Macroeconomics Project

Samuel and Matthias

07/06/2021

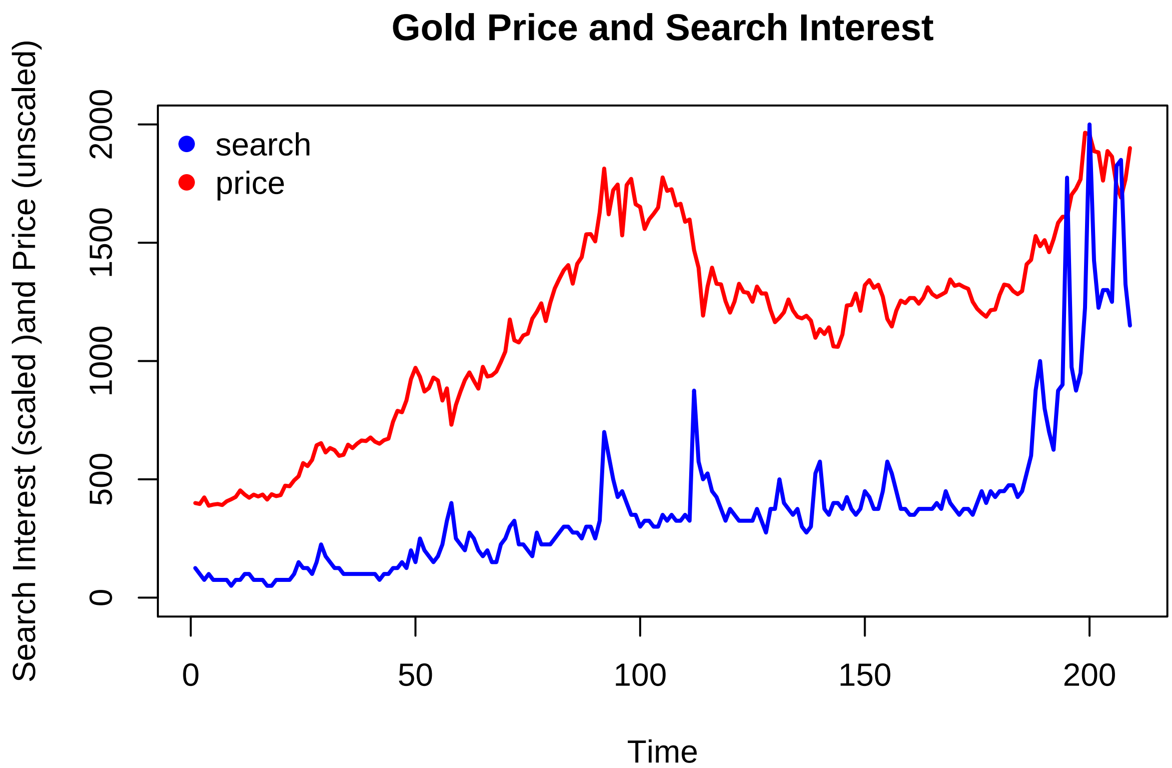
Idea

We want to look at the relationship between certain prices and the respective search interest on google for these prices. Can we find granger causality for this relationship? What are possible issues? For example: modern trading algorithms scrape data from the internet and then buy or sell based on the sentiment. Large spikes in search interest may trigger such algorithms. As media spreads the news of price increases more people will look up prices of goods and commodities, again triggering the algorithms. This is basically a feedback loop.

