Lab3 Q3

Michelle Kim, Sue Yang, Legg Yeung

August 2, 2017

```
# copied
library(moments)
library(psych)
```

```
## Warning: package 'psych' was built under R version 3.3.3
```

```
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 3.3.3
```

```
library(tseries)
```

```
## Warning: package 'tseries' was built under R version 3.3.3
```

```
library(effects)
```

```
## Warning: package 'effects' was built under R version 3.3.3
```

```
library(vars)
```

```
## Warning: package 'vars' was built under R version 3.3.3
## Loading required package: MASS
## Loading required package: strucchange
## Warning: package 'strucchange' was built under R version 3.3.3
## Loading required package: zoo
## Warning: package 'zoo' was built under R version 3.3.3
##
## Attaching package: 'zoo'
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
## Loading required package: sandwich
## Warning: package 'sandwich' was built under R version 3.3.3
## Loading required package: urca
## Warning: package 'urca' was built under R version 3.3.3
## Loading required package: lmtest
## Warning: package 'lmtest' was built under R version 3.3.3
```

```
unem.data = read.csv("UNRATENSA.csv", header = T)
auto.data = read.csv("TOTALNSA.csv", header = T)
```

```
unem.ts = ts(unem.data$UNRATENSA, frequency = 12, start = c(1948,1))
auto.ts = ts(auto.data$TOTALNSA, frequency = 12, start = c(1976,1))
```

```
unem.train = unem.data[0:816,]
unem.test = unem.data[817:834,]
unem.train.ts = ts(unem.train$UNRATENSA)
unem.test.ts = ts(unem.test$UNRATENSA)
```

Question 3: VAR

You also have data on automotive car sales. Use a VAR model to produce a 1 year forecast on both the unemployment rate and automotive sales for 2017 in the US.

Compare the 1 year forecast for unemployment produced by the VAR and SARIMA models, examining both the accuracy AND variance of the forecast. Do you think the addition of the automotive sales data helps? Why or why not?

Training VAR models

Continuing insights from the EDA, we attempt to train a model for each of the following differencing options:

- Both series first differenced and seasonal variable
- Both series seasonal differenced
- Both series first and seasonal differenced

```
combined.raw = ts.intersect(unem.ts, auto.ts)

combined.raw.train = window(combined.raw, end = c(2015, 12))
combined.raw.test = window(combined.raw, start = c(2016, 1))

unem.var.train = combined.raw.train[,1]
auto.var.train = combined.raw.train[,2]
unem.var.test = combined.raw.test[,1]
auto.var.test = combined.raw.test[,2]
```

```
# 1st differenced
unem.var.train.diff = diff(unem.var.train)
auto.var.train.diff = diff(auto.var.train)

# 12th differenced
unem.var.train.diff12 = diff(unem.var.train, lag = 12)
auto.var.train.diff12 = diff(auto.var.train, lag = 12)

# 1st & 12th differenced
unem.var.train.diff.diff12 = diff(diff(unem.var.train), lag = 12)
auto.var.train.diff.diff12 = diff(diff(auto.var.train), lag = 12)
```

Estimate Model Order

```
# 1st differenced & seasonal dummies
vars::VARselect(cbind(unem.var.train.diff, auto.var.train.diff),
                lag.max = 30, season = 12)
```

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

```
##     12     12      3     12
##
## $criteria
##                 1          2          3          4          5          6
## AIC(n)   6.077358   5.908156   5.816714   5.792824   5.779709   5.773954
## HQ(n)    6.178313   6.023533   5.946513   5.937045   5.938352   5.947019
## SC(n)    6.333475   6.200861   6.146008   6.158706   6.182179   6.213012
## FPE(n) 435.893801 368.049133 335.895625 327.976595 323.715984 321.873231
##                 7          8          9         10         11         12
## AIC(n)   5.764036   5.770100   5.757378   5.752039   5.686467   5.636140
## HQ(n)    5.951523   5.972010   5.973709   5.982793   5.931643   5.895738
## SC(n)    6.239682   6.282335   6.306200   6.337450   6.308466   6.294728
## FPE(n) 318.714432 320.673837 316.643488 314.984499 295.021494 280.572002
##                13         14         15         16         17         18
## AIC(n)   5.643104   5.640704   5.652517   5.662337   5.662471   5.669571
## HQ(n)    5.917124   5.929145   5.955380   5.979623   5.994179   6.015701
## SC(n)    6.338279   6.372467   6.420868   6.467277   6.503999   6.547688
## FPE(n) 282.567032 281.927736 285.320635 288.183970 288.274650 290.386181
##                19         20         21         22         23         24
## AIC(n)   5.680029   5.669705   5.683042   5.697468   5.705056   5.706180
## HQ(n)    6.040581   6.044679   6.072438   6.101286   6.123296   6.138843
## SC(n)    6.594733   6.620997   6.670923   6.721937   6.766113   6.803826
## FPE(n) 293.501816 290.554894 294.530325 298.891002 301.255621 301.688977
##                25         26         27         28         29         30
## AIC(n)   5.710710   5.722897   5.736465   5.749737   5.763830   5.769005
## HQ(n)    6.157795   6.184403   6.212394   6.240088   6.268603   6.288200
## SC(n)    6.844944   6.893718   6.943875   6.993735   7.044416   7.086179
## FPE(n) 303.160220 306.987132 311.299587 315.586610 320.203733 322.012063
```

```
# 12th differenced
vars::VARselect(cbind(unem.var.train.diff12, auto.var.train.diff12),
                lag.max = 30)
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##     26     14      4     26
##
## $criteria
##                 1          2          3          4          5          6
## AIC(n)   6.542704   6.511599   6.474749   6.429119   6.422979   6.423339
## HQ(n)    6.564769   6.548373   6.526234   6.495314   6.503884   6.518953
## SC(n)    6.598625   6.604800   6.605231   6.596882   6.628022   6.665662
## FPE(n) 694.161291 672.902573 648.560246 619.635187 615.848039 616.078085
##                 7          8          9         10         11         12
## AIC(n)   6.438604   6.451243   6.439144   6.443364   6.453997   6.384818
## HQ(n)    6.548929   6.576277   6.578888   6.597819   6.623161   6.568692
## SC(n)    6.718208   6.768127   6.793309   6.834810   6.882723   6.850824
## FPE(n) 625.566751 633.538535 625.938826 628.610219 635.358956 592.923849
##                13         14         15         16         17         18
## AIC(n)   6.142594   6.114890   6.127063   6.138595   6.150025   6.152342
## HQ(n)    6.341178   6.328185   6.355067   6.381309   6.407449   6.424476
## SC(n)    6.645881   6.655458   6.704911   6.753724   6.802434   6.842032
## FPE(n) 465.404476 452.722079 458.306079 463.667438 469.049251 470.195956
##                19         20         21         22         23         24
## AIC(n)   6.167907   6.182291   6.191168   6.202501   6.209688   6.158668
```

```
## HQ(n)    6.454751   6.483845   6.507431   6.533474   6.555372   6.519062
## SC(n)    6.894877   6.946542   6.992699   7.041313   7.085781   7.072042
## FPE(n) 477.637767 484.632487 489.036186 494.702111 498.372207 473.688212
##               25         26         27         28         29         30
## AIC(n)   6.095890   6.063245   6.079437   6.091389   6.098720   6.107708
## HQ(n)    6.470993   6.453058   6.483961   6.510623   6.532664   6.556361
## SC(n)    7.046543   7.051179   7.104652   7.153884   7.198496   7.244764
## FPE(n) 444.972170 430.793083 437.948987 443.349287 446.757233 450.947889
```

```
# 1st & 12th differenced
vars::VARselect(cbind(unem.var.train.diff.diff12, auto.var.train.diff.diff12),
                lag.max = 30)
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##     25     13     12     25
##
## $criteria
##                1          2          3          4          5          6
## AIC(n)   6.792267   6.689290   6.591205   6.552788   6.540800   6.552800
## HQ(n)    6.814372   6.726132   6.642783   6.619103   6.621852   6.648589
## SC(n)    6.848285   6.782652   6.721912   6.720840   6.746197   6.795542
## FPE(n) 890.931504 803.753057 728.662009 701.204570 692.855477 701.229668
##                7          8          9         10         11         12
## AIC(n)   6.557691   6.545845   6.543894   6.548317   6.495410   6.223039
## HQ(n)    6.668217   6.671107   6.683893   6.703053   6.664883   6.407248
## SC(n)    6.837778   6.863277   6.898670   6.940439   6.924877   6.689851
## FPE(n) 704.681091 696.399541 695.063610 698.171864 662.224659 504.359380
##               13         14         15         16         17         18
## AIC(n)   6.190978   6.196872   6.208379   6.218623   6.227509   6.245167
## HQ(n)    6.389924   6.410554   6.436798   6.461779   6.485401   6.517797
## SC(n)    6.695135   6.738373   6.787225   6.834814   6.881045   6.936048
## FPE(n) 488.477430 491.401874 497.132275 502.300804 506.840434 515.934530
##               19         20         21         22         23         24
## AIC(n)   6.259514   6.266115   6.279218   6.279393   6.237198   6.178473
## HQ(n)    6.546880   6.568218   6.596058   6.610969   6.583511   6.539523
## SC(n)    6.987740   7.031686   7.082134   7.119654   7.114803   7.093424
## FPE(n) 523.462889 527.011162 534.053466 534.246843 512.278271 483.169338
##               25         26         27         28         29         30
## AIC(n)   6.141623   6.156255   6.171134   6.183556   6.192209   6.201373
## HQ(n)    6.517410   6.546778   6.576394   6.603553   6.626942   6.650843
## SC(n)    7.093919   7.145895   7.198120   7.247887   7.293884   7.340393
## FPE(n) 465.801684 472.791461 480.015237 486.163861 490.549514 495.239864
```

