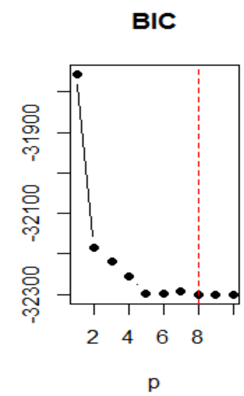
HW3 1585107 이지인




```r
library(forecast);library(lmtest)
rv.data<-read.csv("RV_IV_data.csv", header=TRUE)

#============================#
#  KOSPI.RV = Y, VKOSPI = X  #
#============================#
# Make data
y.t = rv.data$KOSPI[1:(nrow(rv.data))]   # KOSPI
x.t = rv.data$VKOSPI[1:nrow(rv.data)]      # VKOSPI


# AR 모형 BIC
AR.bic = c()
for( p in 1:10){
  AR.fit = arima(y.t, order = c(p,0,0))
  AR.bic[p] = AIC(AR.fit,k = log(length(y.t)))
}
par(mfrow=c(1,2))
plot(AR.bic, type = "b", pch =19, main = "BIC", ylab = "", xlab = "p")
abline(v = which.min(AR.bic), col=2, lty = 2)  # AR(8)

y_t = y.t[9:length(y.t)]
y_t_1 = y.t[8:(length(y.t)-1)]
y_t_2 = y.t[7:(length(y.t)-2)]
y_t_3 = y.t[6:(length(y.t)-3)]
y_t_4 = y.t[5:(length(y.t)-4)]
y_t_5 = y.t[4:(length(y.t)-5)]
y_t_6 = y.t[3:(length(y.t)-6)]
y_t_7 = y.t[2:(length(y.t)-7)]
y_t_8 = y.t[1:(length(y.t)-8)]

x_t = x.t[9:length(x.t)]
x_t_1 = x.t[8:(length(x.t)-1)]
x_t_2 = x.t[7:(length(x.t)-2)]
x_t_3 = x.t[6:(length(x.t)-3)]
x_t_4 = x.t[5:(length(x.t)-4)]
x_t_5 = x.t[4:(length(x.t)-5)]
x_t_6 = x.t[3:(length(x.t)-6)]
x_t_7 = x.t[2:(length(x.t)-7)]
x_t_8 = x.t[1:(length(x.t)-8)]

y.data = cbind(y_t_1 = y_t_1, y_t_2= y_t_2, y_t_3 = y_t_3, y_t_4 = y_t_4,
               y_t_5 = y_t_5, y_t_6= y_t_6, y_t_7 = y_t_7, y_t_8 = y_t_8)
x.data = cbind(x_t_1 = x_t_1, x_t_2= x_t_2, x_t_3 = x_t_3, x_t_4 = x_t_4,
               x_t_5 = x_t_5, x_t_6= x_t_6, x_t_7 = x_t_7, x_t_8 = x_t_8)

# ADL 모형 BIC : p=5, q=3
ADL.bic = matrix(0, ncol = 8, nrow = 8)
for( p in 1:8){
  for(q in 1:8){
```

```r
    ADL.fit = lm(y_t ~ y.data[,1:p] + x.data[,1:q])
    ADL.bic[p,q] = AIC(ADL.fit,k = log(length(y.t)))
  }
}
colnames(ADL.bic) = c("q=1", "q=2", "q=3", "q=4", "q=5", "q=6", "q=7", "q=8")
row.names(ADL.bic) = c("p=1", "p=2", "p=3", "p=4", "p=5", "p=6", "p=7", "p=8")
ADL.bic; min(ADL.bic) # ADL(5,3)
```

```
##           q=1       q=2       q=3       q=4       q=5       q=6       q=7
## p=1 -32279.06 -32272.59 -32287.76 -32285.58 -32278.34 -32276.63 -32269.12
## p=2 -32439.66 -32477.99 -32482.44 -32486.07 -32479.44 -32474.00 -32469.12
## p=3 -32435.54 -32481.74 -32503.21 -32509.51 -32503.86 -32496.41 -32492.85
## p=4 -32435.24 -32485.29 -32516.65 -32512.76 -32507.92 -32500.17 -32499.06
## p=5 -32440.29 -32492.30 -32528.03 -32521.25 -32513.10 -32505.09 -32505.22
## p=6 -32432.20 -32485.13 -32521.50 -32514.47 -32506.30 -32499.09 -32499.87
## p=7 -32425.45 -32477.71 -32513.40 -32506.40 -32498.23 -32490.92 -32493.94
## p=8 -32419.65 -32473.20 -32509.90 -32502.54 -32494.38 -32487.46 -32488.02
##           q=8
## p=1 -32276.20
## p=2 -32469.87
## p=3 -32491.54
## p=4 -32496.63
## p=5 -32500.60
## p=6 -32494.90
## p=7 -32489.37
## p=8 -32487.57
```

```
## [1] -32528.03
```

```r
## 1-step ahead forecasting
AR8.fore = c(); ADL53.fore = c()
for(i in 3106:(length(y.t)-1)){
