Vector-Autoregression HW4

Austin Lee

February 26, 2019

```
library(readxl)
```

```
## Warning: package 'readxl' was built under R version 3.5.2
```

```
library(vars)
```

```
## Warning: package 'vars' was built under R version 3.5.2
## Loading required package: MASS
## Loading required package: strucchange
## Warning: package 'strucchange' was built under R version 3.5.2
## Loading required package: zoo
## Warning: package 'zoo' was built under R version 3.5.2
##
## 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 3.5.2
## Loading required package: urca
## Warning: package 'urca' was built under R version 3.5.2
## Loading required package: lmtest
## Warning: package 'lmtest' was built under R version 3.5.2
```

```
library('tseries')
```

```
## Warning: package 'tseries' was built under R version 3.5.2
```

```
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 3.5.2
```

Problem 1: VAR modeling

```
data<-read_excel("Copy of Chapter11_exercises_data.xls", sheet = "Exercise 1 - 10")
```

```
## New names:
## * `` -> `..8`
## * `` -> `..9`
```

```
MSA1 <- (ts(data$GSJ[1:120], start = 1975, frequency = 4))
MSA2 <- (ts(data$GSF[1:120], start = 1975, frequency = 4))
```

```r
MSA1o <- as.numeric(ts(data$GSJ[1:120], start = 1975, frequency = 4))#San Jose
```

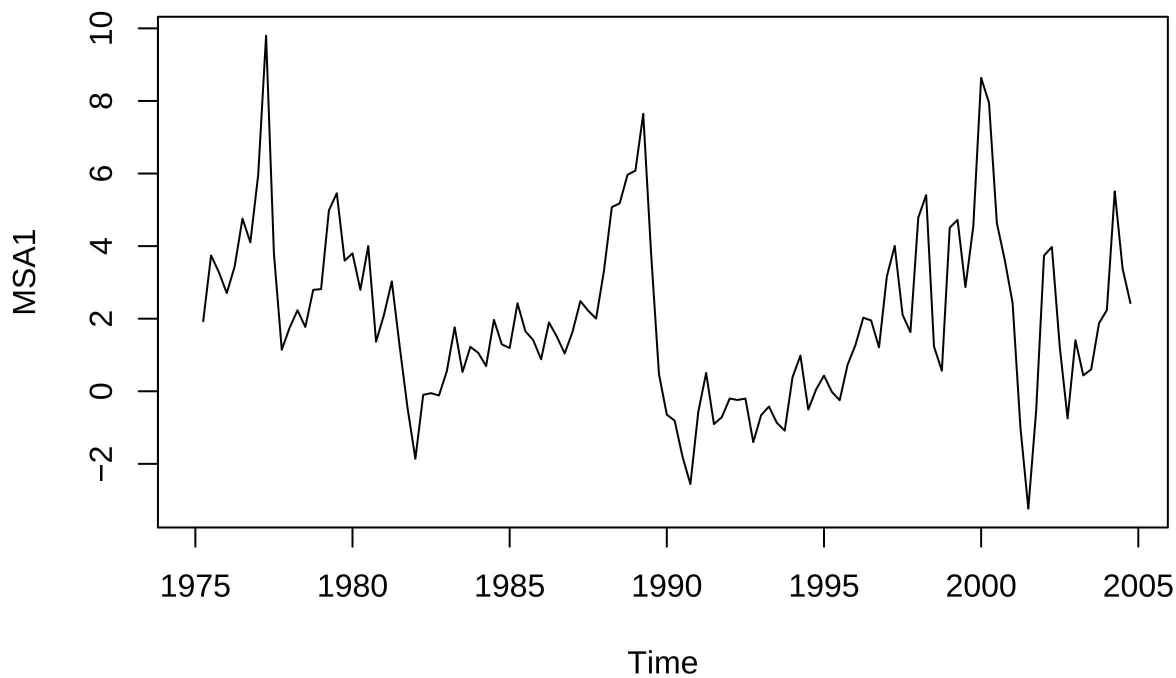
```
## Warning: NAs introduced by coercion
```

```r
MSA2o <- as.numeric(ts(data$GSF[1:120], start = 1975, frequency = 4))#San Francisco
```

```
## Warning: NAs introduced by coercion
```