Notice that the above VARselect results suggests very different number of lags to include by AICc vs BIC. This is because the two information criteria penalize by different terms. The R documentation shows the following formulation for AICc and BIC.

$$AIC(n) = \ln \hat{\sigma_k^2} + [2k + 2k(k+1)/(n-k-1)]$$

$$SC(n) = \ln \hat{\sigma_k^2} + kIn(n)$$

While first component in each formula are the same and refer to the log of sum square error, the second term, that is the penalty term are different. In general, as sample size n grows, BIC's penalty term grows

faster than AICc's penalty term with number of estimated lags k. Therefore, BIC is better at penalizing large samples more consistenly and heavily. Considering our sample size of 479, we should choose the orders estimated by BIC over AICc, if both model residuals exhibit white noise behavior. The estimated orders are:

- Both series first differenced and seasonal variable : (Order 3 by BIC) (Order 12 by AIC)
- Both series seasonal differenced : (Order 4 by BIC) (Order 26 by AIC)
  - Recall from EDA that peak of cross-correlation took place between lag 3-7
- Both series first and seasonal differenced : (Order 12 by BIC) (Order 25 by AIC)

Residuals Examination

```
# 1st differenced & seasonal dummies
var.diff.mod.lag3 = vars::VAR(cbind(unem.var.train.diff, auto.var.train.diff),
                              p = 3, season = 12)

var.diff.mod.lag12 = vars::VAR(cbind(unem.var.train.diff, auto.var.train.diff),
                              p = 12, season = 12)

# 12th differenced
var.diff12.mod.lag4 = vars::VAR(cbind(unem.var.train.diff12,
                                      auto.var.train.diff12),
                                p = 4)

var.diff12.mod.lag26 = vars::VAR(cbind(unem.var.train.diff12,
                                       auto.var.train.diff12),
                              p = 26)

# 1st & 12th differenced
var.diff.diff12.mod.lag12 = vars::VAR(cbind(unem.var.train.diff.diff12,
                                            auto.var.train.diff.diff12),
                              p = 12)

var.diff.diff12.mod.lag25 = vars::VAR(cbind(unem.var.train.diff.diff12,
                                             auto.var.train.diff.diff12),
                              p = 25)
```

We perform the Portmanteau Test on the model residuals to detect autocorrelation. Test hypothesis is as follows:

- Ho : The residuals are not serially correlated
- Ha : The residuals are serially correlated
- The null hypothesis is only rejected for the first differenced model with seasonal variable and 12th lags.

```
# 1st differenced & seasonal dummies
vars::serial.test(var.diff.mod.lag3, lags.pt = 30, type = "PT.adjusted")
```

```
##
##  Portmanteau Test (adjusted)
##
## data:  Residuals of VAR object var.diff.mod.lag3
## Chi-squared = 268.06, df = 108, p-value = 1.332e-15
```

```
vars::serial.test(var.diff.mod.lag12, lags.pt = 30, type = "PT.adjusted")
```

```
##
```

```
##  Portmanteau Test (adjusted)
##
## data:  Residuals of VAR object var.diff.mod.lag12
## Chi-squared = 80.814, df = 72, p-value = 0.2232
```

```
# 12th differenced
vars::serial.test(var.diff12.mod.lag4, lags.pt = 30, type = "PT.adjusted")
```

```
##
##  Portmanteau Test (adjusted)
##
## data:  Residuals of VAR object var.diff12.mod.lag4
## Chi-squared = 278.37, df = 104, p-value < 2.2e-16
```

```
vars::serial.test(var.diff12.mod.lag26, lags.pt = 30, type = "PT.adjusted")
```

```
##
##  Portmanteau Test (adjusted)
##
## data:  Residuals of VAR object var.diff12.mod.lag26
## Chi-squared = 49.329, df = 16, p-value = 2.931e-05
```

```
# 1st & 12th differenced
vars::serial.test(var.diff.diff12.mod.lag12, lags.pt = 30, type = "PT.adjusted")
```

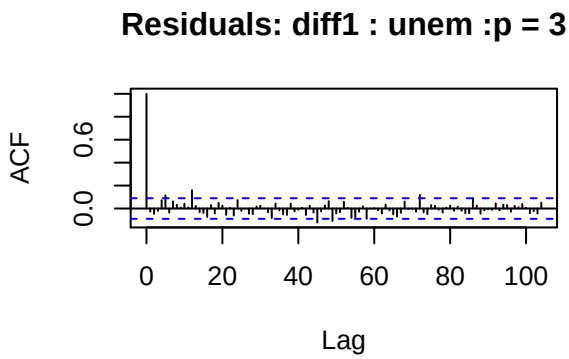
```
##
##  Portmanteau Test (adjusted)
##
## data:  Residuals of VAR object var.diff.diff12.mod.lag12
## Chi-squared = 171.39, df = 72, p-value = 4.407e-10
```

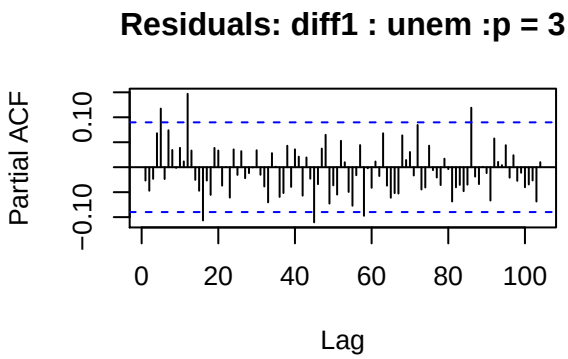
```
vars::serial.test(var.diff.diff12.mod.lag25, lags.pt = 30, type = "PT.adjusted")
```

