Financial Data Analysis

Contents

Data

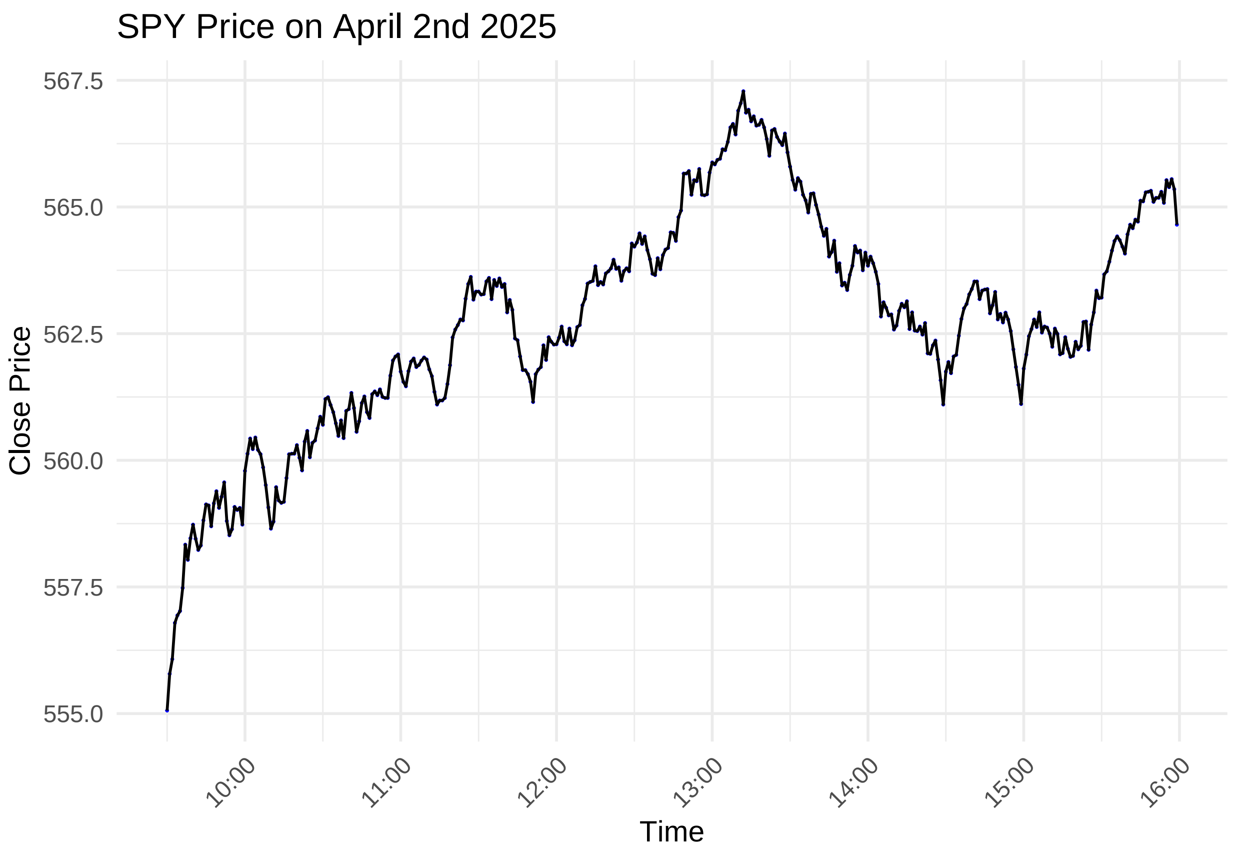
Raw Data

```r
#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 (loads and names them automatically)
#raw_ONEQ <- read.csv(here("data/market_data", "ONEQ.csv")) #USA
#raw_SMI <- read.csv(here("data/market_data", "SMI.csv")) #CH
#raw_VTHR <- read.csv(here("data/market_data", "VTHR.csv")) #USA
#raw_VTI <- read.csv(here("data/market_data", "VTI.csv")) #USA
#raw_DAX <- read.csv(here("data/market_data", "DAX.csv")) #DE
#raw_ASHR <- read.csv(here("data/market_data", "ASHR.csv")) #CHINA

#S&P500
data_loader(symbol="SPY")

#STOXX50
data_loader(symbol="VGK")




#CSI 300 (China)
data_loader(symbol="ASHR")
```