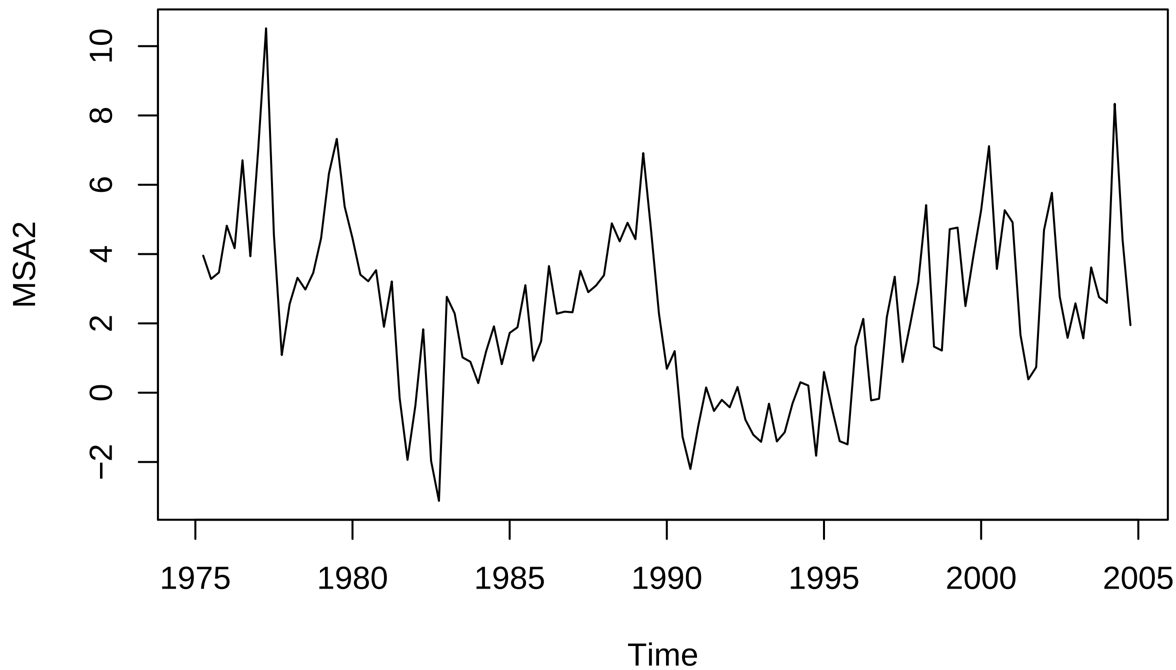
```r
plot(MSA1)
```

```
## Warning in xy.coords(x, NULL, log = log, setLab = FALSE): NAs introduced by
## coercion

## Warning in xy.coords(x, y): NAs introduced by coercion
```




```r
plot(MSA2)
```

```
## Warning in xy.coords(x, NULL, log = log, setLab = FALSE): NAs introduced by
## coercion

## Warning in xy.coords(x, NULL, log = log, setLab = FALSE): NAs introduced by
## coercion
```



The plots of these two time series are very similar. In general, it appears there are larger amptitudes for the San Francisco time series.

```
comboMSA= (na.remove(cbind(MSA1o,MSA2o)))
tot_comboMSA = data.frame(comboMSA)
VARselect(tot_comboMSA)
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      3      3      3      3
##
## $criteria
##                1        2        3         4         5        6         7
## AIC(n) 1.079820 1.062953 0.867412 0.8691033 0.8773325 0.940032 0.8732427
## HQ(n)  1.139900 1.163085 1.007597 1.0493410 1.0976231 1.200376 1.1736390
## SC(n)  1.227968 1.309865 1.213090 1.3135460 1.4205403 1.582005 1.6139807
## FPE(n) 2.944233 2.895279 2.381584 2.3865693 2.4077947 2.565911 2.4030952
##                8         9        10
## AIC(n) 0.934031 0.9574882 0.9855751
## HQ(n)  1.274480 1.3379902 1.4061300
## SC(n)  1.773534 1.8957563 2.0226083
## FPE(n) 2.557843 2.6239523 2.7056122
```

From our Varselect, we can see that the best model according to AIC, BIC, HQ, and FPE is at lag, p = 3.

```
var_model<-VAR(tot_comboMSA, p = 3)
summary(var_model)
```

```
##
## VAR Estimation Results:
## =========================
## Endogenous variables: MSA1o, MSA2o
## Deterministic variables: const
## Sample size: 116
## Log Likelihood: -374.15
## Roots of the characteristic polynomial:
## 0.9002 0.6279 0.6279 0.5964 0.3438 0.3438
## Call:
## VAR(y = tot_comboMSA, p = 3)
##
##
## Estimation results for equation MSA1o:
## ======================================
## MSA1o = MSA1o.l1 + MSA2o.l1 + MSA1o.l2 + MSA2o.l2 + MSA1o.l3 + MSA2o.l3 + const
##
##          Estimate Std. Error t value Pr(>|t|)
## MSA1o.l1  1.04964    0.13910   7.546 1.44e-11 ***
## MSA2o.l1 -0.09279    0.12470  -0.744   0.4584
## MSA1o.l2 -0.34053    0.15796  -2.156   0.0333 *
## MSA2o.l2 -0.17436    0.12837  -1.358   0.1772
## MSA1o.l3  0.01353    0.13793   0.098   0.9221
## MSA2o.l3  0.35488    0.12445   2.852   0.0052 **
## const     0.35424    0.19875   1.782   0.0775 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 1.453 on 109 degrees of freedom
## Multiple R-Squared: 0.6413,  Adjusted R-squared: 0.6216
## F-statistic: 32.48 on 6 and 109 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation MSA2o:
## ======================================
## MSA2o = MSA1o.l1 + MSA2o.l1 + MSA1o.l2 + MSA2o.l2 + MSA1o.l3 + MSA2o.l3 + const
##
##          Estimate Std. Error t value Pr(>|t|)
## MSA1o.l1  0.71855    0.14937   4.811 4.86e-06 ***
## MSA2o.l1  0.20342    0.13391   1.519 0.131627
## MSA1o.l2 -0.41743    0.16962  -2.461 0.015423 *
## MSA2o.l2 -0.07820    0.13785  -0.567 0.571693
## MSA1o.l3 -0.05052    0.14812  -0.341 0.733679
## MSA2o.l3  0.52371    0.13364   3.919 0.000156 ***
## const     0.32344    0.21343   1.515 0.132551
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 1.56 on 109 degrees of freedom
## Multiple R-Squared: 0.6332,  Adjusted R-squared: 0.613
## F-statistic: 31.36 on 6 and 109 DF,  p-value: < 2.2e-16
##
```

```
##
##
## Covariance matrix of residuals:
##       MSA1o MSA2o
## MSA1o 2.111 1.637
## MSA2o 1.637 2.434
##
## Correlation matrix of residuals:
##        MSA1o  MSA2o
## MSA1o 1.0000 0.7222
## MSA2o 0.7222 1.0000
```