  train.data = data.frame(y.t = y.t[1:i], x.t = x.t[1:i])

    y_t = train.data[9:nrow(train.data),1]
  y_t_1 = train.data[8:(nrow(train.data)-1),1]
  y_t_2 = train.data[7:(nrow(train.data)-2),1]
  y_t_3 = train.data[6:(nrow(train.data)-3),1]
  y_t_4 = train.data[5:(nrow(train.data)-4),1]
  y_t_5 = train.data[4:(nrow(train.data)-5),1]
  y_t_6 = train.data[3:(nrow(train.data)-6),1]
  y_t_7 = train.data[2:(nrow(train.data)-7),1]
  y_t_8 = train.data[1:(nrow(train.data)-8),1]

    x_t = train.data[9:nrow(train.data),2]
  x_t_1 = train.data[8:(nrow(train.data)-1),2]
  x_t_2 = train.data[7:(nrow(train.data)-2),2]
  x_t_3 = train.data[6:(nrow(train.data)-3),2]
  x_t_4 = train.data[5:(nrow(train.data)-4),2]
  x_t_5 = train.data[4:(nrow(train.data)-5),2]
  x_t_6 = train.data[3:(nrow(train.data)-6),2]
  x_t_7 = train.data[2:(nrow(train.data)-7),2]
  x_t_8 = train.data[1:(nrow(train.data)-8),2]

    # AR(8)
  AR8.fit = lm(y_t ~ y_t_1 + y_t_2 + y_t_3 + y_t_4 +
```

```r
                y_t_5 + y_t_6 + y_t_7 + y_t_8)
  AR8.fore[i-3105] = sum(AR8.fit$coef*c(1, y_t[length(y_t)],
                                        y_t_1[length(y_t)],
                                        y_t_2[length(y_t)],
                                        y_t_3[length(y_t)],
                                        y_t_4[length(y_t)],
                                        y_t_5[length(y_t)],
                                        y_t_6[length(y_t)],
                                        y_t_7[length(y_t)]))
    # ADL(5,3)
  ADL.fit1 = lm(y_t ~
                 y_t_1 + y_t_2 +y_t_3 + y_t_4 + y_t_5 +
                 x_t + x_t_1 + x_t_2 );
  ADL53.fore[i-3105] = sum(ADL.fit1$coef*c(1, y_t[length(y_t)],
                                       y_t_1[length(y_t)],
                                       y_t_2[length(y_t)],
                                       y_t_3[length(y_t)],
                                       y_t_4[length(y_t)],
                                       x_t[length(y_t)],
                                       x_t_1[length(y_t)],
                                       x_t_2[length(y_t)]))

}
# 1. VKOSPI 가 KOSPI 5 분 실현변동성을 GRANGER CAUSE 하는가
grangertest(y.t~x.t)
```

```
## Granger causality test
##
## Model 1: y.t ~ Lags(y.t, 1:1) + Lags(x.t, 1:1)
## Model 2: y.t ~ Lags(y.t, 1:1)
##   Res.Df Df     F    Pr(>F)
## 1   3554
## 2   3555 -1 587.3 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# 2. 예측력 비교
# AR(8)     VS    ADL(5,3)
# AR8.fore  VS    ADL53.fore
# MAE
AR8.MAE = mean(abs(AR8.fore - y.t[3107:length(y.t)]))
ADL53.MAE = mean(abs(ADL53.fore - y.t[3107:length(y.t)]))
AR8.MAE < ADL53.MAE      # AR(8) 모형이 MAE 값이 작으므로
```

```
## [1] TRUE
```

```r
# MSE
AR8.MSE = mean((AR8.fore - y.t[3107:length(y.t)])^2)
ADL53.MSE = mean((ADL53.fore - y.t[3107:length(y.t)])^2)
AR8.MSE < ADL53.MSE       # ADL(5,3) 모형이 MSE 값이 더 작으므로
```