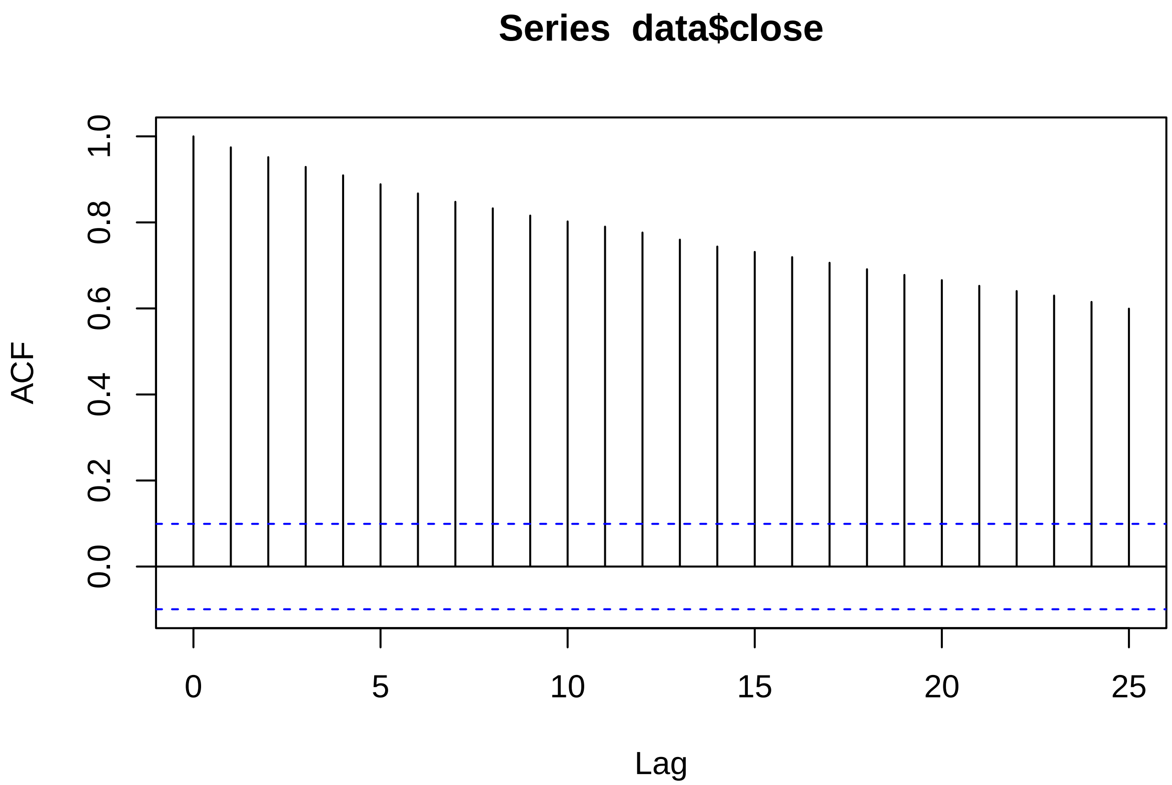
Quick Analysis

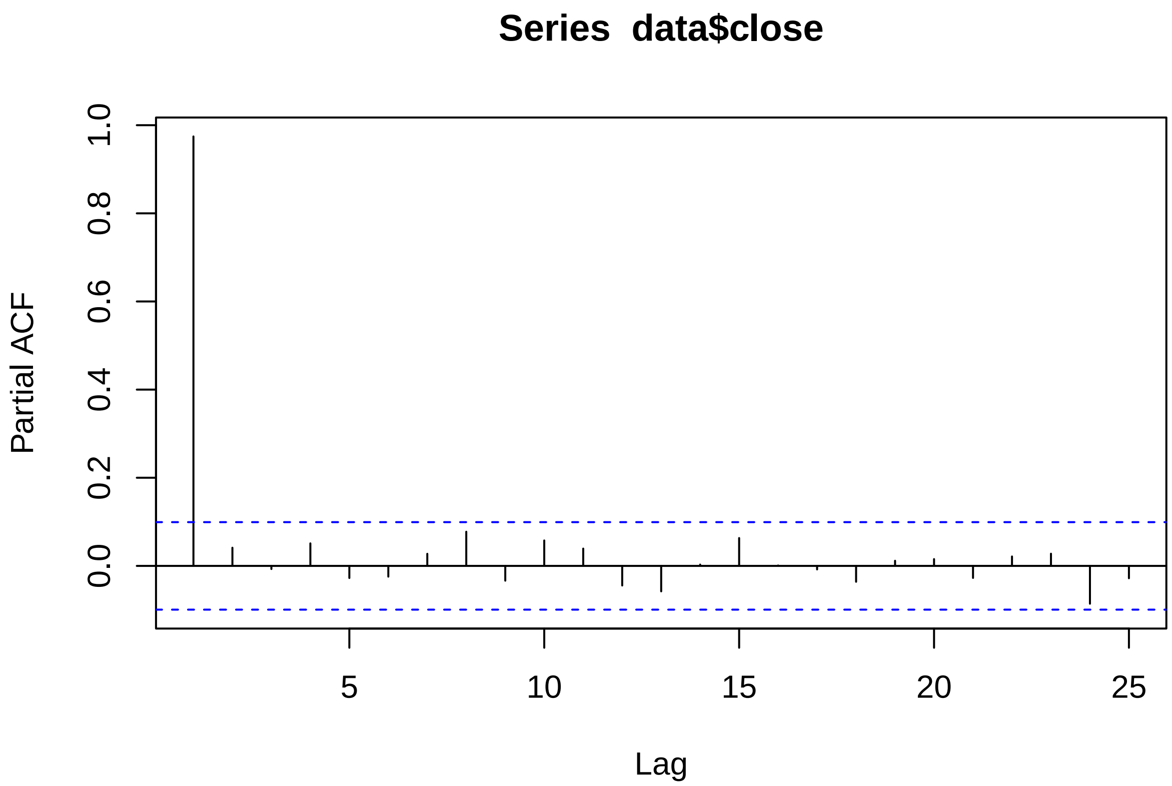
SPY April 2nd 2025

```r
#extract a particular day
SPY_25_04_02 = day_selector(raw_SPY,2025,04,02) #april 2nd 2025

#let's plot it
price_plotter_day(SPY_25_04_02,"SPY Price on April 2nd 2025")
```



```
#quickly test some ARMA specifications
quick_arma(SPY_25_04_02,1,0,0) #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(SPY_25_04_02,2,0,0) #checking AR2,AR3,AR4


#extract a particular month
SPY_24_09 = month_selector(raw_SPY,2024,09) #november 2024


#extract a particular year
SPY_24 = year_selector(raw_SPY,2024) #2024
```

Realised Volatility

Computations

```
#average per day (outputs scalar)
r.vol_day(SPY_25_04_02)
```

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

```
#average per day for each day in a month (outputs vector of each day's realised volatility)
r.vol_month(SPY_24_09)
```

```
##  [1] 0.03554182 0.06306683 0.04483728 0.07865960 0.02596162 0.03080083
##  [7] 0.06853948 0.04630338 0.02524256 0.02271454 0.03173591 0.14493815
## [13] 0.03160202 0.02320854 0.01822570 0.01616798 0.01071128 0.01843709
## [19] 0.01466890 0.02055323
```

```
#for each hour in a day (outputs a vector of each hour's realised volatility)
r.vol_day_hour(SPY_25_04_02)
```

```
## [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(SPY_24_09)
huxtable(head(data.frame(month_hour)))
```