According to our var model, it appears as though we have many statistically insignificant coefficients when predicting price growth for San Jose. Using price growth in San Francisco as well as San Jose, we determine that good predictor of growth would include lags for previous growth for San Jose at 1 and 2 and San Francisco growth at lag 3. Similarly, price growth for San Francisco resembles the same coefficients used for San Jose. The two models have similar R^2 adjusted values of around .62.

Problem 2: Granger Casuality

```r
grangertest(MSA1o~MSA2o, order = 3)
```

```
## Granger causality test
##
## Model 1: MSA1o ~ Lags(MSA1o, 1:3) + Lags(MSA2o, 1:3)
## Model 2: MSA1o ~ Lags(MSA1o, 1:3)
##   Res.Df Df      F  Pr(>F)
## 1    109
## 2    112 -3 2.8403 0.04128 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

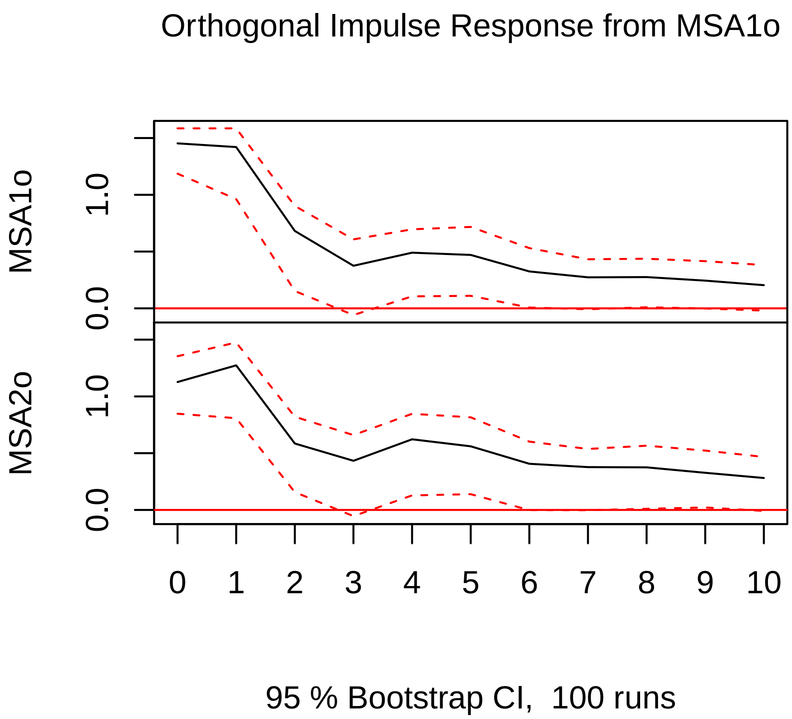
```r
grangertest(MSA2o~MSA1o, order = 3)
```

```
## Granger causality test
##
## Model 1: MSA2o ~ Lags(MSA2o, 1:3) + Lags(MSA1o, 1:3)
## Model 2: MSA2o ~ Lags(MSA2o, 1:3)
##   Res.Df Df      F    Pr(>F)
## 1    109
## 2    112 -3 7.8278 8.852e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

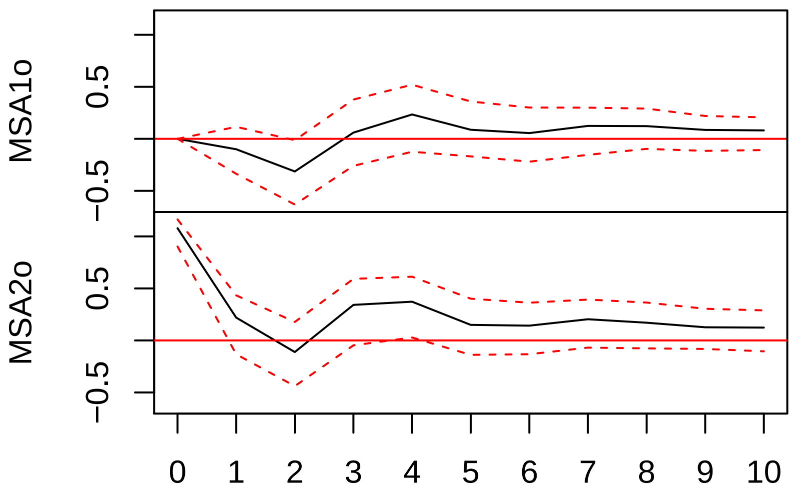
The Granger casuality test tests whether there is predictive casuality between the two sets of time series. At lag 3, we look whether either San Jose price growth may have some predictive casuality with San Francisco and vice Versa. After running the test, we see that we would reject the null hypothesis for both at the 5% level, indicating that there is casuality between predicting San Jose price growth with San Francisco and vice versa. Since they are both statistically significant, the granger casuality test itself is inconclusive.