```
## [1] FALSE
```

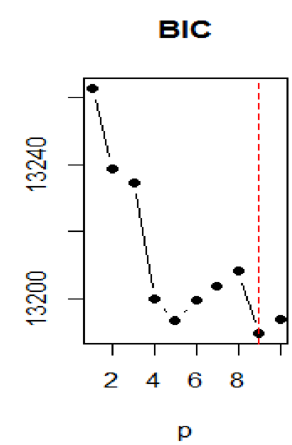
```r
result = matrix(c(AR8.MAE, ADL53.MAE, AR8.MSE, ADL53.MSE), nrow = 2, byrow = T)
row.names(result) = c("MAE", "MSE")
```

```r
colnames(result) = c("AR8", "ADL(5,3)")
result   # MAE 기준 AR(8), MSE 기준 ADL(5,3) 모형이 더 잘 예측

##              AR8     ADL(5,3)
## MAE 9.802705e-04 9.854411e-04
## MSE 3.404521e-06 3.313436e-06


#============================#
#  VKOSPI = Y, KOSPI.RV = X  #
#============================#
# MAKE DATA
y.t = rv.data$VKOSPI[1:nrow(rv.data)]      # VKOSPI
x.t = rv.data$KOSPI[1:(nrow(rv.data))]   # KOSPI.RV


# AR 모형 BIC : p=9
AR.bic = c()
for( p in 1:10){
  AR.fit = arima(y.t, order = c(p,0,0))
  AR.bic[p] = AIC(AR.fit,k = log(length(y.t)))
}
par(mfrow=c(1,2))
plot(AR.bic, type = "b", pch =19, main = "BIC", ylab = ""
, xlab = "p")
abline(v = which.min(AR.bic), col=2, lty = 2)  # AR(9)
```




```r
y_t = y.t[10:length(y.t)]
y_t_1 = y.t[9:(length(y.t)-1)]
y_t_2 = y.t[8:(length(y.t)-2)]
y_t_3 = y.t[7:(length(y.t)-3)]
y_t_4 = y.t[6:(length(y.t)-4)]
y_t_5 = y.t[5:(length(y.t)-5)]
y_t_6 = y.t[4:(length(y.t)-6)]
y_t_7 = y.t[3:(length(y.t)-7)]
y_t_8 = y.t[2:(length(y.t)-8)]
y_t_9 = x.t[1:(length(x.t)-9)]

  x_t = x.t[10:length(x.t)]
x_t_1 = x.t[9:(length(x.t)-1)]
x_t_2 = x.t[8:(length(x.t)-2)]
x_t_3 = x.t[7:(length(x.t)-3)]
x_t_4 = x.t[6:(length(x.t)-4)]
x_t_5 = x.t[5:(length(x.t)-5)]
x_t_6 = x.t[4:(length(x.t)-6)]
x_t_7 = x.t[3:(length(x.t)-7)]
x_t_8 = x.t[2:(length(x.t)-8)]
x_t_9 = x.t[1:(length(x.t)-9)]

y.data = cbind(y_t_1 = y_t_1, y_t_2= y_t_2, y_t_3 = y_t_3, y_t_4 = y_t_4,
               y_t_5 = y_t_5, y_t_6= y_t_6, y_t_7 = y_t_7, y_t_8 = y_t_8,
               y_t_9 = y_t_9)
x.data = cbind(x_t_1 = x_t_1, x_t_2= x_t_2, x_t_3 = x_t_3, x_t_4 = x_t_4,
               x_t_5 = x_t_5, x_t_6= x_t_6, x_t_7 = x_t_7, x_t_8 = x_t_8,
               x_t_9 = x_t_9)
```

```r
# ADL 모형 BIC : p=5, q=1
ADL.bic = matrix(0, ncol = 9, nrow = 9)
for( p in 1:9){
  for(q in 1:9){
    ADL.fit = lm(y_t ~ y.data[,1:p] + x.data[,1:q])
    ADL.bic[p,q] = AIC(ADL.fit,k = log(length(y.t)))
  }
}
colnames(ADL.bic) = c("q=1", "q=2", "q=3", "q=4", "q=5", "q=6", "q=7", "q=8", "q=9")
row.names(ADL.bic) = c("p=1", "p=2", "p=3", "p=4", "p=5", "p=6", "p=7", "p=8", "p=9")
ADL.bic; min(ADL.bic) # ADL(5,1)
```

```
##          q=1      q=2      q=3      q=4      q=5      q=6      q=7
## p=1 13232.15 13239.76 13245.36 13234.71 13241.76 13246.03 13252.76
## p=2 13207.66 13213.54 13221.51 13214.31 13221.66 13227.57 13234.78
## p=3 13202.57 13209.38 13216.42 13214.03 13221.73 13227.97 13235.59
## p=4 13163.22 13170.84 13178.62 13186.57 13194.43 13201.83 13209.75
## p=5 13154.38 13162.54 13170.57 13178.20 13180.37 13188.54 13196.70
## p=6 13161.27 13169.41 13177.27 13185.00 13186.74 13194.67 13202.68
## p=7 13164.47 13172.65 13180.64 13187.84 13190.22 13197.98 13205.86
## p=8 13168.50 13176.67 13184.76 13191.70 13195.31 13202.92 13210.92
## p=9 13162.36 13170.35 13177.73 13185.73 13186.40 13194.57 13200.30
##          q=8      q=9
## p=1 13249.65 13232.99
## p=2 13233.08 13217.56
## p=3 13234.46 13219.90
## p=4 13211.29 13198.34
## p=5 13199.10 13189.20
## p=6 13204.39 13193.90
## p=7 13209.54 13200.19
## p=8 13216.77 13208.35
## p=9 13208.35 13208.35

## [1] 13154.38
```

