Financial Data Analysis

Contents

Data

Raw Data

```
#political shocks
#raw_truths <- read.csv(here("data/political_data", "trump_all_truths.csv"))
#raw_tweets <- read.csv(here("data/political_data", "tweets.csv"))


#market prices
raw_ONEQ <- read.csv(here("data/market_data", "ONEQ.csv")) #USA
raw_SMI <- read.csv(here("data/market_data", "SMI.csv")) #CH
raw_SPY <- read.csv(here("data/market_data", "SPY.csv")) #USA
raw_VTHR <- read.csv(here("data/market_data", "VTHR.csv")) #
raw_VTI <- read.csv(here("data/market_data", "VTI.csv")) #
raw_VGK <- read.csv(here("data/market_data", "VGK.csv")) #
raw_DAX <- read.csv(here("data/market_data", "DAX.csv")) #DE
raw_ASHR <- read.csv(here("data/market_data", "ASHR.csv")) #CHINA
raw_SPY2021_01 <- read.csv(here("data/market_data", "SPY-2021-01.csv"))
```

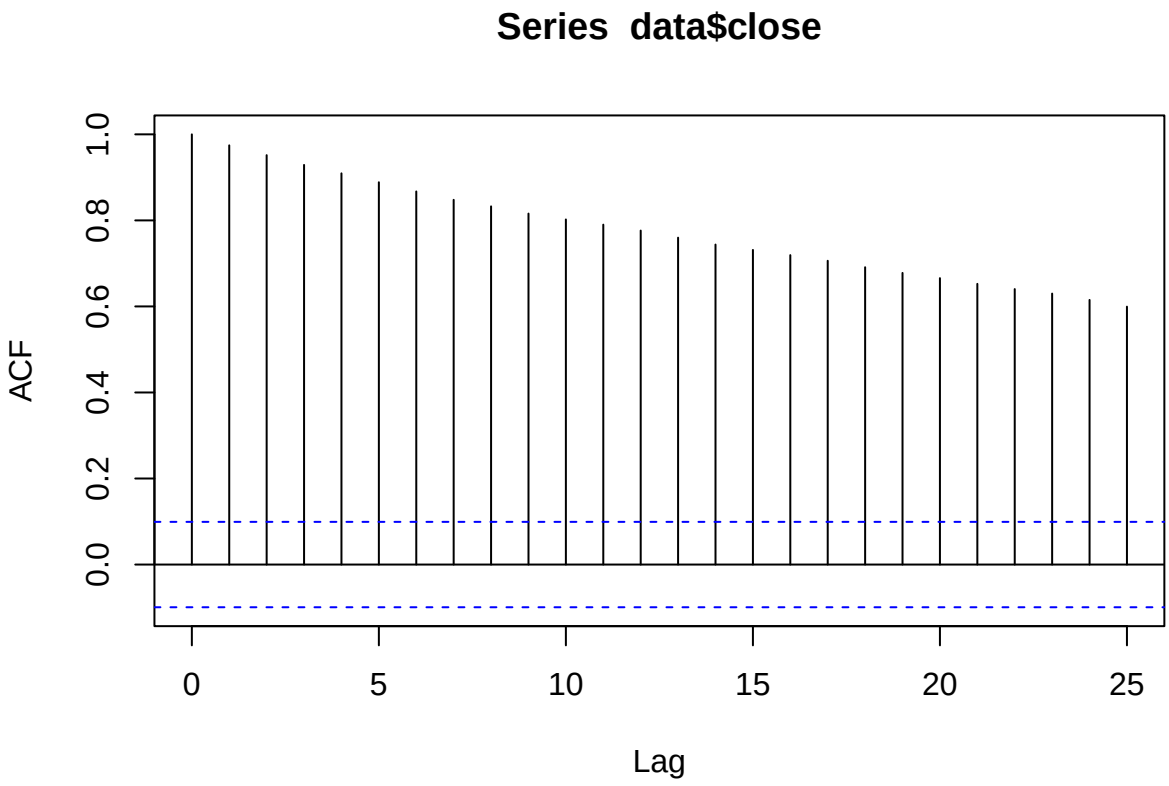
Quick Analysis

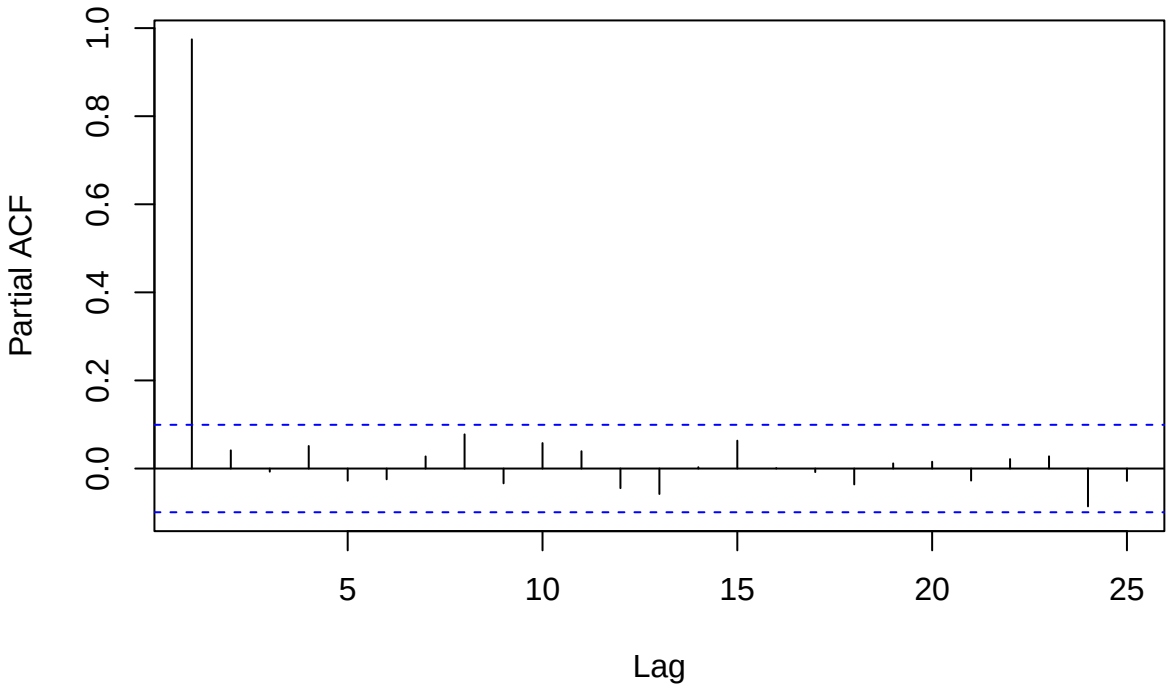
SPY April 2nd 2025

```
#extract a particular day
day_SPY_0402 = day_selector(raw_SPY,2025,04,02) #april 2nd 2025

#let's plot it
day_plotter(day_SPY_0402,"SPY Price on April 2nd 2025")
```


```
#quickly test some ARMA specifications
quick_arma(day_SPY_0402,1,0,0) #checking AR1,AR2,AR3
```


Series data$close
ACF
1.0
0.8
0.6
0.4
0.2
0.0
0
5
10
15
20
25
Lag




```
##                                AR Estimations
##
##                                  AR-1        AR-2         AR-3
##
##                  ar1            0.9975      0.9728       1.4609
##                                 (0.0030)    (0.0514)   (NaN)
##                  intercept    561.0971    561.3655      562.5635
##                                 (3.2897)    (3.4352)    (22.1897)
##                  ar2                         0.0249       