Problem 3: Impulse Response Functions

```r
plot(irf(var_model))
```


Orthogonal Impulse Response from MSA1o
MSA1o
MSA2o
0.0
1.0
0.0
1.0
0
1
2
3
4
5
6
7
8
9
10
95 % Bootstrap CI, 100 runs

Orthogonal Impulse Response from MSA2o



95 % Bootstrap CI, 100 runs

When looking at the Impluse Response Function for our Var model, we see that predictive powers of price growth in San Jose has a large impact on San Jose(larger amplitude) and San Francisco prices at lag 1, which begins to decay from that point on. This would mean that recent increases or decreases in price growth are a better predictor of future price growth in San Jose. When looking at the predictive powers of price growth in San Francisco, we see that it has little impact for San Jose initially but reverts, with small amplitude, a few times above and below 0. When looking at San Francisco price growth with lagged San Francisco price growth, there is immediate casual relationship at lag 1, which begins to decline from that point on.

Problem 4: VAR VS ARIMA VS Actual Observations

```
predict(var_model, n.ahead=1)
```

```
## $MSA1o
##                fcst     lower    upper       CI
## MSA1o.fcst 3.837096 0.9893583 6.684833 2.847738
##
## $MSA2o
##                fcst   lower    upper       CI
## MSA2o.fcst 4.797472 1.73948 7.855464 3.057992
```

```
print(paste("Real observations for MSA1:", data$GSJ[121]))
```

```
## [1] "Real observations for MSA1: 3.2182520000000001"
```

```
print(paste("Real observations for MSA2:", data$GSF[121]))
```

```
## [1] "Real observations for MSA2: 5.6466190000000003"
```

```r
auto.arima(MSA1o)#best one is ARIMA(1,0,2)
```

```
## Series: MSA1o
## ARIMA(1,0,2) with non-zero mean
##
## Coefficients:
##          ar1     ma1      ma2    mean
##       0.7619  0.2281  -0.2806  2.0123
## s.e.  0.1171  0.1554   0.1447  0.5204
##
## sigma^2 estimated as 2.201:  log likelihood=-214.34
## AIC=438.68   AICc=439.21   BIC=452.57
```

```r
auto.arima(MSA2o)#best one is ARIMA(2,1,2)
```

```
## Series: MSA2o
## ARIMA(2,1,2)
##
## Coefficients:
##          ar1      ar2      ma1     ma2
##       0.0401  -0.8274  -0.2489  0.5655
## s.e.  0.0917   0.1194   0.1419  0.1709
##
## sigma^2 estimated as 2.879:  log likelihood=-228.21
## AIC=466.42   AICc=466.96   BIC=480.27
```

```r
MSA1_ARIMA_mod <- arima(MSA1o, order = c(1,0,2))
MSA2_ARIMA_mod <- arima(MSA2o, order = c(2,1,2))

predict_MSA1_ARIMA_mod1 <- predict(MSA1_ARIMA_mod, n.ahead = 1)
predict_MSA2_ARIMA_mod2 <- predict(MSA2_ARIMA_mod, n.ahead =1)

print(paste("Prediction using ARIMA for MSA1:", predict_MSA1_ARIMA_mod1[1]))
```

```
## [1] "Prediction using ARIMA for MSA1: 3.03887915872747"
```

```r
print(paste("Upper bound of ARIMA for MSA1:",
            as.numeric(predict_MSA1_ARIMA_mod1[1])+
              as.numeric(predict_MSA1_ARIMA_mod1[2])*1.96))
```

```
## [1] "Upper bound of ARIMA for MSA1: 5.89738280942995"
```

```r
print(paste("Lower bound of ARIMA for MSA1:",
            as.numeric(predict_MSA1_ARIMA_mod1[1])-
              as.numeric(predict_MSA1_ARIMA_mod1[2])*1.96))
```

```
## [1] "Lower bound of ARIMA for MSA1: 0.180375508024988"
```

```r
print(paste("Prediction using ARIMA for MSA2:", predict_MSA2_ARIMA_mod2[1]))
```

```
## [1] "Prediction using ARIMA for MSA2: 3.77866278585655"
```

```r
print(paste("Upper bound of ARIMA for MSA2:",
            as.numeric(predict_MSA2_ARIMA_mod2[1])+
              as.numeric(predict_MSA2_ARIMA_mod2[2])*1.96))
```

```
## [1] "Upper bound of ARIMA for MSA2: 7.0476434626992"
```