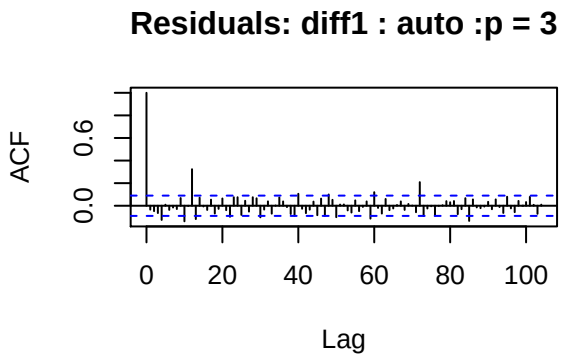
```
##
##  Portmanteau Test (adjusted)
##
## data:  Residuals of VAR object var.diff.diff12.mod.lag25
## Chi-squared = 56.225, df = 20, p-value = 2.689e-05
```

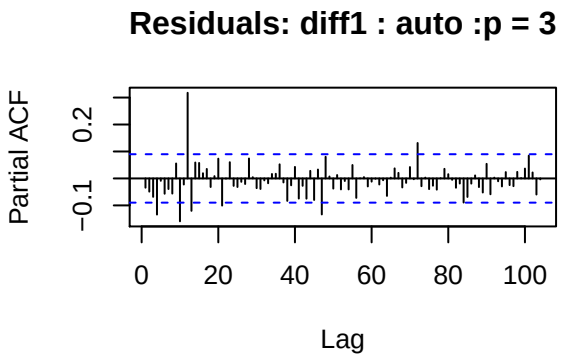
```
# 1st differenced & seasonal dummies
# p = 3
par(mfrow = c(2,2))
acf(residuals(var.diff.mod.lag3)[,1],104, main = "")
title("Residuals: diff1 : unem :p = 3")
pacf(residuals(var.diff.mod.lag3)[,1],104,main = "")
title("Residuals: diff1 : unem :p = 3")

acf(residuals(var.diff.mod.lag3)[,2],104, main = "")
title("Residuals: diff1 : auto :p = 3")
pacf(residuals(var.diff.mod.lag3)[,2],104,main = "")
title("Residuals: diff1 : auto :p = 3")
```



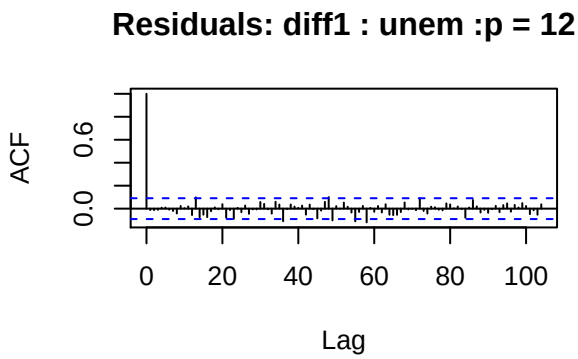


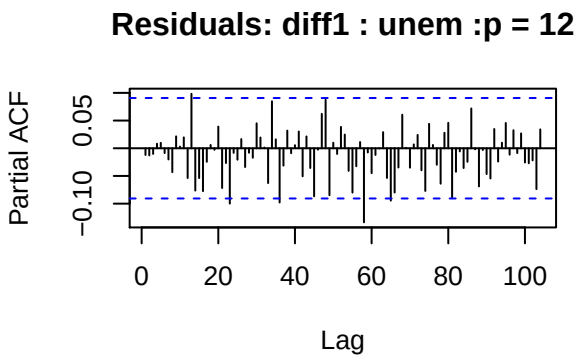


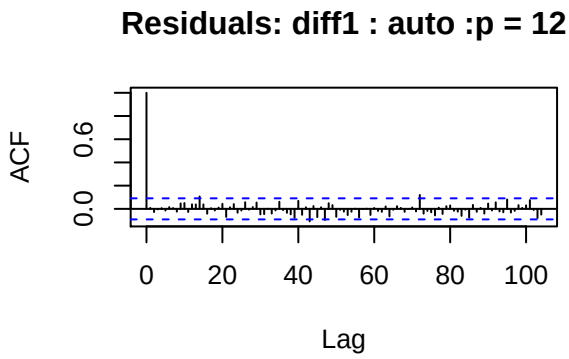


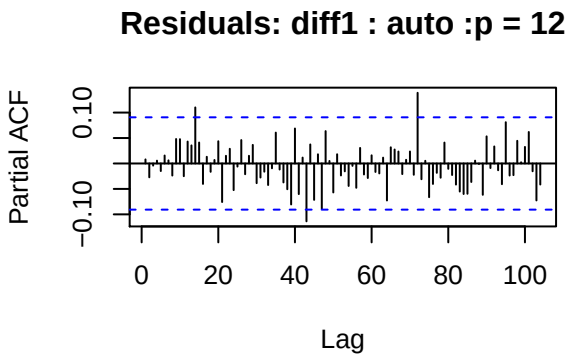

```r
# 1st differenced & seasonal dummies
# p = 12
par(mfrow = c(2,2))
acf(residuals(var.diff.mod.lag12)[,1],104, main = "")
title("Residuals: diff1 : unem :p = 12")
pacf(residuals(var.diff.mod.lag12)[,1],104,main = "")
title("Residuals: diff1 : unem :p = 12")

acf(residuals(var.diff.mod.lag12)[,2],104, main = "")
title("Residuals: diff1 : auto :p = 12")
pacf(residuals(var.diff.mod.lag12)[,2],104,main = "")
title("Residuals: diff1 : auto :p = 12")
```



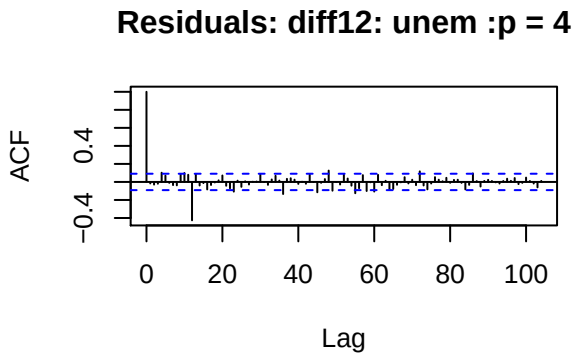


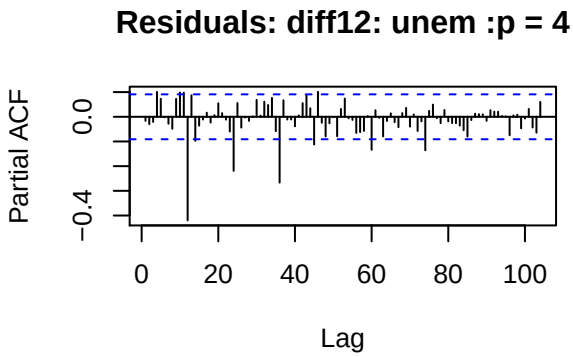


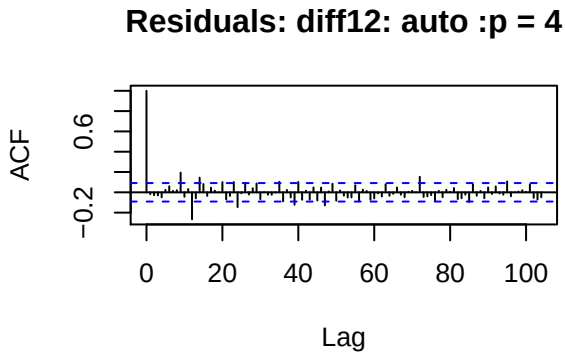


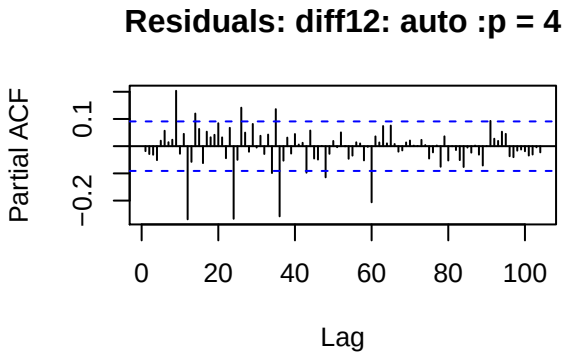

```r
# seasonal differenced
# p = 4
par(mfrow = c(2,2))
acf(residuals(var.diff12.mod.lag4)[,1],104, main = "")
title("Residuals: diff12: unem :p = 4")
pacf(residuals(var.diff12.mod.lag4)[,1],104,main = "")
title("Residuals: diff12: unem :p = 4")

acf(residuals(var.diff12.mod.lag4)[,2],104, main = "")
title("Residuals: diff12: auto :p = 4")
pacf(residuals(var.diff12.mod.lag4)[,2],104,main = "")
title("Residuals: diff12: auto :p = 4")
```