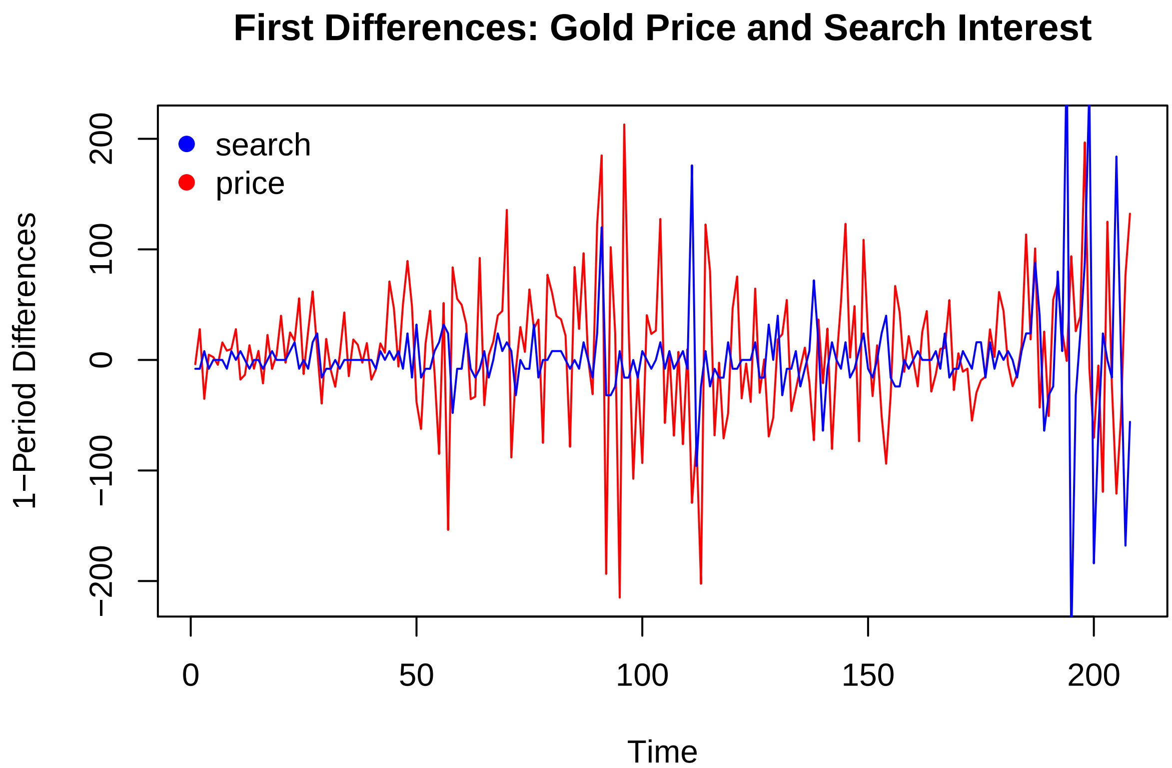
Project Code

```r
# clear workspace
rm(list=ls())

# load needed libraries
library(readr)
library(vars)

# set working directory
setwd("/Users/samue/Downloads/Studium/Economics (Master - Vienna)/2. Semester/Macroeconometrics/Project/

# import search trends
data <- read.csv("btc-vs-gold-2004.csv")
# import prices data:
gold_pr <- read.csv("gold-2004.csv")
```

```r
# plot gold price
plot(gold_pr$GOLDPMGBD228NLBM,type = 'l', lwd = 2, col = 'red',
     ylim = c(0,2000), main = 'Gold Price and Search Interest',
     xlab = 'Time', ylab = 'Search Interest (scaled )and Price (unscaled)')
# add gold search interest scaled up
lines(25*data$GOLD, lwd = 2, col = 'blue')
legend('topleft', legend = c('search','price'),
       col = c('blue','red'), bty = "n", pch = c(19,19))
```

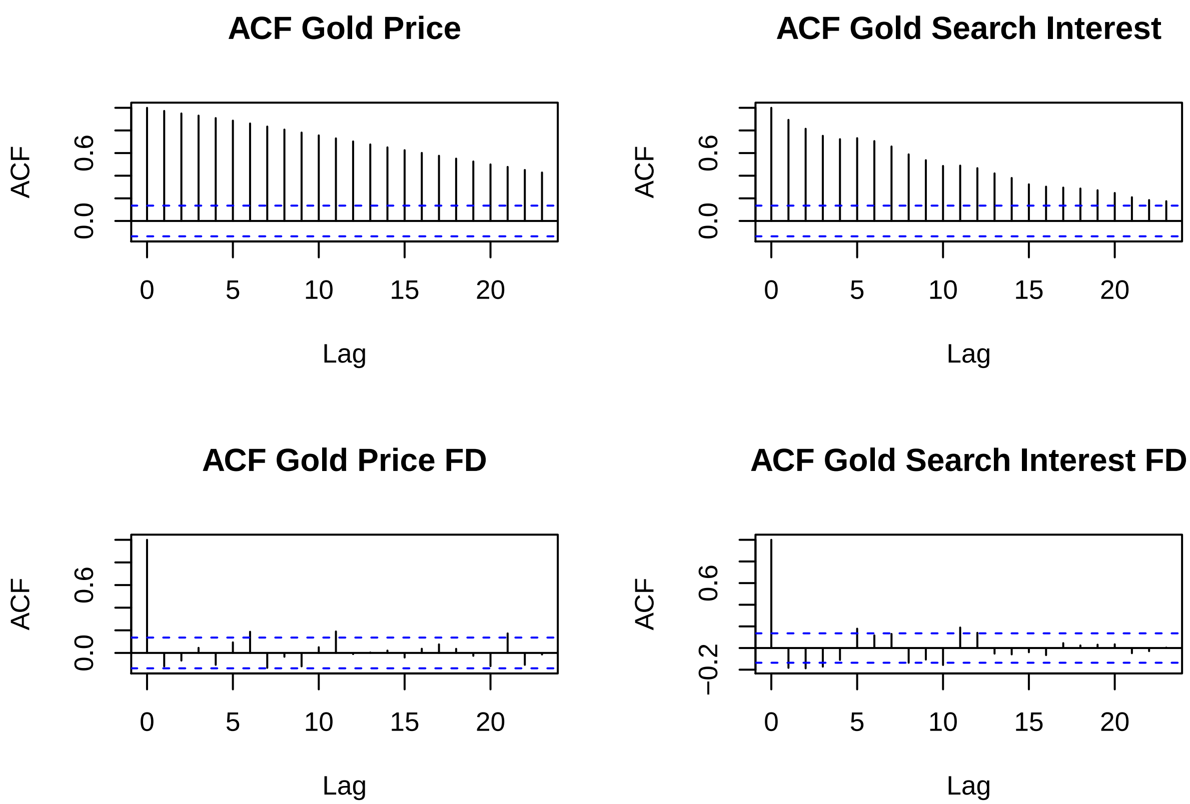



```
# create first differenced prices and search interest
t <- length(gold_pr$DATE)
gold_price_FD <- rep(0,t-1)
for(i in 2:209){gold_price_FD[i-1] <- gold_pr$GOLDPMGBD228NLBM[i]-gold_pr$GOLDPMGBD228NLBM[i-1]}
gold_search_FD <- rep(0,t-1)
for(i in 2:209){gold_search_FD[i-1] <- data$GOLD[i]-data$GOLD[i-1]}

# plot first differenced variables
plot(gold_price_FD, type = 'l', lwd = 1, col = 'red',
     xlab = 'Time', ylab = '1-Period Differences',
     main = 'First Differences: Gold Price and Search Interest')
lines(gold_search_FD*8, lwd = 1, col = 'blue')
legend('topleft', legend = c('search','price'),
       col = c('blue','red'), bty = "n", pch = c(19,19))
```