```r
## 1-step ahead forecasting
AR9.fore = c(); ADL51.fore = c()
for(i in 3106:(length(y.t)-1)){
  train.data = data.frame(y.t = y.t[1:i], x.t = x.t[1:i])

  y_t = train.data[10:nrow(train.data),1]
  y_t_1 = train.data[9:(nrow(train.data)-1),1]
  y_t_2 = train.data[8:(nrow(train.data)-2),1]
  y_t_3 = train.data[7:(nrow(train.data)-3),1]
  y_t_4 = train.data[6:(nrow(train.data)-4),1]
  y_t_5 = train.data[5:(nrow(train.data)-5),1]
  y_t_6 = train.data[4:(nrow(train.data)-6),1]
  y_t_7 = train.data[3:(nrow(train.data)-7),1]
  y_t_8 = train.data[2:(nrow(train.data)-8),1]
  y_t_9 = train.data[1:(nrow(train.data)-9),1]

  x_t = train.data[10:nrow(train.data),2]
  x_t_1 = train.data[9:(nrow(train.data)-1),2]
  x_t_2 = train.data[8:(nrow(train.data)-2),2]
  x_t_3 = train.data[7:(nrow(train.data)-3),2]
```

```r
  x_t_4 = train.data[6:(nrow(train.data)-4),2]
  x_t_5 = train.data[5:(nrow(train.data)-5),2]
  x_t_6 = train.data[4:(nrow(train.data)-6),2]
  x_t_7 = train.data[3:(nrow(train.data)-7),2]
  x_t_8 = train.data[2:(nrow(train.data)-8),2]
  x_t_9 = train.data[1:(nrow(train.data)-9),2]

  # AR(9)
  AR9.fit = lm(y_t ~ y_t_1 + y_t_2 + y_t_3 + y_t_4 +
                 y_t_5 + y_t_6 + y_t_7 + y_t_8 + y_t_9)
  AR9.fore[i-3105] = sum(AR9.fit$coef*c(1, y_t[length(y_t)],
                                        y_t_1[length(y_t)],
                                        y_t_2[length(y_t)],
                                        y_t_3[length(y_t)],
                                        y_t_4[length(y_t)],
                                        y_t_5[length(y_t)],
                                        y_t_6[length(y_t)],
                                        y_t_7[length(y_t)],
                                        y_t_8[length(y_t)]))
  # ADL(5,1)
  ADL.fit1 = lm(y_t ~
                  y_t_1 + y_t_2 +y_t_3 + y_t_4 + y_t_5 +
                  x_t );
  ADL51.fore[i-3105] = sum(ADL.fit1$coef*c(1, y_t[length(y_t)],
                                          y_t_1[length(y_t)],
                                          y_t_2[length(y_t)],
                                          y_t_3[length(y_t)],
                                          y_t_4[length(y_t)],
                                          x_t[length(y_t)] ))
}

# 1. VKOSPI 가 KOSPI 5 분 실현변동성을 GRANGER CAUSE 하는가
grangertest(y.t~x.t)
```

```
## Granger causality test
##
## Model 1: y.t ~ Lags(y.t, 1:1) + Lags(x.t, 1:1)
## Model 2: y.t ~ Lags(y.t, 1:1)
##   Res.Df Df      F Pr(>F)
## 1   3554
## 2   3555 -1 0.0184 0.8922
```