0.0770
##                                              (0.0515)     (0.0013)
##                  ar3                                      -0.5386
##                                                           (0.0007)
##
##                  nobs         390         390           390
##                  sigma          0.2854      0.2853        0.3414
##                  logLik       -67.0847    -66.9808     -135.4359
##                  AIC          140.1693    141.9615      280.8718
##                  BIC          152.0678    157.8261      300.7025
##                  nobs.1       390.0000    390.0000      390.0000
##
##                  *** p < 0.001; ** p < 0.01; * p <
##                  0.05.
##
## Column names: names, AR-1, AR-2, AR-3
##                                Checking Residuals
##
```

```
##                         AR-1 Residuals   AR-2 Residuals   AR-3 Residuals
##
##        (Intercept)            0.0302 *         0.0291 *      -0.0051
##                              (0.0145)         (0.0145)       (0.0171)
##        REG1res_lagged        -0.0476
##                              (0.0510)
##        REG2res_lagged                         -0.0217
##                                               (0.0511)
##        REG3res_lagged                                        -0.1733 ***
##                                                              (0.0503)
##
##        N                    389              389            389
##        R2                     0.0022           0.0005         0.0297
##
##        *** p < 0.001; ** p < 0.01; * p < 0.05.
##
## Column names: names, AR-1 Residuals, AR-2 Residuals, AR-3 Residuals
```

```
#quick_arma(day_SPY_0402,2,0,0) #checking AR2,AR3,AR4
```