It appears that the more volatile regions match. Issue seems to be the scaling of the variables.

```
# plot ACF for unmodified variables:
par(mfrow=c(2,2))      # changes the plot layout to more easily compare them
acf(gold_pr$GOLDPMGBD228NLBM, main = 'ACF Gold Price')
acf(data$GOLD, main = 'ACF Gold Search Interest')

# plot ACF for differenced variables
acf(gold_price_FD,main = 'ACF Gold Price FD')
acf(gold_search_FD, main = 'ACF Gold Search Interest FD')
```

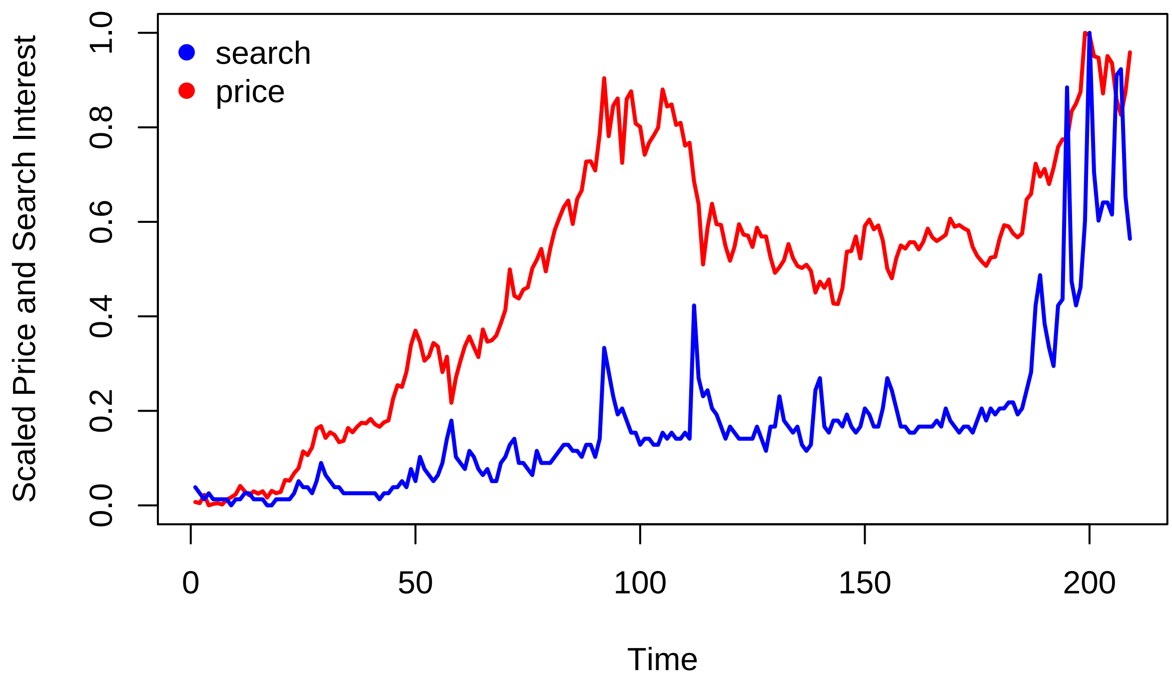



```
par(mfrow = c(1,1))    # revert layout changes
```