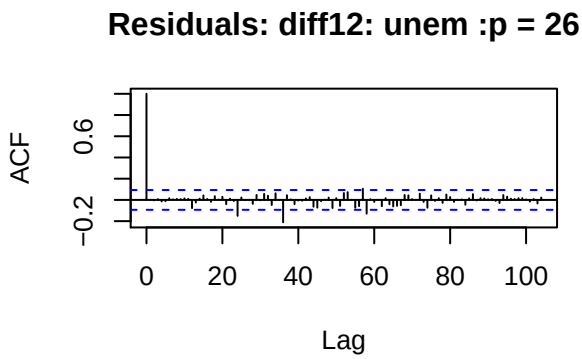


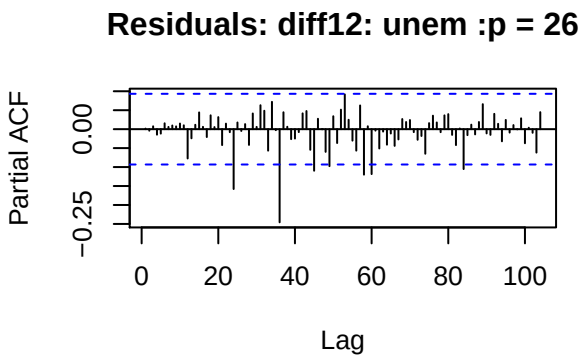


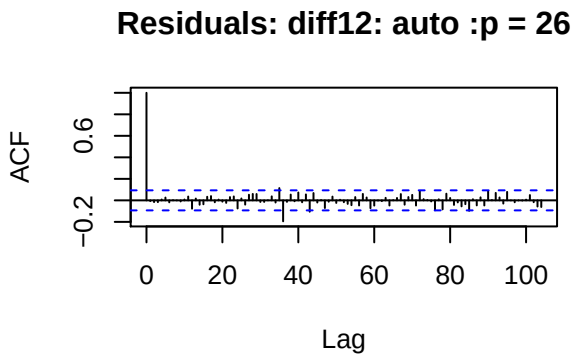


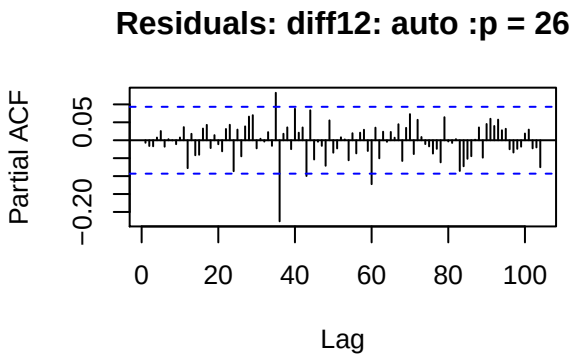

```r
# seasonal differenced
# p = 26
par(mfrow = c(2,2))
acf(residuals(var.diff12.mod.lag26)[,1],104, main = "")
title("Residuals: diff12: unem :p = 26")
pacf(residuals(var.diff12.mod.lag26)[,1],104,main = "")
title("Residuals: diff12: unem :p = 26")

acf(residuals(var.diff12.mod.lag26)[,2],104, main = "")
title("Residuals: diff12: auto :p = 26")
pacf(residuals(var.diff12.mod.lag26)[,2],104,main = "")
title("Residuals: diff12: auto :p = 26")
```



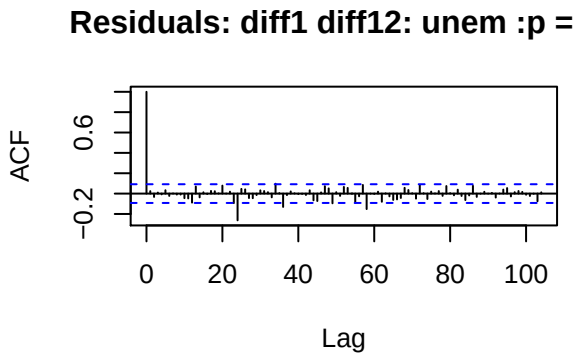


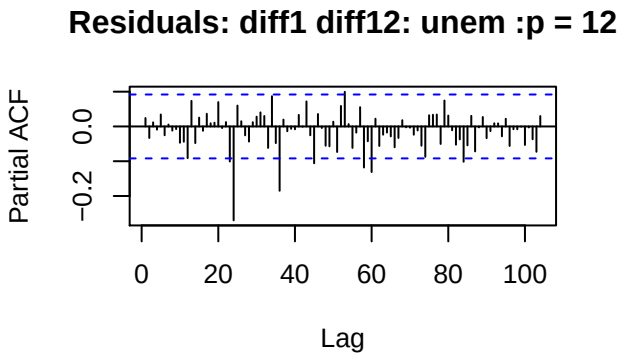


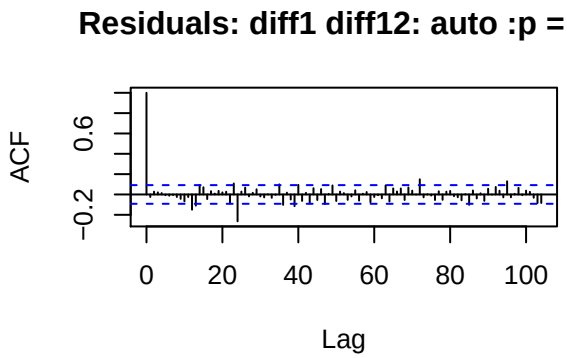


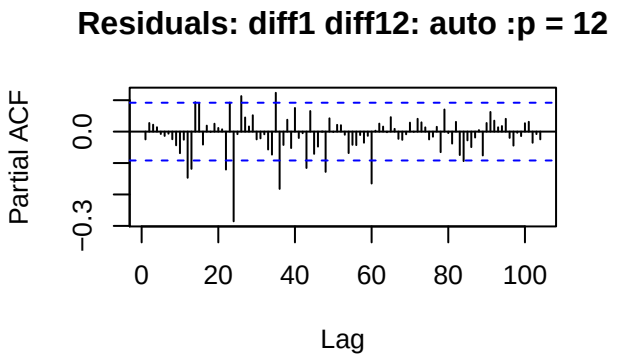

```r
# 1st differenced & seasonal differenced
# p = 12
par(mfrow = c(2,2))
acf(residuals(var.diff.diff12.mod.lag12)[,1],104, main = "")
title("Residuals: diff1 diff12: unem :p = 12")
pacf(residuals(var.diff.diff12.mod.lag12)[,1],104,main = "")
title("Residuals: diff1 diff12: unem :p = 12")

acf(residuals(var.diff.diff12.mod.lag12)[,2],104, main = "")
title("Residuals: diff1 diff12: auto :p = 12")
pacf(residuals(var.diff.diff12.mod.lag12)[,2],104,main = "")
title("Residuals: diff1 diff12: auto :p = 12")
```