```r
# 2. 예측력 비교
# AR(8)     VS   ADL(5,1)
# AR8.fore  VS   ADL51.fore
# MAE
AR9.MAE = mean(abs(AR9.fore - y.t[3107:length(y.t)]))
ADL51.MAE = mean(abs(ADL51.fore - y.t[3107:length(y.t)]))
AR9.MAE < ADL51.MAE      # AR(9) 모형이 MAE 값이 더 작으므로
```

```
## [1] TRUE
```

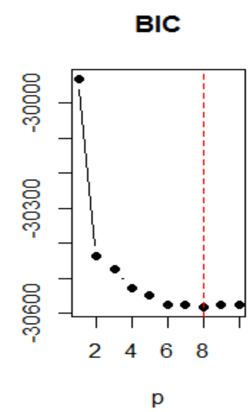
```r
# MSE
AR9.MSE = mean((AR9.fore - y.t[3107:length(y.t)])^2)
ADL51.MSE = mean((ADL51.fore - y.t[3107:length(y.t)])^2)
AR9.MSE < ADL51.MSE      # AR(9) 모형이 MSE 값이 더 작으므로
```

```
## [1] TRUE
```

```
result = matrix(c(AR9.MAE, ADL51.MAE, AR9.MSE, ADL51.MSE), nrow = 2, byrow = T)
row.names(result) = c("MAE", "MSE")
colnames(result) = c("AR9", "ADL(5,1)")
result    # AR(1) 모형이 더 잘 예측
```

```
##           AR9  ADL(5,1)
## MAE 0.6205788 0.6765098
## MSE 0.9326661 1.1859905
```

```
#============================#
#  SNP.RV = Y, VIX = X       #
#============================#
# MAKE DATA
y.t = rv.data$SNP.RV[1:nrow(rv.data)]      # SNP.RV
x.t = rv.data$VIX[1:(nrow(rv.data))]   # VIX
```




```
# AR 모형 BIC : p=8
AR.bic = c()
for( p in 1:10){
  AR.fit = arima(y.t, order = c(p,0,0))
  AR.bic[p] = AIC(AR.fit,k = log(length(y.t)))
}
par(mfrow=c(1,2))
plot(AR.bic, type = "b", pch =19, main = "BIC", ylab = "", xlab = "p")
abline(v = which.min(AR.bic), col=2, lty = 2) # AR(8)

y_t = y.t[9:length(y.t)]
y_t_1 = y.t[8:(length(y.t)-1)]
y_t_2 = y.t[7:(length(y.t)-2)]
y_t_3 = y.t[6:(length(y.t)-3)]
y_t_4 = y.t[5:(length(y.t)-4)]
y_t_5 = y.t[4:(length(y.t)-5)]
y_t_6 = y.t[3:(length(y.t)-6)]
y_t_7 = y.t[2:(length(y.t)-7)]
y_t_8 = y.t[1:(length(y.t)-8)]

x_t = x.t[9:length(x.t)]
x_t_1 = x.t[8:(length(x.t)-1)]
x_t_2 = x.t[7:(length(x.t)-2)]
x_t_3 = x.t[6:(length(x.t)-3)]
x_t_4 = x.t[5:(length(x.t)-4)]
x_t_5 = x.t[4:(length(x.t)-5)]
x_t_6 = x.t[3:(length(x.t)-6)]
x_t_7 = x.t[2:(length(x.t)-7)]
x_t_8 = x.t[1:(length(x.t)-8)]

y.data = cbind(y_t_1 = y_t_1, y_t_2= y_t_2, y_t_3 = y_t_3, y_t_4 = y_t_4,
               y_t_5 = y_t_5, y_t_6= y_t_6, y_t_7 = y_t_7, y_t_8 = y_t_8)
x.data = cbind(x_t_1 = x_t_1, x_t_2= x_t_2, x_t_3 = x_t_3, x_t_4 = x_t_4,
               x_t_5 = x_t_5, x_t_6= x_t_6, x_t_7 = x_t_7, x_t_8 = x_t_8)

# ADL 모형 BIC : p=6, q=6
ADL.bic = matrix(0, ncol = 8, nrow = 8)
```

```
for( p in 1:8){
  for(q in 1:8){
    ADL.fit = lm(y_t ~ y.data[,1:p] + x.data[,1:q])
    ADL.bic[p,q] = AIC(ADL.fit,k = log(length(y.t)))
  }
}
colnames(ADL.bic) = c("q=1", "q=2", "q=3", "q=4", "q=5", "q=6", "q=7", "q=8")
row.names(ADL.bic) = c("p=1", "p=2", "p=3", "p=4", "p=5", "p=6", "p=7", "p=8")
ADL.bic; min(ADL.bic)    # ADL(6,6)
```

```