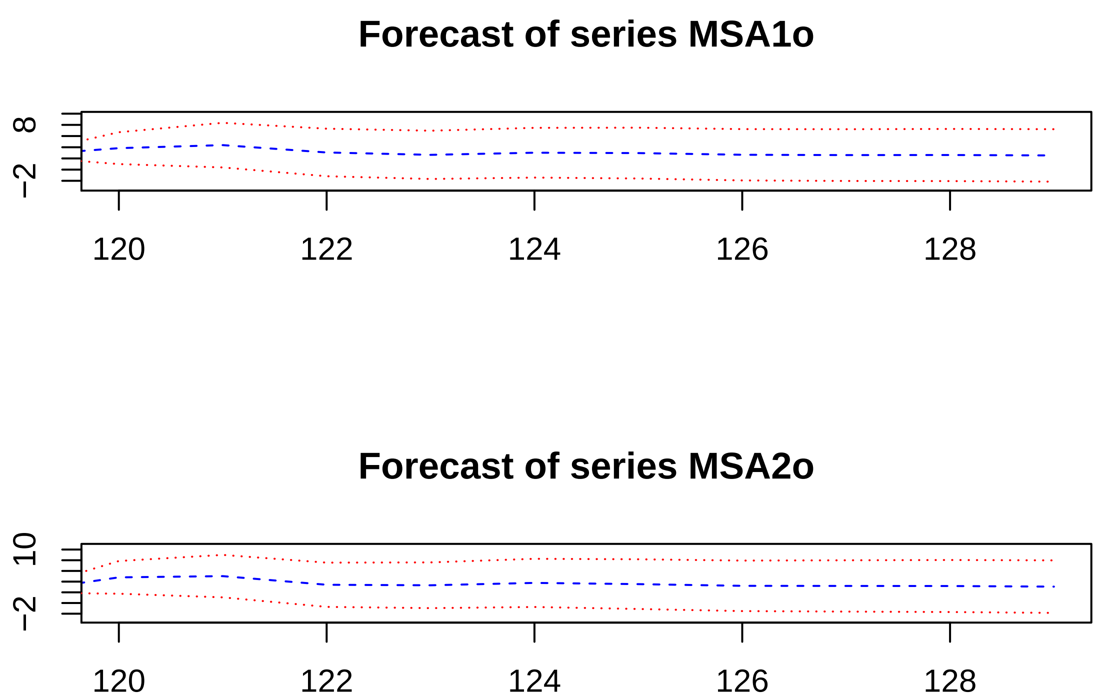
```r
print(paste("Lower bound of ARIMA for MSA2:",
            as.numeric(predict_MSA2_ARIMA_mod2[1])-
              as.numeric(predict_MSA2_ARIMA_mod2[2])*1.96))
```

```
## [1] "Lower bound of ARIMA for MSA2: 0.509682109013895"
```

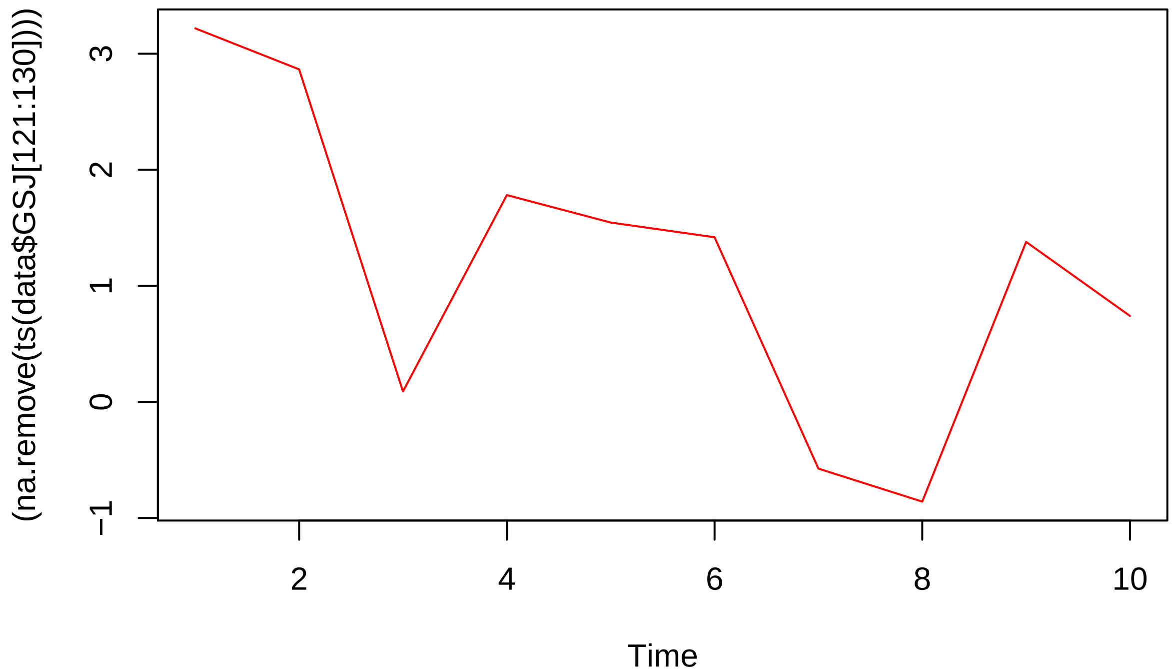
For our given prediction and confidence interval, we are within the levels of real price growth for San Jose and San Francisco. For San Jose, our forecast was a 3.87% price growth with an upper bound of 6.68% and lower bound of .989 for Q1 2005 ; the real price growth was 3.2182%. For San Francisco, our forecast was 4.79% with a lower bound 1.74% and upper bound of 7.86%; the real price growth was 5.64%. Using the univariate ARIMA model, our prediction for San Jose was 3.04%, with the actual observation for San Jose at 3.21%, which is within their confidence interval at the 5% level. Using the univariate ARIMA model, the prediction for San Francisco was 3.79% with the actual value lying between its confidence interval as well. Both models did a decent job of predicting 1 step ahead.