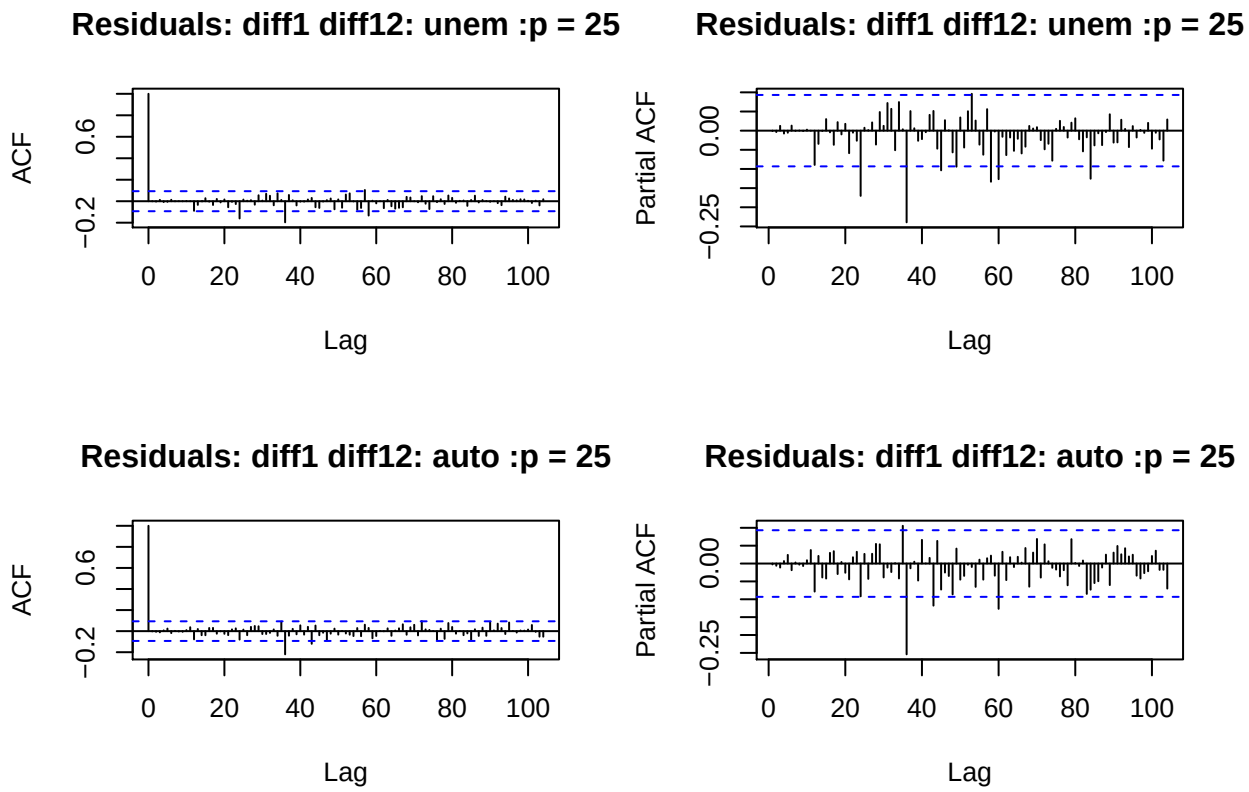







```r
# 1st differenced & seasonal differenced
# p = 25
par(mfrow = c(2,2))
acf(residuals(var.diff.diff12.mod.lag25)[,1],104, main = "")
title("Residuals: diff1 diff12: unem :p = 25")
pacf(residuals(var.diff.diff12.mod.lag25)[,1],104,main = "")
title("Residuals: diff1 diff12: unem :p = 25")

acf(residuals(var.diff.diff12.mod.lag25)[,2],104, main = "")
title("Residuals: diff1 diff12: auto :p = 25")
pacf(residuals(var.diff.diff12.mod.lag25)[,2],104,main = "")
title("Residuals: diff1 diff12: auto :p = 25")
```



From the residual plots above, the model with first differenced series and seasonal variables perform better in general. Lag order 12 gives residuals that's closest white noise. We reduced it down to lag order 11 with similar white noise pattern, but the Portmanteau Test rejected the null hypothesis that the residuals are not serially correlated. Model of lag order 10 and 9 starts to show significant pacfs at lag 12 therefore we stop search down from there. Time plots and residual plots for lag order 11 and 10 are given below.

```
var.diff.mod.lag11 = vars::VAR(cbind(unem.var.train.diff, auto.var.train.diff),
                               p = 11, season = 12)
vars::serial.test(var.diff.mod.lag11, lags.pt = 30, type = "PT.adjusted")
```

```
##
##  Portmanteau Test (adjusted)
##
## data:  Residuals of VAR object var.diff.mod.lag11
## Chi-squared = 101.01, df = 76, p-value = 0.02914
```

```
var.diff.mod.lag10 = vars::VAR(cbind(unem.var.train.diff, auto.var.train.diff),
                               p = 10, season = 12)
vars::serial.test(var.diff.mod.lag10, lags.pt = 30, type = "PT.adjusted")
```

```
##
##  Portmanteau Test (adjusted)
##
## data:  Residuals of VAR object var.diff.mod.lag10
## Chi-squared = 135.61, df = 80, p-value = 0.000104
```

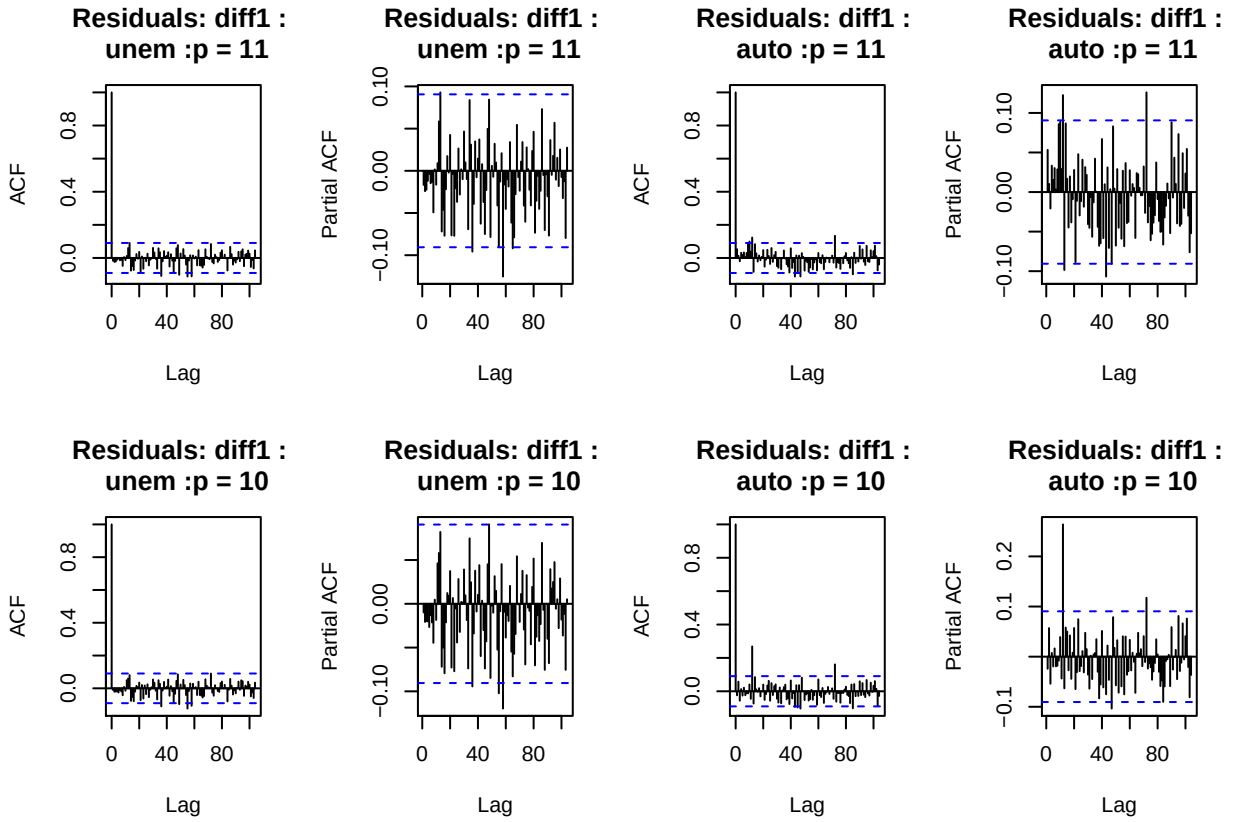
```
# 1st differenced & seasonal dummies
# p = 11
```

```
par(mfrow = c(2,4))
acf(residuals(var.diff.mod.lag11)[,1],104, main = "")
title("Residuals: diff1 : \n unem :p = 11")
pacf(residuals(var.diff.mod.lag11)[,1],104,main = "")
title("Residuals: diff1 : \n unem :p = 11")

acf(residuals(var.diff.mod.lag11)[,2],104, main = "")
title("Residuals: diff1 : \n auto :p = 11")
pacf(residuals(var.diff.mod.lag11)[,2],104,main = "")
title("Residuals: diff1 : \n auto :p = 11")

# p = 10
acf(residuals(var.diff.mod.lag10)[,1],104, main = "")
title("Residuals: diff1 : \n unem :p = 10")
pacf(residuals(var.diff.mod.lag10)[,1],104,main = "")
title("Residuals: diff1 : \n unem :p = 10")

acf(residuals(var.diff.mod.lag10)[,2],104, main = "")
title("Residuals: diff1 : \n auto :p = 10")
pacf(residuals(var.diff.mod.lag10)[,2],104,main = "")
title("Residuals: diff1 : \n auto :p = 10")
```