##           q=1       q=2       q=3       q=4       q=5       q=6       q=7
## p=1 -30730.66 -30764.09 -30790.13 -30792.10 -30806.08 -30823.50 -30825.73
## p=2 -30836.12 -30943.35 -30972.21 -30966.69 -30964.98 -30972.28 -30966.61
## p=3 -30827.97 -30946.21 -31002.40 -30998.35 -30993.98 -30996.00 -30989.24
## p=4 -30822.93 -30942.19 -31013.12 -31021.53 -31018.83 -31017.89 -31009.99
## p=5 -30814.76 -30935.23 -31006.56 -31019.42 -31024.30 -31024.98 -31016.84
## p=6 -30809.18 -30933.90 -31010.01 -31023.47 -31037.82 -31057.77 -31050.08
## p=7 -30803.53 -30926.69 -31001.90 -31015.31 -31029.70 -31052.63 -31046.82
## p=8 -30795.44 -30920.92 -30997.84 -31012.95 -31029.58 -31053.31 -31051.16
##           q=8
## p=1 -30836.17
## p=2 -30967.10
## p=3 -30986.51
## p=4 -31005.71
## p=5 -31010.98
## p=6 -31042.91
## p=7 -31040.05
## p=8 -31051.37
```

```
## [1] -31057.77
```

```
## 1-step ahead forecasting
AR8.fore = c(); ADL66.fore = c()
for(i in 3106:(length(y.t)-1)){
  train.data = data.frame(y.t = y.t[1:i], x.t = x.t[1:i])

  y_t = train.data[9:nrow(train.data),1]
  y_t_1 = train.data[8:(nrow(train.data)-1),1]
  y_t_2 = train.data[7:(nrow(train.data)-2),1]
  y_t_3 = train.data[6:(nrow(train.data)-3),1]
  y_t_4 = train.data[5:(nrow(train.data)-4),1]
  y_t_5 = train.data[4:(nrow(train.data)-5),1]
  y_t_6 = train.data[3:(nrow(train.data)-6),1]
  y_t_7 = train.data[2:(nrow(train.data)-7),1]
  y_t_8 = train.data[1:(nrow(train.data)-8),1]

  x_t = train.data[9:nrow(train.data),2]
  x_t_1 = train.data[8:(nrow(train.data)-1),2]
  x_t_2 = train.data[7:(nrow(train.data)-2),2]
  x_t_3 = train.data[6:(nrow(train.data)-3),2]
  x_t_4 = train.data[5:(nrow(train.data)-4),2]
  x_t_5 = train.data[4:(nrow(train.data)-5),2]
  x_t_6 = train.data[3:(nrow(train.data)-6),2]
  x_t_7 = train.data[2:(nrow(train.data)-7),2]
  x_t_8 = train.data[1:(nrow(train.data)-8),2]
```

```r
  # AR(8)
  AR8.fit = lm(y_t ~ y_t_1 + y_t_2 + y_t_3 + y_t_4 +
                 y_t_5 + y_t_6 + y_t_7 + y_t_8)
  AR8.fore[i-3105] = sum(AR8.fit$coef*c(1, y_t[length(y_t)],
                                        y_t_1[length(y_t)],
                                        y_t_2[length(y_t)],
                                        y_t_3[length(y_t)],
                                        y_t_4[length(y_t)],
                                        y_t_5[length(y_t)],
                                        y_t_6[length(y_t)],
                                        y_t_7[length(y_t)]))
  # ADL(6,6)
  ADL.fit1 = lm(y_t ~
                  y_t_1 + y_t_2 +y_t_3 + y_t_4 + y_t_5 + y_t_6 +
                    x_t + x_t_1 + x_t_2 + x_t_3+ x_t_4 + x_t_5 );
  ADL66.fore[i-3105] = sum(ADL.fit1$coef*c(1, y_t[length(y_t)],
                                           y_t_1[length(y_t)],
                                           y_t_2[length(y_t)],
                                           y_t_3[length(y_t)],
                                           y_t_4[length(y_t)],
                                           y_t_5[length(y_t)],
                                           x_t[length(y_t)],
                                           x_t_1[length(y_t)],