Problem 5: Predicing 10 steps out

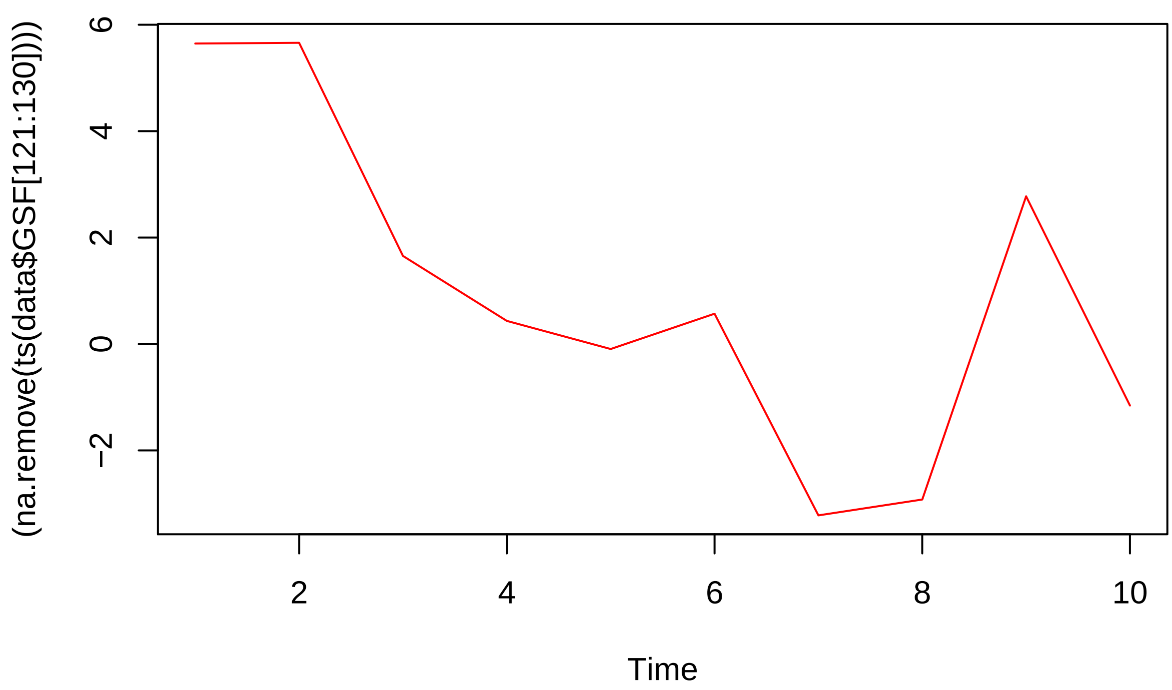
```r
library('tseries')

plot(predict(var_model, n.ahead=10), xlim= c(120,129))
```




```r
plot((na.remove(ts(data$GSJ[121:130]))), col = 'red')
```




```r
plot((na.remove(ts(data$GSF[121:130]))), col = 'red')
```




```r
predict(var_model,n.ahead=10)
```

```
## $MSA1o
##            fcst      lower    upper       CI
##  [1,] 3.837096  0.9893583 6.684833 2.847738
##  [2,] 4.378618  0.3911705 8.366065 3.987447
##  [3,] 3.065073 -1.1849304 7.315077 4.250004
##  [4,] 2.641721 -1.6729660 6.956407 4.314687
##  [5,] 3.023349 -1.4208165 7.467515 4.444166
##  [6,] 2.953252 -1.5888474 7.495352 4.542100
##  [7,] 2.654231 -1.9336528 7.242115 4.587884
##  [8,] 2.596513 -2.0289033 7.221930 4.625416
##  [9,] 2.612759 -2.0501702 7.275689 4.662929
## [10,] 2.536604 -2.1538169 7.227024 4.690421
##
## $MSA2o
##            fcst      lower    upper       CI
##  [1,] 4.797472  1.7394804 7.855464 3.057992
##  [2,] 5.026414  1.0560185 8.996810 3.970395
##  [3,] 3.413153 -0.7253326 7.551639 4.138486
##  [4,] 3.317977 -0.9593916 7.595346 4.277369
##  [5,] 3.761425 -0.7458838 8.268733 4.507309
##  [6,] 3.531498 -1.1170124 8.180009 4.648510
##  [7,] 3.211920 -1.5126577 7.936499 4.724578
##  [8,] 3.192246 -1.6064639 7.990956 4.798710
##  [9,] 3.179708 -1.6864990 8.045915 4.866207
```

```
## [10,] 3.062204 -1.8523160 7.976724 4.914520
```

```
for (i in (1:10)){
  print(paste("San Jose Actual:",data$GSJ[i+120]))
}
```

```
## [1] "San Jose Actual: 3.2182520000000001"
## [1] "San Jose Actual: 2.8652880000000001"
## [1] "San Jose Actual: 0.089889999999999998"
## [1] "San Jose Actual: 1.78193"
## [1] "San Jose Actual: 1.5454410000000001"
## [1] "San Jose Actual: 1.418787"
## [1] "San Jose Actual: -0.57453299999999996"
## [1] "San Jose Actual: -0.85894599999999999"
## [1] "San Jose Actual: 1.37886"
## [1] "San Jose Actual: 0.740846"
```

```
for (i in (1:10)){
  print(paste("San Francisco Actual:",data$GSF[i+120]))
}
```

```
## [1] "San Francisco Actual: 5.6466190000000003"
## [1] "San Francisco Actual: 5.660196"
## [1] "San Francisco Actual: 1.654981"
## [1] "San Francisco Actual: 0.43387300000000001"
## [1] "San Francisco Actual: -0.094186000000000006"
## [1] "San Francisco Actual: 0.56904999999999994"
## [1] "San Francisco Actual: -3.220736"
## [1] "San Francisco Actual: -2.9230450000000001"
## [1] "San Francisco Actual: 2.7754889999999999"
## [1] "San Francisco Actual: -1.1571370000000001"
```