Based on the Portmanteau Tests and residual results, we proceed with the following three models. In all three, both raw series are first differenced and seasonal variables (as indicators) are included in the VAR model.

- VAR(3), VAR(11) and VAR(12)

Examine in-sample fit

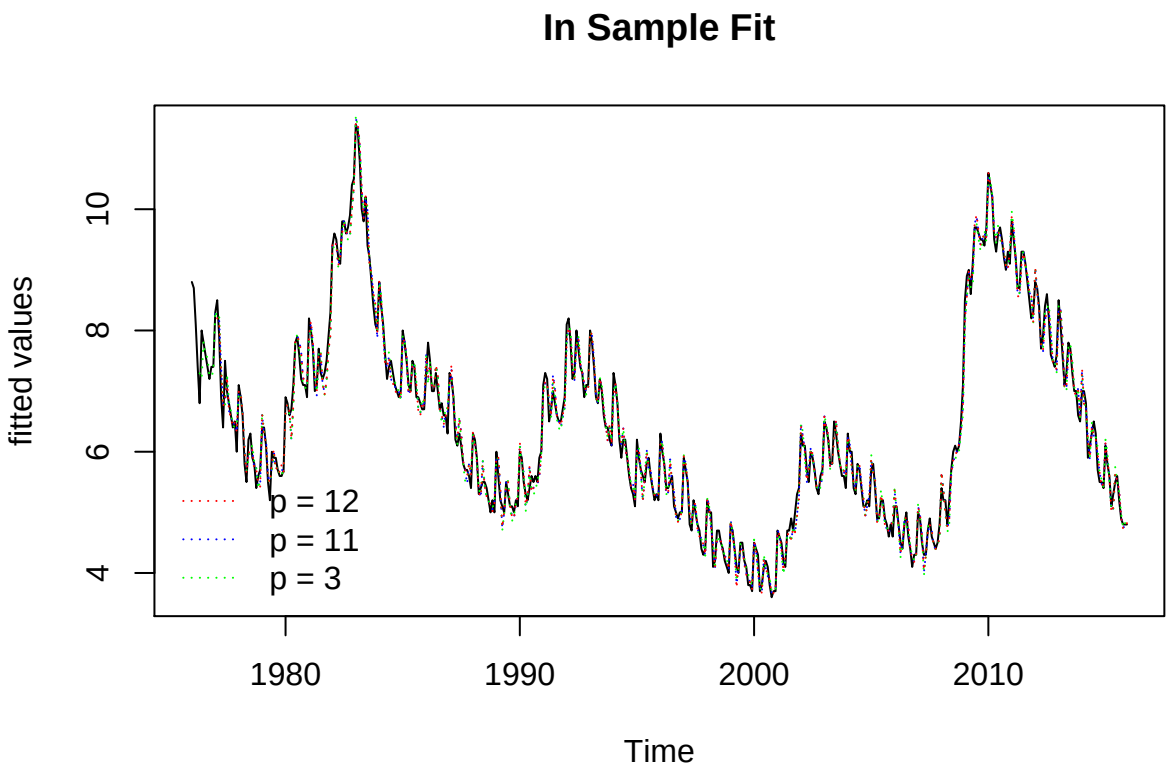
```
# Get in-sample-fit by manually adding differenced estimates
get_insampfit_delta = function(mod, train){
  na.pad = length(train) - length(fitted(mod)[,1])
  delta <-c(rep(NA,na.pad),fitted(mod)[,1])
  insampfit = rep(NA,length(train))
  for (i in (na.pad+1):480) insampfit[i] <- train[i-1] + delta[i]
  insampfit
}
# var.diff.mod.lag12$y[,1]
```

```
insampfit.lag12 = get_insampfit_delta(var.diff.mod.lag12, unem.var.train)
insampfit.lag11 = get_insampfit_delta(var.diff.mod.lag11, unem.var.train)
insampfit.lag3 = get_insampfit_delta(var.diff.mod.lag3, unem.var.train)

insampfit.lag12 = ts(insampfit.lag12, start = c(1976,1), frequency = 12)
insampfit.lag11 = ts(insampfit.lag11, start = c(1976,1), frequency = 12)
insampfit.lag3 =  ts(insampfit.lag3, start = c(1976,1), frequency = 12)
```

```
plot(unem.var.train, main = "", ylab = "fitted values")
lines(insampfit.lag11, col = "blue", lty = "dotted")
lines(insampfit.lag3, col = "green", lty = "dotted")
lines(insampfit.lag12, col = "red", lty = "dotted")
title("In Sample Fit")

legend("bottomleft", legend = c("p = 12", "p = 11", "p = 3"),
       lty = c("dotted","dotted", "dotted"), bty = "n",
       col = c("red", "blue", "green"))
```



The in-sample-fit for all three models are very close to the raw series. To further discriminate them, we examine their out-of-sample performance next.