X5	X6	X7	X8	X9	X10	X11	X12	X13	X14	X15	X16	X
0.0296	0.0304	0.121	0.0735	0.0232	0.0419	0.0384	0.0141	0.075	0.0243	0.0624	0.0155	0.020
0.0398	0.0607	0.106	0.0779	0.0539	0.0585	0.0284	0.026	0.0428	0.0253	0.0296	0.0349	0.011
0.0256	0.0486	0.0732	0.0547	0.0178	0.0179	0.0181	0.0168	0.0319	0.0315	0.013	0.0132	0.009
0.0124	0.0302	0.0683	0.0275	0.0133	0.0199	0.0471	0.00939	0.0124	0.0112	0.0225	0.00894	0.007
0.0219	0.0189	0.0408	0.0135	0.0093	0.00948	0.0376	0.0152	0.0117	0.013	0.0111	0.00717	0.014
0.0194	0.0147	0.0452	0.0745	0.0279	0.0104	0.035	0.333	0.0253	0.0237	0.00372	0.0118	0.007

```
#avg per day for each month of any dataset
#works for datasets with more than 1 year!
vol_SPY_daily = r.vol_daily(raw_SPY,merge=F)
head(vol_SPY_daily)
```

timestamp	r__vol__d
2019-01-02	0.0295
2019-01-03	0.0365
2019-01-04	0.0241
2019-01-07	0.0165
2019-01-08	0.0136
2019-01-09	0.0144

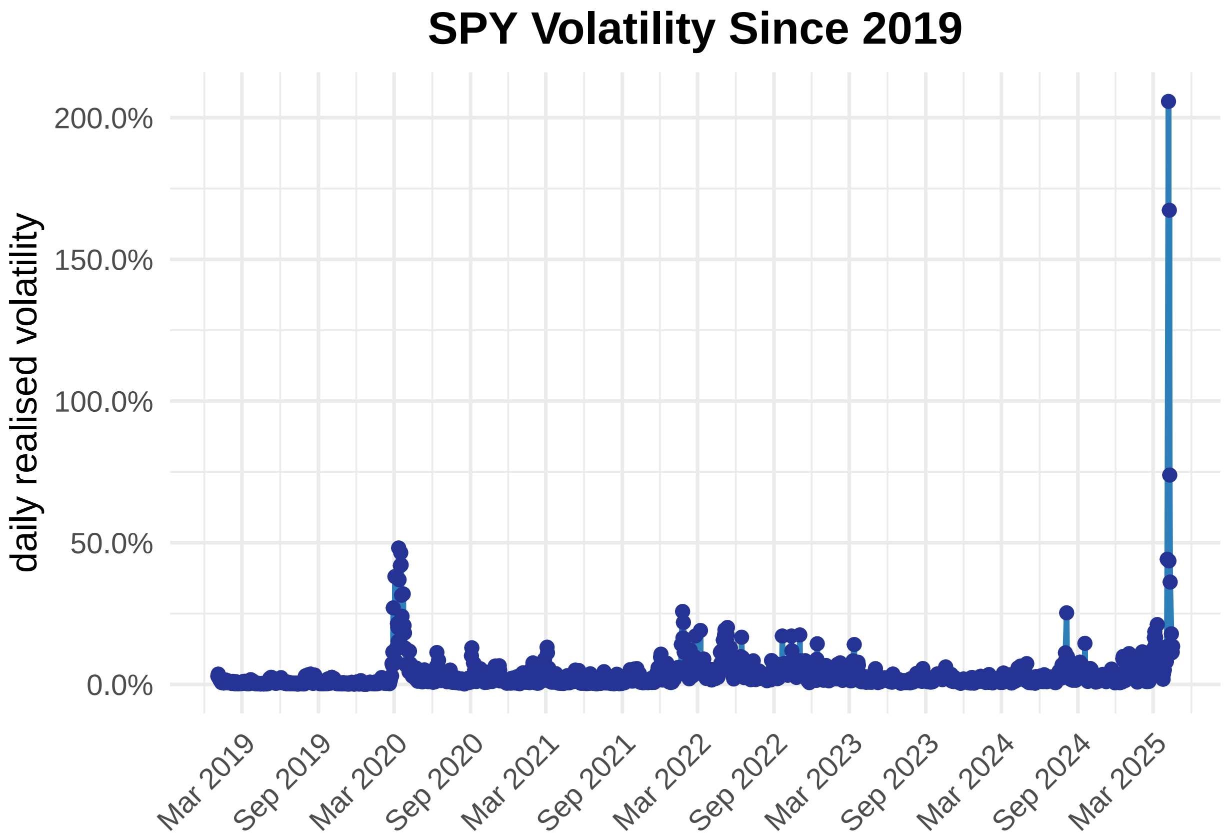
```
#can then filter out years, months, or days
vol_24d = year_selector(vol_SPY_daily,2024)
vol_24_08d = month_selector(vol_SPY_daily,2024,08)
vol_24_11_04d = day_selector(vol_SPY_daily,2024,11,04) #scalar


#avg per hour for each day of each month of any dataset
#works for datasets with more than 1 year!
vol_SPY_hourly = r.vol_hourly(raw_SPY,merge=F)
head(vol_SPY_hourly)
```

```
#can then filter out years, months, or days
vol_24h = year_selector(vol_SPY_hourly,2024)
vol_24_08h = month_selector(vol_SPY_hourly,2024,08)
vol_24_11_04h = day_selector(vol_SPY_hourly,2024,11,04) #vector
```