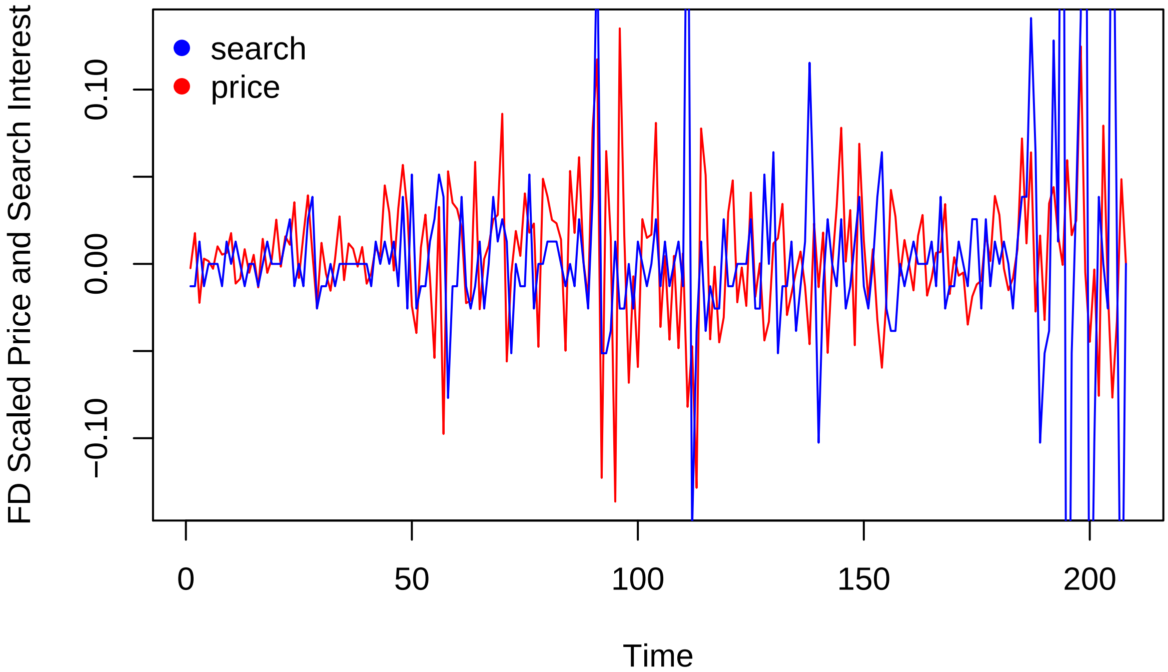
Autocorrelation for the differenced variables seems like no month-on-month relationship between the changes. Kind of like a random walk?

Might help with the interpretation: scale all variables $\mathbf{X}$ such that $X_t \in [0,1] \forall t \in T$.

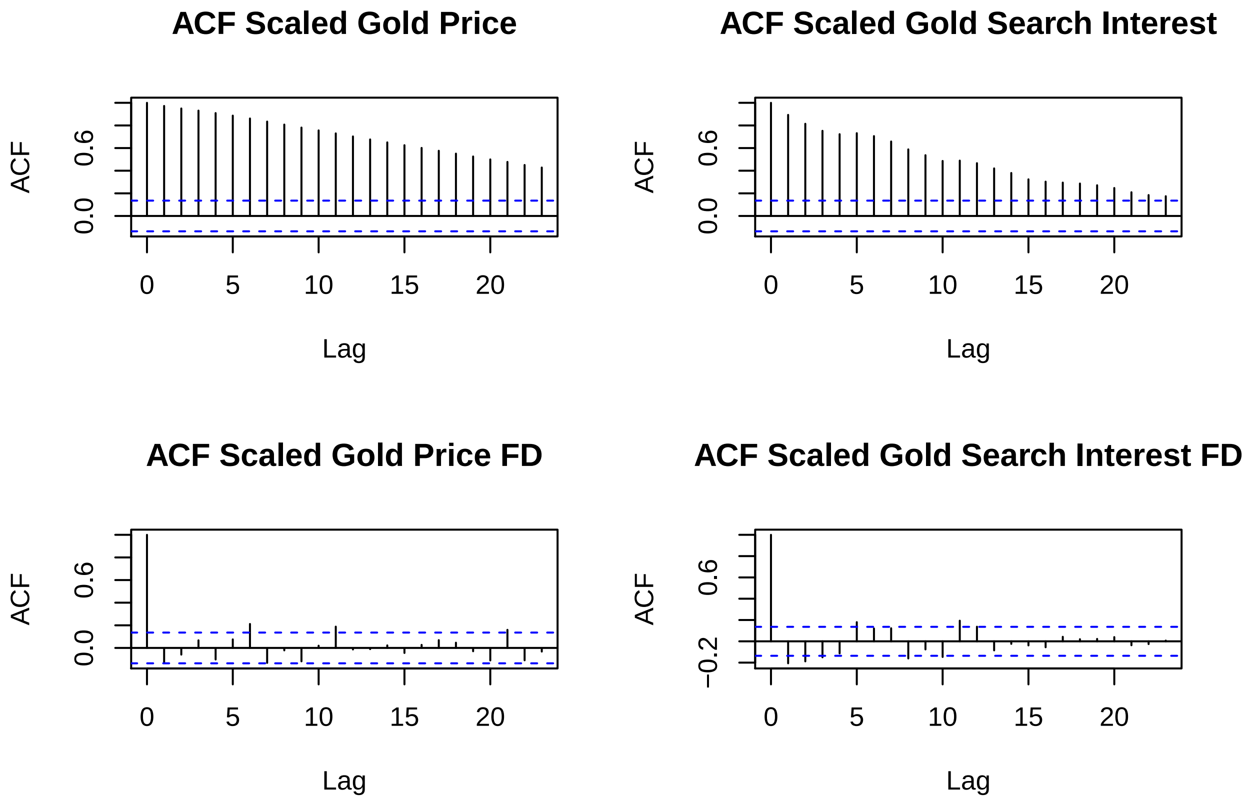
```
range01 <- function(x){(x-min(x))/(max(x)-min(x))}
plot(range01(gold_pr$GOLDPMGBD228NLBM), lwd = 2, type = 'l',
     ylab = 'Scaled Price and Search Interest',
     xlab = 'Time', col = 'red')
lines(range01(data$GOLD), lwd = 2, col = 'blue')
legend('topleft', legend = c('search','price'),
       col = c('blue','red'), bty = "n", pch = c(19,19))
```




```r
# save scaled variables
gold_price_scaled <- range01(gold_pr$GOLDPMGBD228NLBM)
gold_search_scaled <- range01(data$GOLD)

# create first difference on scaled variables:
gold_search_scaled_FD <- rep(0,t-1)
gold_price_scaled_FD <- rep(0,t-1)

for(i in 2:t-1){
  gold_price_scaled_FD[i-1] <- gold_price_scaled[i]-gold_price_scaled[i-1]
}
for(i in 2:t-1){
  gold_search_scaled_FD[i-1] <- gold_search_scaled[i]-gold_search_scaled[i-1]
}

# plot first differenced:
plot(gold_price_scaled_FD, lwd = 1, type = 'l',
     ylab = 'FD Scaled Price and Search Interest',
     xlab = 'Time', col = 'red')
lines(gold_search_scaled_FD, lwd = 1, col = 'blue')
legend('topleft', legend = c('search','price'),
       col = c('blue','red'), bty = "n", pch = c(19,19))
```




```
# plot ACFs
par(mfrow=c(2,2))      # changes the plot layout to more easily compare them
acf(gold_price_scaled, main = 'ACF Scaled Gold Price')
acf(gold_search_scaled, main = 'ACF Scaled Gold Search Interest')
acf(gold_price_scaled_FD,main = 'ACF Scaled Gold Price FD')
acf(gold_search_scaled_FD, main = 'ACF Scaled Gold Search Interest FD')
```




```r
par(mfrow = c(1,1))    # revert layout changes
```