When looking at a multistep forecast, compared to the actual observed data, we can see that the two first initial steps do a good job of capturing the overall data. However, it does not capture the data afterwards. The actual data is well within the confidence interval, but the predicted points are staying constant at around the overall mean; this is due to the properties of multiforecasting.

```
MSA3o <- as.numeric(ts(data$GAL[1:120], start = 1975, freq = 4))
```

```
## Warning: NAs introduced by coercion
```

Using the same excel file, we look at the pricing growth rates of Albany-Schenectady-Troy metropolitan areas. In problem 6, we will do the same VAR and Granger casuality analysis with the San Jose pricing growth data.

Problem 6: GAL and GSJ

```
combo_GAL_GSJ <- data.frame(na.remove(cbind(MSA1o, MSA3o)))
VARselect((combo_GAL_GSJ))
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      4      3      1      4
##
## $criteria
##                1        2        3        4        5        6
## AIC(n)  2.433870  2.380675  2.303751 2.294589  2.314422  2.327849
## HQ(n)   2.493949  2.480808  2.443936 2.474827  2.534712  2.588192
```

```
## SC(n)   2.582018  2.627588  2.649428 2.739032  2.857630  2.969822
## FPE(n) 11.403244 10.813597 10.015209 9.927839 10.133032 10.279279
##                7         8         9        10
## AIC(n)  2.314784  2.366068  2.401403  2.417299
## HQ(n)   2.615180  2.706517  2.781905  2.837853
## SC(n)   3.055522  3.205571  3.339671  3.454332
## FPE(n) 10.158376 10.710254 11.118348 11.325441
```

After combining both sets of data into a data frame and performing the VARselect function, there is a difference in the optimial model chosen by AIC, BIC, and FPE. Since AIC tends to over parameterize, we will choose BIC's optimal model and use a lag of 1 instead.

```r
VAR_model_GAL_GSJ<- VAR(combo_GAL_GSJ,p=1)
summary(VAR_model_GAL_GSJ)
```

```
##
## VAR Estimation Results:
## =========================
## Endogenous variables: MSA1o, MSA3o
## Deterministic variables: const
## Sample size: 118
## Log Likelihood: -481.184
## Roots of the characteristic polynomial:
## 0.7342 0.5313
## Call:
## VAR(y = combo_GAL_GSJ, p = 1)
##
##
## Estimation results for equation MSA1o:
## ======================================
## MSA1o = MSA1o.l1 + MSA3o.l1 + const
##
##          Estimate Std. Error t value Pr(>|t|)
## MSA1o.l1  0.74241    0.06180  12.013  < 2e-16 ***
## MSA3o.l1 -0.08064    0.05473  -1.473  0.14336
## const     0.61718    0.20231   3.051  0.00283 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 1.571 on 115 degrees of freedom
## Multiple R-Squared: 0.561,   Adjusted R-squared: 0.5533
## F-statistic: 73.47 on 2 and 115 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation MSA3o:
## ======================================
## MSA3o = MSA1o.l1 + MSA3o.l1 + const
##
##          Estimate Std. Error t value Pr(>|t|)
## MSA1o.l1  0.02145    0.08889   0.241   0.8097
## MSA3o.l1  0.52315    0.07872   6.646 1.06e-09 ***
## const     0.57762    0.29098   1.985   0.0495 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
##
## Residual standard error: 2.259 on 115 degrees of freedom
## Multiple R-Squared: 0.2776,  Adjusted R-squared: 0.265
## F-statistic:  22.1 on 2 and 115 DF,  p-value: 7.583e-09
##
##
##
## Covariance matrix of residuals:
##        MSA1o  MSA3o
## MSA1o 2.4675 0.1574
## MSA3o 0.1574 5.1047
##
## Correlation matrix of residuals:
##         MSA1o   MSA3o
## MSA1o 1.00000 0.04434
## MSA3o 0.04434 1.00000
```