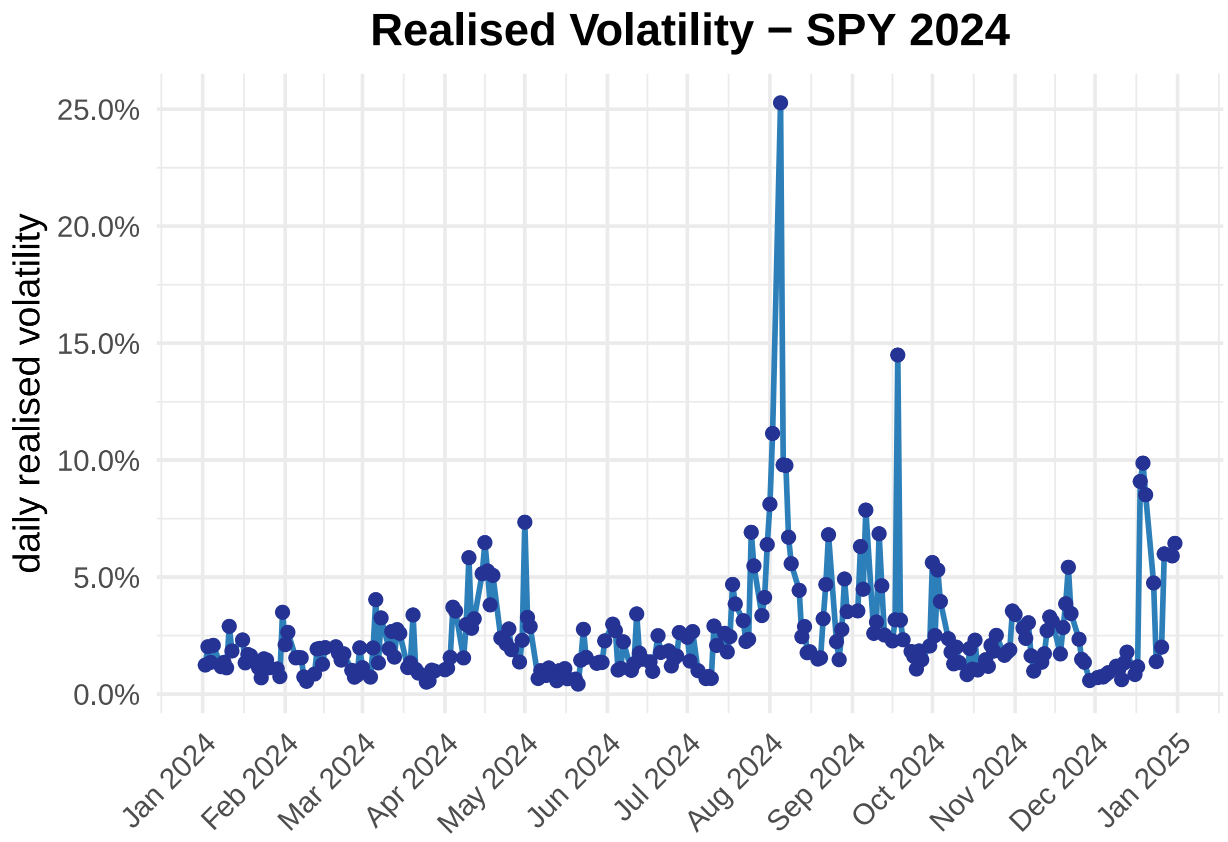
timestamp	r_vol_h
2019-01-02 09:00:00	0.034
2019-01-02 10:00:00	0.0401
2019-01-02 11:00:00	0.0363
2019-01-02 12:00:00	0.0185
2019-01-02 13:00:00	0.0185
2019-01-02 14:00:00	0.0199