Unsurprisingly the rescaling does not matter for the autocorrelation as it is a scaled measure of linear relationships anyway.

```r
#####################################################
####### From here on: data saved as time series #####
#####################################################

# save variable vectors as time series format:
gold_price_scaled <- ts(gold_price_scaled, frequency = 12,
                        start = c(2004, 1), end = c(2021, 5))
gold_search_scaled <- ts(gold_search_scaled, frequency = 12,
                         start = c(2004,1), end = c(2021,5))

# set up data for estimation using `VAR()`
VAR_data <- window(ts.union(gold_price_scaled, gold_search_scaled),
                   start = c(2004, 1), end = c(2021, 5))

# estimate model coefficients using `VAR()`
VAR_est <- VAR(y = VAR_data, p = 2)
summary(VAR_est)
```

```
##
## VAR Estimation Results:
## =========================
## Endogenous variables: gold_price_scaled, gold_search_scaled
```

```
## Deterministic variables: const
## Sample size: 207
## Log Likelihood: 636.839
## Roots of the characteristic polynomial:
## 0.9924 0.8559 0.1625 0.1625
## Call:
## VAR(y = VAR_data, p = 2)
##
##
## Estimation results for equation gold_price_scaled:
## ==================================================
## gold_price_scaled = gold_price_scaled.l1 + gold_search_scaled.l1 + gold_price_scaled.l2 + gold_searc
##
##                        Estimate Std. Error t value Pr(>|t|)
## gold_price_scaled.l1   0.869163   0.070442  12.339   <2e-16 ***
## gold_search_scaled.l1 -0.020645   0.039085  -0.528   0.5979
## gold_price_scaled.l2   0.110635   0.070855   1.561   0.1200
## gold_search_scaled.l2  0.042045   0.039316   1.069   0.2862
## const                  0.011315   0.005945   1.903   0.0584 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.0395 on 202 degrees of freedom
## Multiple R-Squared: 0.9771,  Adjusted R-squared: 0.9766
## F-statistic:  2151 on 4 and 202 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation gold_search_scaled:
## ===================================================
## gold_search_scaled = gold_price_scaled.l1 + gold_search_scaled.l1 + gold_price_scaled.l2 + gold_sear
##
##                       Estimate Std. Error t value Pr(>|t|)
## gold_price_scaled.l1   0.34683    0.12492   2.776  0.00601 **
## gold_search_scaled.l1  0.74430    0.06931  10.738  < 2e-16 ***
## gold_price_scaled.l2  -0.26598    0.12565  -2.117  0.03551 *
## gold_search_scaled.l2  0.10172    0.06972   1.459  0.14615
## const                 -0.01107    0.01054  -1.050  0.29498
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.07006 on 202 degrees of freedom
## Multiple R-Squared: 0.8372,  Adjusted R-squared: 0.834
## F-statistic: 259.8 on 4 and 202 DF,  p-value: < 2.2e-16
##
##
##
## Covariance matrix of residuals:
##                    gold_price_scaled gold_search_scaled
## gold_price_scaled          1.561e-03         -4.454e-05
## gold_search_scaled        -4.454e-05          4.908e-03
##
## Correlation matrix of residuals:
```

```
##                    gold_price_scaled gold_search_scaled
## gold_price_scaled            1.0000            -0.0161
## gold_search_scaled          -0.0161             1.0000
```

```r
# augmented df test on only the gold price
df_test_gold <- urca::ur.df(gold_price_scaled, type = c('drift'),
                      selectlags = 'BIC')
summary(df_test_gold)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.137377 -0.019230 -0.000925  0.022274  0.126963
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.010394   0.005824   1.785   0.0758 .
## z.lag.1     -0.010964   0.010659  -1.029   0.3049
## z.diff.lag  -0.116850   0.070135  -1.666   0.0972 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03946 on 204 degrees of freedom
## Multiple R-squared:  0.01948,    Adjusted R-squared:  0.009863
## F-statistic: 2.026 on 2 and 204 DF,  p-value: 0.1345
##
##
## Value of test-statistic is: -1.0286 2.2503
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.46 -2.88 -2.57
## phi1  6.52  4.63  3.81
```