Realised Volatility

```
#for a particular day (outputs scalar)
r.vol(day_SPY_0402)
```

```
## [1] 0.08152862
```

```
#for a month (outputs vector of each day's realised volatility)
r.vol_month(raw_SPY2021_01)
```

```
##  [1] 0.039752157 0.017626575 0.041334441 0.010690380 0.023508351 0.016389520
##  [7] 0.016212187 0.008140512 0.007755798 0.019529301 0.008294740 0.007371578
## [13] 0.006302003 0.007263893 0.035831801 0.011293094 0.058478159 0.035438506
## [19] 0.075898392
```

```
#for each hour in a day (outputs a vector of each hour's realised volatility)
r.vol_hour(day_SPY_0402)
```

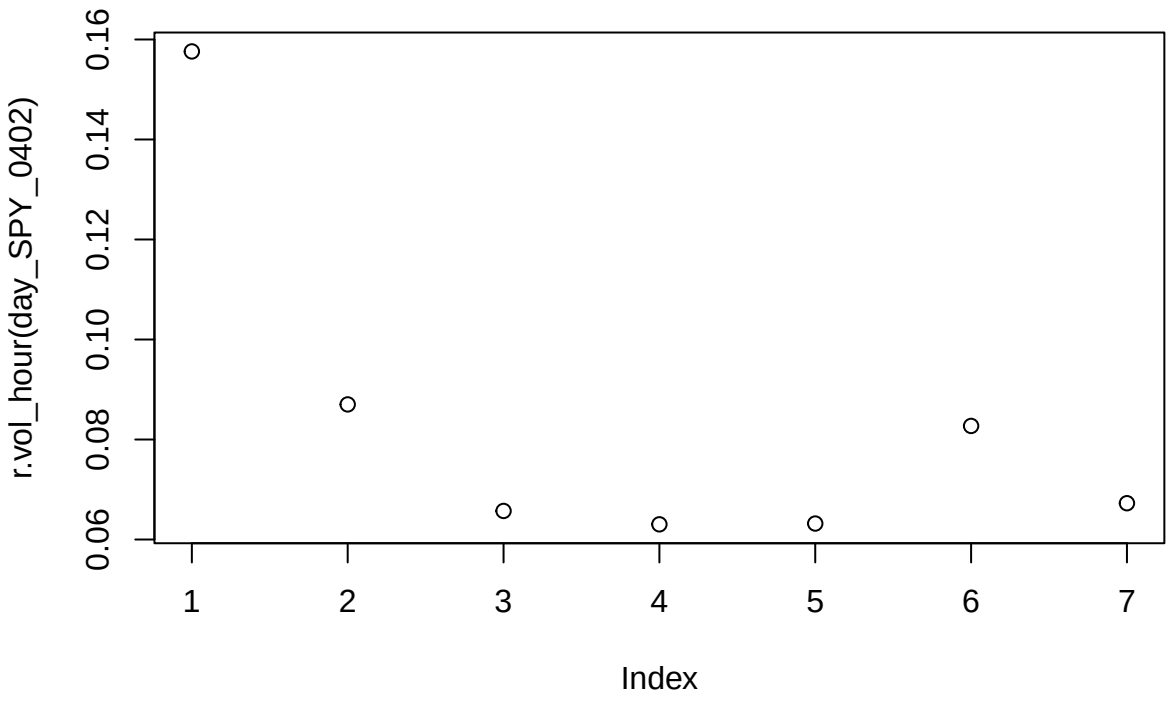
```
## [1] 0.15760939 0.08701794 0.06571201 0.06303564 0.06319524 0.08271313 0.06726031
```

```
#for each hour in a day for each day in a month (outputs a matrix)
month_hour = r.vol_month_hour(raw_SPY2021_01)
huxtable(data.frame(month_hour))
```