                                           x_t_2[length(y_t)],
                                           x_t_3[length(y_t)],
                                           x_t_4[length(y_t)],
                                           x_t_5[length(y_t)] ))

}

# 1. VKOSPI 가 KOSPI 5 분 실현변동성을 GRANGER CAUSE 하는가
grangertest(y.t~x.t)

## Granger causality test
##
## Model 1: y.t ~ Lags(y.t, 1:1) + Lags(x.t, 1:1)
## Model 2: y.t ~ Lags(y.t, 1:1)
##   Res.Df Df      F    Pr(>F)
## 1   3554
## 2   3555 -1 1000.6 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

# 2. 예측력 비교
# AR(8)     VS   ADL(6,6)
# AR8.fore  VS   ADL66.fore
# MAE
AR8.MAE = mean(abs(AR8.fore - y.t[3107:length(y.t)]))
ADL66.MAE = mean(abs(ADL66.fore - y.t[3107:length(y.t)]))
AR8.MAE < ADL66.MAE      # AR(8) 모형이 MAE 값이 더 작음

## [1] TRUE

# MSE
AR8.MSE = mean((AR8.fore - y.t[3107:length(y.t)])^2)
```

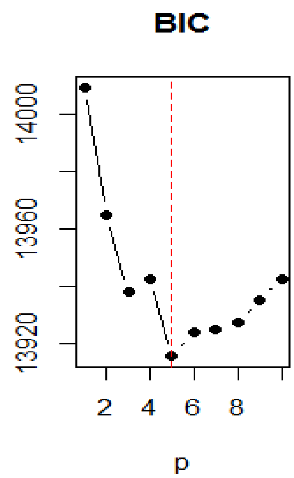
```r
ADL66.MSE = mean((ADL66.fore - y.t[3107:length(y.t)])^2)
AR8.MSE < ADL66.MSE      # AR(8) 모형이 MSE 값이 더 작음

## [1] TRUE

result = matrix(c(AR8.MAE, ADL66.MAE, AR8.MSE, ADL66.MSE), nrow = 2, byrow = T)
row.names(result) = c("MAE", "MSE")
colnames(result) = c("AR8", "ADL(6,6)")
result  # AR(8) 모형이 더 잘 예측

##              AR8     ADL(6,6)
## MAE 1.293314e-03 1.357345e-03
## MSE 3.976575e-06 4.643699e-06
```

```r
#=============================#
#     VIX = Y,    SNP.RV = X  #
#=============================#
# MAKE DATA
y.t = rv.data$VIX[1:nrow(rv.data)]       # VIX
x.t = rv.data$SNP.RV[1:(nrow(rv.data))]    # SNP.RV
```




```r
# AR 모형 BIC : p=5
AR.bic = c()
for( p in 1:10){
  AR.fit = arima(y.t, order = c(p,0,0))
  AR.bic[p] = AIC(AR.fit,k = log(length(y.t)))
}
par(mfrow=c(1,2))
plot(AR.bic, type = "b", pch =19, main = "BIC", ylab = "", xlab = "p")
abline(v = which.min(AR.bic), col=2, lty = 2) # AR(5)

y_t = y.t[6:length(y.t)]
y_t_1 = y.t[5:(length(y.t)-1)]
y_t_2 = y.t[4:(length(y.t)-2)]
y_t_3 = y.t[3:(length(y.t)-3)]
y_t_4 = y.t[2:(length(y.t)-4)]
y_t_5 = y.t[1:(length(y.t)-5)]

x_t = x.t[6:length(x.t)]
x_t_1 = x.t[5:(length(x.t)-1)]
x_t_2 = x.t[4:(length(x.t)-2)]
x_t_3 = x.t[3:(length(x.t)-3)]
x_t_4 = x.t[2:(length(x.t)-4)]
x_t_5 = x.t[1:(length(x.t)-5)]

y.data = cbind(y_t_1 = y_t_1, y_t_2= y_t_2, y_t_3 = y_t_3,
               y_t_4 = y_t_4, y_t_5 = y_t_5)
x.data = cbind(x_t_1 = x_t_1, x_t_2= x_t_2, x_t_3 = x_t_3,
               x_t_4 = x_t_4, x_t_5 = x_t_5)

# ADL 모형 BIC : p=5, q=2
ADL.bic = matrix(0, ncol = 5, nrow = 5)
for( p in 1:5){
```

```r
  for(q in 1:5){
    ADL.fit = lm(y_t ~ y.data[,1:p] + x.data[,1:q])
    ADL.bic[p,q] = AIC(ADL.fit,k = log(length(y.t)))
  }
}
colnames(ADL.bic) = c("q=1", "q=2", "q=3", "q=4", "q=5")
row.names(ADL.bic) = c("p=1", "p=2", "p=3", "p=4", "p=5")
ADL.bic; min(ADL.bic)    # ADL(5,2)
```

```