Compare models based on out of sample errors uptil Jun 2017

```
# Get out-of-sample-performance by manually adding differenced estimates
# set X_t-1 obs as base
base = window(combined.raw, start = c(2015,12))[,1]
# Function
get_outsamp = function(mod, base, ahead){
  pred = predict(mod, n.ahead = ahead)$fcst$unem.var.train.diff
  delta.estimate = pred[,1]
  lower.ci = pred[,2]
  upper.ci = pred[,3]
  var.fore = rep(0,18)
  var.low.ci = rep(0,18)
  var.upp.ci = rep(0,18)
  for (i in 1:18) var.fore[i] <- base[i] + delta.estimate[i]
  for (i in 1:18) var.low.ci[i] <- base[i] + lower.ci[i]
  for (i in 1:18) var.upp.ci[i] <- base[i] + upper.ci[i]
  data.frame(var.fore,var.low.ci,var.upp.ci)
}
```

```
f.var.3 = get_outsamp(var.diff.mod.lag3, base, 18)
f.var.11 = get_outsamp(var.diff.mod.lag11, base, 18)
```

```r
f.var.12 = get_outsamp(var.diff.mod.lag12, base, 18)
```

```r
var.fore.df = data.frame("year" = as.numeric(floor(time(unem.var.test))),
             "month" = c(seq(1,12),seq(1,6)),
             "VAR3.est" = f.var.3$var.fore,
             "VAR3.err" = f.var.3$var.fore - as.numeric(unem.var.test),
             "VAR11.est" = f.var.11$var.fore,
             "VAR11.err" = f.var.11$var.fore - as.numeric(unem.var.test),
             "VAR12.est" = f.var.12$var.fore,
             "VAR12.err" = f.var.12$var.fore - as.numeric(unem.var.test) )

var.fore.df
```

```
##    year month VAR3.est     VAR3.err VAR11.est    VAR11.err VAR12.est
## 1  2016     1 5.621538  0.321538258  5.535232  0.235232221  5.539000
## 2  2016     2 5.170461 -0.029539481  5.129731 -0.070268514  5.103836
## 3  2016     3 4.945519 -0.154481485  4.922263 -0.177736786  4.912951
## 4  2016     4 4.591826 -0.108174249  4.512829 -0.187171475  4.507045
## 5  2016     5 4.650086  0.150086482  4.694008  0.194007781  4.696482
## 6  2016     6 4.971358 -0.128642488  4.818766 -0.281233625  4.792894
## 7  2016     7 5.049099 -0.050901065  5.062902 -0.037097732  5.062423
## 8  2016     8 4.861781 -0.138219150  4.829834 -0.170166461  4.798711
## 9  2016     9 4.817715  0.017715031  4.784713 -0.015286714  4.764793
## 10 2016    10 4.672266 -0.027733920  4.662989 -0.037010867  4.658773
## 11 2016    11 4.770360  0.370359571  4.725756  0.325755722  4.705915
## 12 2016    12 4.394766 -0.105234304  4.451277 -0.048723444  4.427971
## 13 2017     1 5.392585  0.292584804  5.347755  0.247755277  5.341262
## 14 2017     2 4.994263  0.094262624  4.981403  0.081403130  4.972461
## 15 2017     3 4.648712  0.048711942  4.633748  0.033747730  4.634662
## 16 2017     4 4.092220 -0.007780154  4.082557 -0.017442921  4.069027
## 17 2017     5 4.066310 -0.033689518  4.089880 -0.010120199  4.094404
## 18 2017     6 4.574833  0.074833218  4.503284  0.003284395  4.483601
##       VAR12.err
## 1   0.238999864
## 2  -0.096163848
## 3  -0.187048880
## 4  -0.192955419
## 5   0.196481988
## 6  -0.307106409
## 7  -0.037576563
## 8  -0.201289249
## 9  -0.035206560
## 10 -0.041226819
## 11  0.305915000
## 12 -0.072028593
## 13  0.241262228
## 14  0.072460961
## 15  0.034661522
## 16 -0.030972640
## 17 -0.005595562
## 18 -0.016398607
```

```r
cat("VAR(12) RMSE: ", sqrt(mean((var.fore.df$VAR12.err)^2)), "\n")
```

```
## VAR(12) RMSE:  0.1634807
```

```r
cat("VAR(11) RMSE: ", sqrt(mean((var.fore.df$VAR11.err)^2)), "\n")
```

```
## VAR(11) RMSE:  0.1587485
```

```r
cat("VAR(3) RMSE: ", sqrt(mean((var.fore.df$VAR3.err)^2)), "\n")
```

```
## VAR(3) RMSE:  0.1587414
```

In terms of out-of-sample comparison, the VAR(3) performs marginally better than the other two. This is a small contradiction with the residual plot studies and serial correlation test. Instead, the RMSE results align with the preferred lag order estimated by BIC. Recall that the RMSE for our SARIMA(2,1,1)(1,0,1) is 0.1472144, so the SARIMA model is still better in terms of out-of-sample performance.

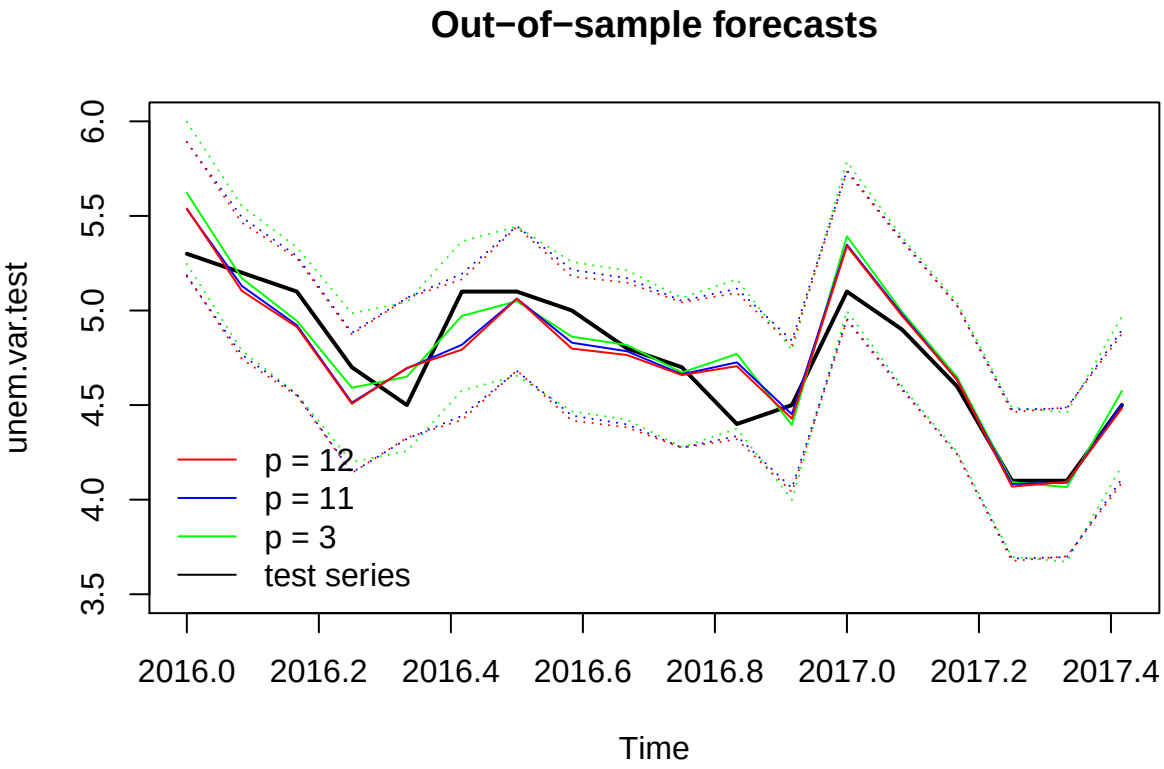
The three VAR models perform very closely, as illustrated by the forecast plot below. The VAR(3) forecast appears more sensitive to the observations at June 2016, which is probably the reason that its RMSE is lower than the other two models. Notice that the confidnce bound of all three VAR models are narrower and more consistent than the SARIMA(2,1,1)(1,0,1) model.

```r
plot(unem.var.test, ylim = c(3.5,6),
     main = "Out-of-sample forecasts", lwd = 2)

lines(y = f.var.3$var.fore,x = as.numeric(time(unem.var.test)),col = "green")
lines(y = f.var.11$var.fore,x = as.numeric(time(unem.var.test)),col = "blue")
lines(y = f.var.12$var.fore,x = as.numeric(time(unem.var.test)),col = "red")

lines(y = f.var.3$var.low.ci,
      x = as.numeric(time(unem.var.test)),col = "green", lty = "dotted")
lines(y = f.var.3$var.upp.ci,
      x = as.numeric(time(unem.var.test)),col = "green", lty = "dotted")
lines(y = f.var.11$var.low.ci,
      x = as.numeric(time(unem.var.test)),col = "blue", lty = "dotted")
lines(y = f.var.11$var.upp.ci,
      x = as.numeric(time(unem.var.test)),col = "blue", lty = "dotted")
lines(y = f.var.12$var.low.ci,
      x = as.numeric(time(unem.var.test)),col = "red", lty = "dotted")
lines(y = f.var.12$var.upp.ci,
      x = as.numeric(time(unem.var.test)),col = "red", lty = "dotted")

legend("bottomleft", legend = c("p = 12","p = 11","p = 3","test series"),
       lty = c("solid","solid", "solid", "solid"), bty = "n",
       col = c("red", "blue", "green", "black"))
```