Null cannot be rejected given the data, the null is non-stationarity. Not very unexpected as prices are often thought about as following a random walk and thus being non-stationary. But we can also look at difference-stationarity.

```r
# augmented df test on only the differenced gold price
df_test_gold_FD <- urca::ur.df(gold_price_scaled_FD, type = 'none',
                              selectlags = 'BIC')
summary(df_test_gold_FD)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
```

```
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.139705 -0.013946  0.004984  0.026891  0.129077
##
## Coefficients:
##            Estimate Std. Error t value Pr(>|t|)
## z.lag.1    -1.19119    0.10475 -11.372   <2e-16 ***
## z.diff.lag  0.06367    0.07010   0.908    0.365
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0393 on 204 degrees of freedom
## Multiple R-squared:  0.562,  Adjusted R-squared:  0.5577
## F-statistic: 130.9 on 2 and 204 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -11.3722
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau1 -2.58 -1.95 -1.62
```

As the test rejects, given the data we cannot say that the data is not stationary.

```r
# VAR model with unscaled prices
# save variable vectors as time series format:
gold_price <- ts(gold_pr$GOLDPMGBD228NLBM, frequency = 12,
                        start = c(2004, 1), end = c(2021, 5))
gold_search <- ts(data$GOLD, frequency = 12,
                         start = c(2004,1), end = c(2021,5))

# set up data for estimation using `VAR()`
VAR_data <- window(ts.union(gold_price, gold_search),
                   start = c(2004, 1), end = c(2021, 5))

# estimate model coefficients using `VAR()`
VAR_est <- VAR(y = VAR_data, p = 1, type = 'both')
summary(VAR_est)
```

```
##
## VAR Estimation Results:
## =========================
## Endogenous variables: gold_price, gold_search
## Deterministic variables: both
## Sample size: 208
## Log Likelihood: -1799.017
## Roots of the characteristic polynomial:
```

```
## 0.9694 0.7759
## Call:
## VAR(y = VAR_data, p = 1, type = "both")
##
##
## Estimation results for equation gold_price:
## ===========================================
## gold_price = gold_price.l1 + gold_search.l1 + const + trend
##
##                  Estimate Std. Error t value Pr(>|t|)
## gold_price.l1     0.96788    0.01798  53.837   <2e-16 ***
## gold_search.l1    0.10532    0.50011   0.211   0.8334
## const            25.95499   13.78646   1.883   0.0612 .
## trend             0.15617    0.13017   1.200   0.2317
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 62.32 on 204 degrees of freedom
## Multiple R-Squared: 0.9772,  Adjusted R-squared: 0.9769
## F-statistic:  2913 on 3 and 204 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation gold_search:
## ============================================
## gold_search = gold_price.l1 + gold_search.l1 + const + trend
##
##                   Estimate Std. Error t value Pr(>|t|)
## gold_price.l1     0.002848   0.001578   1.805   0.0725 .
## gold_search.l1    0.777456   0.043888  17.715   <2e-16 ***
## const            -2.158855   1.209834  -1.784   0.0758 .
## trend             0.023485   0.011423   2.056   0.0411 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 5.469 on 204 degrees of freedom
## Multiple R-Squared: 0.8359,  Adjusted R-squared: 0.8335
## F-statistic: 346.5 on 3 and 204 DF,  p-value: < 2.2e-16
##
##
##
## Covariance matrix of residuals:
##             gold_price gold_search
## gold_price      3883.7      -12.50
## gold_search      -12.5       29.91
##
## Correlation matrix of residuals:
##             gold_price gold_search
## gold_price     1.00000    -0.03669
## gold_search   -0.03669     1.00000
```

```r
# compare the VAR to the AR(1) model for the prices
T <-length(gold_price)
```

```r
gold_price_2 <- as.numeric(gold_price[-1])
gold_price_lagged <- as.numeric(gold_price[-T])

plot(gold_price_2, type = 'l', lwd = 1, col = 'red',
     main = 'Gold Price and Lagged Gold Price',
     ylab = 'Gold Price', xlab = 'Months from 01.2004')
lines(gold_price_lagged, lwd = 1, col = 'blue')
legend('topleft', legend = c('Lagged Price','Price'),
       col = c('blue','red'), bty = "n", pch = c(19,19))
```


```r
# estimate model
gold_price_AR1 <- lm(gold_price_2 ~ gold_price_lagged)
# estimate robust standard errors
coeftest(gold_price_AR1, vcov. = vcovHC, type = "HC1")
```

```
##
## t test of coefficients:
##
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        20.49216   10.59726  1.9337  0.05452 .
## gold_price_lagged   0.98841    0.01114 88.7232  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The values on the intercept seem to differ, but the estimated coefficient on the lag seems to fit.