##          q=1      q=2      q=3      q=4      q=5
## p=1 13997.55 13996.56 13961.55 13959.49 13961.68
## p=2 13952.31 13926.94 13916.32 13915.06 13919.92
## p=3 13920.84 13894.67 13898.65 13903.59 13908.70
## p=4 13923.04 13899.70 13904.10 13910.71 13916.54
## p=5 13890.32 13877.14 13879.06 13886.47 13894.03
```

```
## [1] 13877.14
```

```r
## 1-step ahead forecasting
AR5.fore = c(); ADL52.fore = c()
for(i in 3106:(length(y.t)-1)){
  train.data = data.frame(y.t = y.t[1:i], x.t = x.t[1:i])

  y_t = train.data[6:nrow(train.data),1]
  y_t_1 = train.data[5:(nrow(train.data)-1),1]
  y_t_2 = train.data[4:(nrow(train.data)-2),1]
  y_t_3 = train.data[3:(nrow(train.data)-3),1]
  y_t_4 = train.data[2:(nrow(train.data)-4),1]
  y_t_5 = train.data[1:(nrow(train.data)-5),1]

  x_t = train.data[6:nrow(train.data),2]
  x_t_1 = train.data[5:(nrow(train.data)-1),2]
  x_t_2 = train.data[4:(nrow(train.data)-2),2]
  x_t_3 = train.data[3:(nrow(train.data)-3),2]
  x_t_4 = train.data[2:(nrow(train.data)-4),2]
  x_t_5 = train.data[1:(nrow(train.data)-5),2]

  # AR(5)
  AR5.fit = lm(y_t ~ y_t_1 + y_t_2 + y_t_3 + y_t_4 + y_t_5 )
  AR5.fore[i-3105] = sum(AR5.fit$coef*c(1, y_t[length(y_t)],
                                        y_t_1[length(y_t)],
                                        y_t_2[length(y_t)],
                                        y_t_3[length(y_t)],
                                        y_t_4[length(y_t)] ))
  # ADL(5,2)
  ADL.fit1 = lm(y_t ~
                y_t_1 + y_t_2 +y_t_3 + y_t_4 + y_t_5 +
                x_t + x_t_1 );
  ADL52.fore[i-3105] = sum(ADL.fit1$coef*c(1, y_t[length(y_t)],
                                           y_t_1[length(y_t)],
                                           y_t_2[length(y_t)],
                                           y_t_3[length(y_t)],
                                           y_t_4[length(y_t)],
                                           x_t[length(y_t)],
                                           x_t_1[length(y_t)] ))
}
```

```r
# 1. VKOSPI 가 KOSPI 5 분 실현변동성을 GRANGER CAUSE 하는가
grangertest(y.t~x.t)

## Granger causality test
##
## Model 1: y.t ~ Lags(y.t, 1:1) + Lags(x.t, 1:1)
## Model 2: y.t ~ Lags(y.t, 1:1)
##   Res.Df Df      F Pr(>F)
## 1   3554
## 2   3555 -1 1.0976 0.2949

# 2. 예측력 비교
# AR(5)     VS   ADL(5,2)
# AR5.fore  VS   ADL52.fore
# MAE
AR5.MAE = mean(abs(AR5.fore - y.t[3107:length(y.t)]))
ADL52.MAE = mean(abs(ADL52.fore - y.t[3107:length(y.t)]))
AR5.MAE < ADL52.MAE      # AR(5) 모형이 MAE 값이 더 작음

## [1] TRUE

# MSE
AR5.MSE = mean((AR5.fore - y.t[3107:length(y.t)])^2)
ADL52.MSE = mean((ADL52.fore - y.t[3107:length(y.t)])^2)
AR5.MSE < ADL52.MSE      # AR(5) 모형이 MSE 값이 더 작음

## [1] TRUE

result = matrix(c(AR5.MAE, ADL52.MAE, AR5.MSE, ADL52.MSE), nrow = 2, byrow = T)
row.names(result) = c("MAE", "MSE")
colnames(result) = c("AR5", "ADL(5,2)")
result # AR(5) 모형이 더 잘 예측

##           AR5  ADL(5,2)
## MAE 0.7955403 0.8435758
## MSE 1.4964288 1.6484881
```