When looking at the lagged coefficients predicting GAL(Albany), the only coefficient that is statistically significant is a lagged coefficient from GAL. It would appear that GSJ does not granger cause GAL, but further investigation is necessary. The same is true for the predictors predicting GSJ; the GSJ lagged coefficient is statistically signficant but not for GAL.

```r
grangertest(MSA1o~MSA3o, order = 3)
```

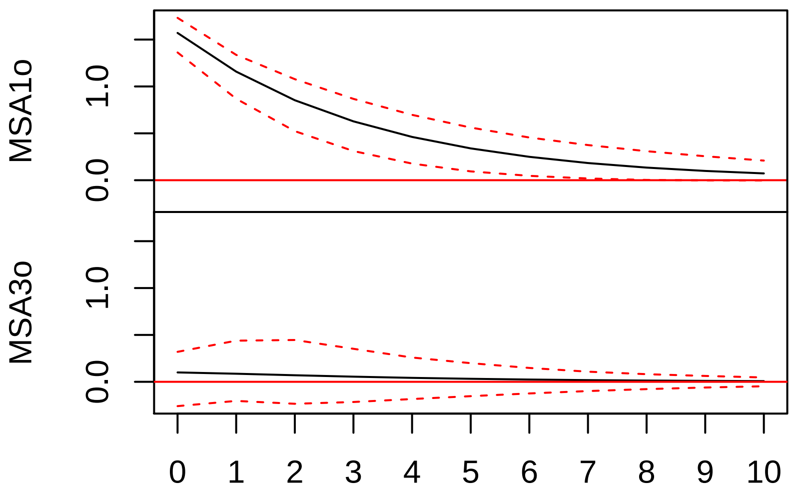
```
## Granger causality test
##
## Model 1: MSA1o ~ Lags(MSA1o, 1:3) + Lags(MSA3o, 1:3)
## Model 2: MSA1o ~ Lags(MSA1o, 1:3)
##   Res.Df Df      F Pr(>F)
## 1    109
## 2    112 -3 2.2372  0.088 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
grangertest(MSA3o~MSA1o, order = 3)
```

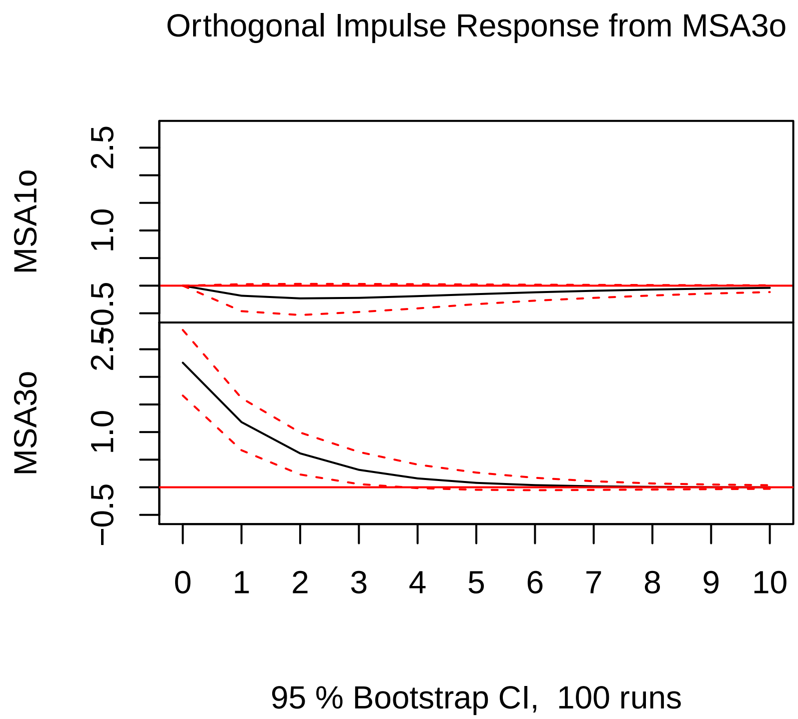
```
## Granger causality test
##
## Model 1: MSA3o ~ Lags(MSA3o, 1:3) + Lags(MSA1o, 1:3)
## Model 2: MSA3o ~ Lags(MSA3o, 1:3)
##   Res.Df Df      F Pr(>F)
## 1    109
## 2    112 -3 0.1307 0.9416
```

```r
plot(irf(VAR_model_GAL_GSJ))
```

Orthogonal Impulse Response from MSA1o



95 % Bootstrap CI, 100 runs



When performing the granger casuality test, our predictions from before were correct. Neither of the lagged time series has lagged serial correlation with each other from looking at the P-Values of both tests.

In conclusion, we should not use the VAR model to predict either time series because neither granger cause each other, meaning there is no predictive power using this model.