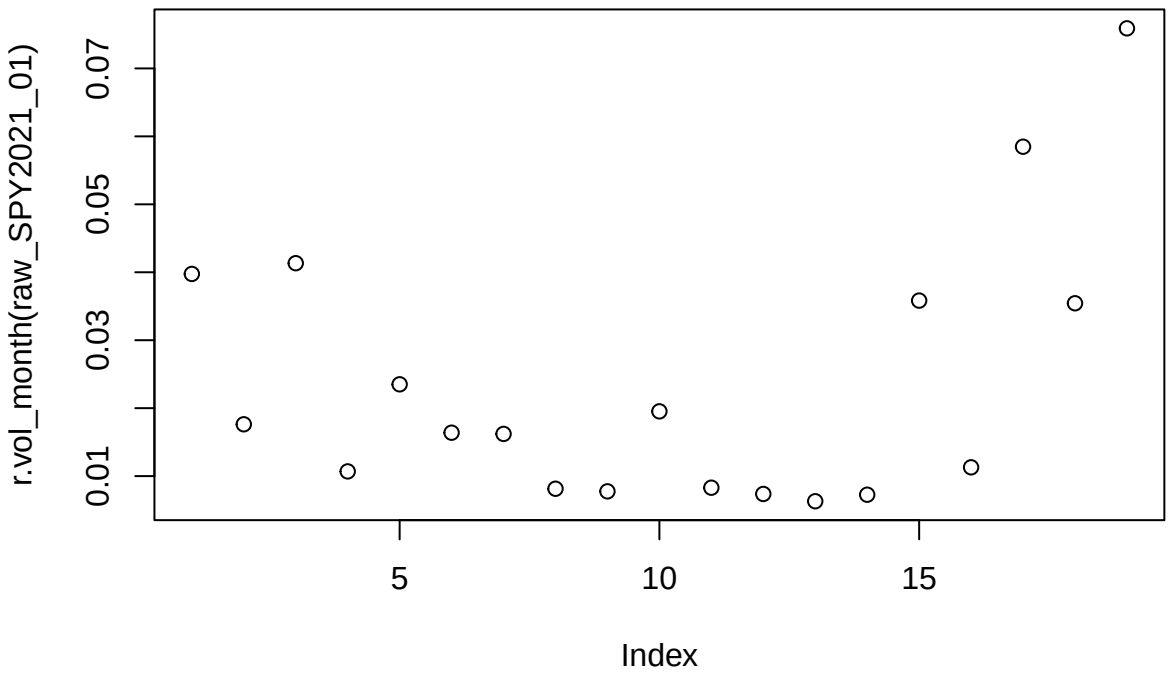
```
#plots

#hours in a day
plot(r.vol_hour(day_SPY_0402))
```

X4	X5	X6	X7	X8	X9	X10	X11	X12	X13	X14	X15	
345	0.0591	0.0648	0.034	0.0157	0.0101	0.0164	0.0235	0.0199	0.0251	0.00928	0.0213	0.0
136	0.0267	0.0139	0.0123	0.0142	0.00667	0.0523	0.0195	0.0115	0.00746	0.0142	0.0361	0.0
0931	0.0106	0.0112	0.0131	0.00537	0.00366	0.0173	0.00795	0.00704	0.00679	0.00456	0.103	0.0
0486	0.00934	0.00664	0.025	0.00549	0.00641	0.0106	0.00439	0.002	0.00517	0.00471	0.0432	0.0
0779	0.0399	0.00935	0.0128	0.00283	0.00514	0.00961	0.00293	0.00338	0.00205	0.00404	0.0147	0.0
0415	0.0228	0.0123	0.00903	0.00686	0.00342	0.00701	0.00294	0.00353	0.00173	0.00288	0.00939	0.0
132	0.0129	0.0215	0.0148	0.00999	0.02	0.0213	0.00488	0.00879	0.00552	0.0108	0.0136	0.0




```
#days in a month
plot(r.vol_month(raw_SPY2021_01))
```


r.vol_month(raw_SPY2021_01)
0.01
0.03
0.05
0.07
5
10
15
Index