Final Model

Given that the residuals of the VAR(3) model has very mild serial correlation, marginally better RMSE and much lower lag order than the VAR(11) and VAR(12) models, we should strongly consider it as the final choice. In the following, we examine the significance and interpretation of its coefficients.

```
summary(var.diff.mod.lag3)
```

```
##
## VAR Estimation Results:
## ========================
## Endogenous variables: unem.var.train.diff, auto.var.train.diff
## Deterministic variables: const
## Sample size: 476
## Log Likelihood: -2730.571
## Roots of the characteristic polynomial:
## 0.6688 0.6688 0.6649 0.5593 0.5017 0.5017
## Call:
## vars::VAR(y = cbind(unem.var.train.diff, auto.var.train.diff),
##     p = 3, season = 12L)
##
##
## Estimation results for equation unem.var.train.diff:
## ====================================================
## unem.var.train.diff = unem.var.train.diff.l1 + auto.var.train.diff.l1 + unem.var.train.diff.l2 + auto
##
```

```
##                           Estimate Std. Error t value Pr(>|t|)
## unem.var.train.diff.l1 -2.955e-03  4.599e-02  -0.064 0.948787
## auto.var.train.diff.l1 -3.424e-04  8.867e-05  -3.862 0.000129 ***
## unem.var.train.diff.l2  1.641e-01  4.509e-02   3.639 0.000305 ***
## auto.var.train.diff.l2 -2.699e-04  9.653e-05  -2.796 0.005387 **
## unem.var.train.diff.l3  1.591e-01  4.534e-02   3.510 0.000493 ***
## auto.var.train.diff.l3 -1.394e-04  8.986e-05  -1.552 0.121400
## const                  -2.672e-03  8.803e-03  -0.303 0.761647
## sd1                    -1.075e+00  6.544e-02 -16.422  < 2e-16 ***
## sd2                    -1.304e+00  5.878e-02 -22.184  < 2e-16 ***
## sd3                    -1.455e+00  7.243e-02 -20.094  < 2e-16 ***
## sd4                    -8.404e-01  5.847e-02 -14.373  < 2e-16 ***
## sd5                    -2.696e-01  5.344e-02  -5.045 6.54e-07 ***
## sd6                    -8.708e-01  5.601e-02 -15.548  < 2e-16 ***
## sd7                    -1.240e+00  4.894e-02 -25.331  < 2e-16 ***
## sd8                    -1.176e+00  5.438e-02 -21.621  < 2e-16 ***
## sd9                    -1.034e+00  5.022e-02 -20.580  < 2e-16 ***
## sd10                   -7.967e-01  4.586e-02 -17.373  < 2e-16 ***
## sd11                   -9.107e-01  4.853e-02 -18.767  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.1918 on 458 degrees of freedom
## Multiple R-Squared: 0.779,   Adjusted R-squared: 0.7708
## F-statistic: 94.99 on 17 and 458 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation auto.var.train.diff:
## ====================================================
## auto.var.train.diff = unem.var.train.diff.l1 + auto.var.train.diff.l1 + unem.var.train.diff.l2 + auto
##
##                           Estimate Std. Error t value Pr(>|t|)
## unem.var.train.diff.l1 -59.89492   23.58542  -2.539 0.011431 *
## auto.var.train.diff.l1  -0.61689    0.04548 -13.564  < 2e-16 ***
## unem.var.train.diff.l2 -20.10760   23.12561  -0.869 0.385032
## auto.var.train.diff.l2  -0.48635    0.04951  -9.824  < 2e-16 ***
## unem.var.train.diff.l3 -25.04746   23.25502  -1.077 0.282012
## auto.var.train.diff.l3  -0.24112    0.04609  -5.232 2.56e-07 ***
## const                    1.68305    4.51490   0.373 0.709487
## sd1                    254.58641   33.56128   7.586 1.86e-13 ***
## sd2                    429.29052   30.14939  14.239  < 2e-16 ***
## sd3                    238.22932   37.14846   6.413 3.56e-10 ***
## sd4                    323.25969   29.98816  10.780  < 2e-16 ***
## sd5                    194.73408   27.40927   7.105 4.64e-12 ***
## sd6                    119.09833   28.72532   4.146 4.03e-05 ***
## sd7                    160.63555   25.10271   6.399 3.87e-10 ***
## sd8                     27.27200   27.88862   0.978 0.328645
## sd9                     89.38955   25.75781   3.470 0.000569 ***
## sd10                    17.67736   23.52082   0.752 0.452700
## sd11                   184.94625   24.88768   7.431 5.32e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
##
## Residual standard error: 98.36 on 458 degrees of freedom
## Multiple R-Squared: 0.6542,  Adjusted R-squared: 0.6414
## F-statistic: 50.97 on 17 and 458 DF,  p-value: < 2.2e-16
##
##
##
## Covariance matrix of residuals:
##                      unem.var.train.diff auto.var.train.diff
## unem.var.train.diff             0.03678             -0.3442
## auto.var.train.diff            -0.34415           9675.6369
##
## Correlation matrix of residuals:
##                      unem.var.train.diff auto.var.train.diff
## unem.var.train.diff             1.00000            -0.01824
## auto.var.train.diff            -0.01824             1.00000
```

From the model summary, we observe that:

1) coefficients of auto sales lags are generally significant and negative, this confirms our intuition in the EDA that auto sales leads unemployment rate in opposite directions.

2) coefficients of unemployment lags on unemployment rate are close to zero or generally positive, this is somewhat intuitive. If unemployment rate was high 2-3 months ago, it will probably stay high especially in times of persistent economic boom or busts.

3) coefficients of unemployment lags are not significant on autosales, this confirms our EDA insight that auto sales leads unemployment, not the other way round.

(!!!! Missing VAR(3) Model specification, and checking for stationarity using determinant of theta matrix)

Comparing the VAR(3) model against the SARIMA(2,1,1)(1,0,1) model:

In terms of in sample fit, both models approximates the raw series well, as depicted by the earlier time plots. In terms of out-of-sample 18 steps ahead performance, both models had close RMSE measures. The two models are so similar in these accuracy measures possibly because : 1)Both VAR and SARIMA models perform step by step forecast that take advantage of autocorrelations with the series's own lags, with heavy reliance on the past observations in the raw series. This is very different from the linear regression model, which can only predict with the time indexes, information is much more restricted. 2) Both the VAR(3) and SARIMA(2,1,1)(1,0,1) models are integrated by order 1 (first differenced series as input) and incorporate AR components of similar order.

However, variance of their forecasts are very different. Variance of the SARIMA model grow drastically with time because its step-by-step forecast has to depend heavier and heavier on estimated instead of observed lag values as time increases. Each estimated lag value contributes its own variance(uncertainty) towards the next forecast. On the other hand, although the VAR model has similar forecast mechanism, has much more consistant forecast variance. Because by accounting for cross-correlation in the raw series and thus modeling residuals as bivariate white noise, VAR mechanism manages to impose heavier structure on the residuals(thus coefficient estimates and forecasts) so they seem more restricted than the SARIMA model.

One may be surpised that the VAR(3) model does not perform much better than the SARIMA(2,1,1)(1,0,1) model, given that we have useful information of auto sales that leads unemployment rate. This is because: 1) The VAR model limits us to auto-regressive terms, whereas the SARIMA model allow moving average terms to directly account for moving average components. 2) The VAR model can only account for seasonal patterns with indicator variables, we are deprived of the option to accound for seasonal auto-regressive or seasonal moving average terms.

In conclusion, to forecast unemployment rate, one can consider a SARIMA model for more unbiased estimators, or a VAR model for more precise predictions.