```r
# verify the 'by-hand' results with built-in function
ar.ols(gold_price, order.max = 1, intercept = T)
```

```
##
## Call:
## ar.ols(x = gold_price, order.max = 1, intercept = T)
##
## Coefficients:
##      1
## 0.9884
##
## Intercept: 7.171 (4.301)
##
## Order selected 1  sigma^2 estimated as  3848
```

```r
forecast::auto.arima(gold_price, ic = 'aic')
```

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

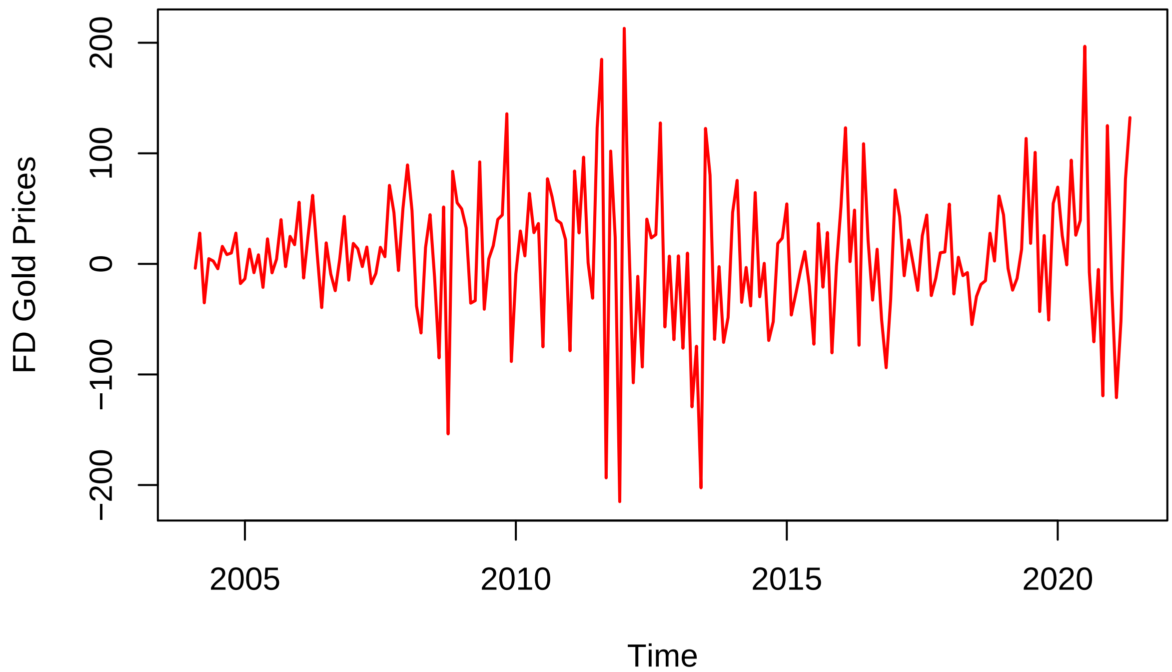
```
## Series: gold_price
## ARIMA(0,1,1) with drift
##
## Coefficients:
##           ma1   drift
##       -0.1411  7.1279
## s.e.   0.0740  3.6766
##
## sigma^2 estimated as 3842:  log likelihood=-1152.52
## AIC=2311.05   AICc=2311.16   BIC=2321.06
```

The last model is automated to difference such that the data is stationary, then the function finds the best forecasting model via the AIC. Here this would be an ARMA(0,1) model:

$$\Delta\widehat{\text{gold price}}_t = \underset{3.6766}{(7.1279)} + \epsilon_t + \underset{(0.0740)}{(-0.1411)}\epsilon_{t-1}$$

Differenced AR(1) and ARCH Model for Gold Prices

