
##           25)*  weights = 7
##         24) V51 > 22.3131
##           26) V60 <= 35.4907; criterion = 1, statistic = 33.063
##             27) V41 <= 35.2384; criterion = 0.997, statistic = 16.711
##               28)*  weights = 91
##             27) V41 > 35.2384
##               29)*  weights = 7
##           26) V60 > 35.4907
##             30)*  weights = 7
##       23) V4 > 36.0264
##         31)*  weights = 41
##   2) V59 > 35.9324
##     32) V54 <= 32.3239; criterion = 1, statistic = 92.267
##       33)*  weights = 31
##     32) V54 > 32.3239
##       34) V19 <= 35.9115; criterion = 1, statistic = 32.264
##         35) V15 <= 35.9747; criterion = 1, statistic = 21.416
##           36)*  weights = 62
##         35) V15 > 35.9747
##           37)*  weights = 7
##       34) V19 > 35.9115
##         38)*  weights = 13
## 1) V59 > 45.6952
##   39) V20 <= 32.8423; criterion = 1, statistic = 57.208
##     40) V41 <= 41.5426; criterion = 1, statistic = 25.128
##       41)*  weights = 9
##     40) V41 > 41.5426
##       42) V57 <= 51.2746; criterion = 0.973, statistic = 12.286
##         43)*  weights = 25
##       42) V57 > 51.2746
##         44)*  weights = 7
##   39) V20 > 32.8423
##     45) V39 <= 38.2428; criterion = 1, statistic = 34.564
##       46)*  weights = 7
##     45) V39 > 38.2428
##       47) V15 <= 30.4466; criterion = 0.959, statistic = 11.478
##         48)*  weights = 7
##       47) V15 > 30.4466
##         49)*  weights = 61
```

```
matriks_konfusi = table(Predicted = predict(model3, newdata), Actual = newdata$pattern100)
matriks_konfusi
```

```
##                     Actual
```

```
## Predicted          Cyclic Decreasing trend Downward shift Increasing trend
##   Cyclic              97                0              3                0
##   Decreasing trend     0               99              8                0
##   Downward shift       0                1             89                0
##   Increasing trend     2                0              0               96
##   Normal               1                0              0                0
##   Upward shift         0                0              0                4
##                     Actual
## Predicted          Normal Upward shift
##   Cyclic               0            0
##   Decreasing trend     0            0
##   Downward shift       0            0
##   Increasing trend     0            6
##   Normal             100            4
##   Upward shift         0           90
```

```r
precision_model = sum(diag(matriks_konfusi))/sum(matriks_konfusi)
precision_model
```

```
## [1] 0.9516667
```