```r
# Differenced AR(1) and ARCH Model for Gold Prices
gold_price_FD <- ts(gold_price_FD, frequency = 12,
                    start = c(2004, 2), end = c(2021, 5))
plot(gold_price_FD, ylab = 'FD Gold Prices', col = 'red', lwd = 1.5)
```

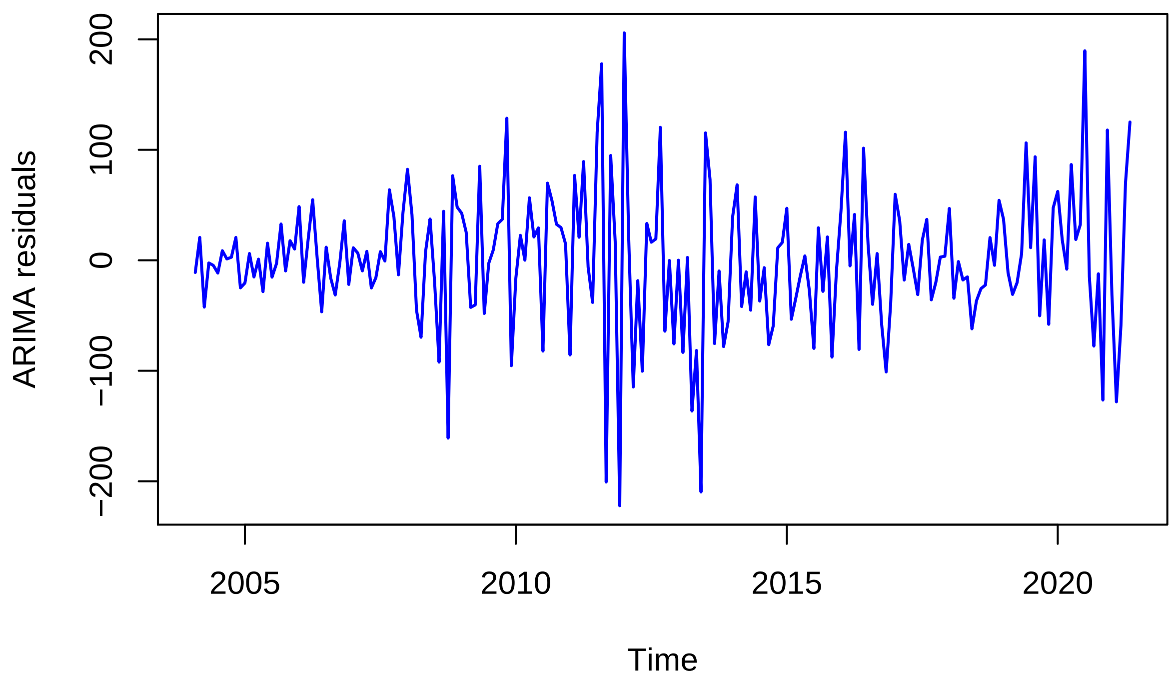



```r
ar1mod_FD <- arima(gold_price_FD, order = c(1,0,0))
ar1mod_FD
```

```
##
## Call:
## arima(x = gold_price_FD, order = c(1, 0, 0))
##
## Coefficients:
##           ar1  intercept
##       -0.1205     7.1540
## s.e.   0.0694     3.8237
##
## sigma^2 estimated as 3814:  log likelihood = -1152.78,  aic = 2311.57
```

```r
plot(forecast::arima.errors(ar1mod_FD),type = 'l', lwd = 1.5, col = 'blue',
     ylab = 'ARIMA residuals')
```

```
## Deprecated, use residuals.Arima(object, type='regression') instead
```




```r
mean(forecast::arima.errors(ar1mod_FD))
```

```
## Deprecated, use residuals.Arima(object, type='regression') instead
```

```
## [1] 0.05853392
```

Going by the plot, it does not appear that the variance of the residuals is constant over time but rather has times of higher and lower volatility.

```r
resi_ar1_FD_2 <- (forecast::arima.errors(ar1mod_FD))^2
```

```
## Deprecated, use residuals.Arima(object, type='regression') instead
```

```r
resi_arch1_FD_2_model <- arima(resi_ar1_FD_2, order = c(1,0,0))
resi_arch1_FD_2_model
```

```
##
## Call:
## arima(x = resi_ar1_FD_2, order = c(1, 0, 0))
##
## Coefficients:
##          ar1  intercept
##       0.3095  3887.0806
## s.e.  0.0662   727.6653
##
## sigma^2 estimated as 52737140:  log likelihood = -2144.4,  aic = 4294.79
```

```r
# plot the squared residuals:
plot(resi_ar1_FD_2, ylab = 'Squared AR(1) Residuals of FD Gold Price',
```

```
    xlab = 'Time from 02.2004', col = 'red', lwd = 1.5)
```