Plots

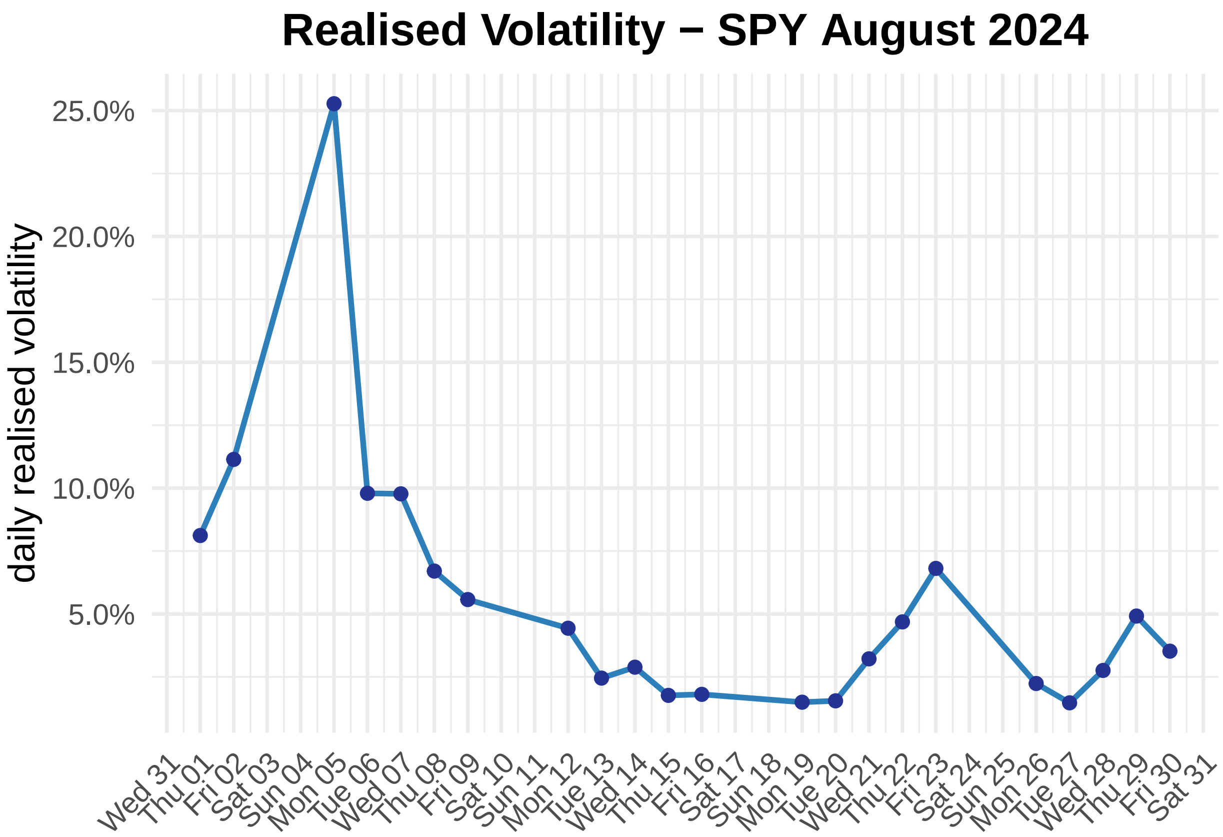
```
#avg per day volatility all time
dvol_plotter(vol_SPY_daily,breaks="yearly",
             title="SPY Volatility Since 2019")
```



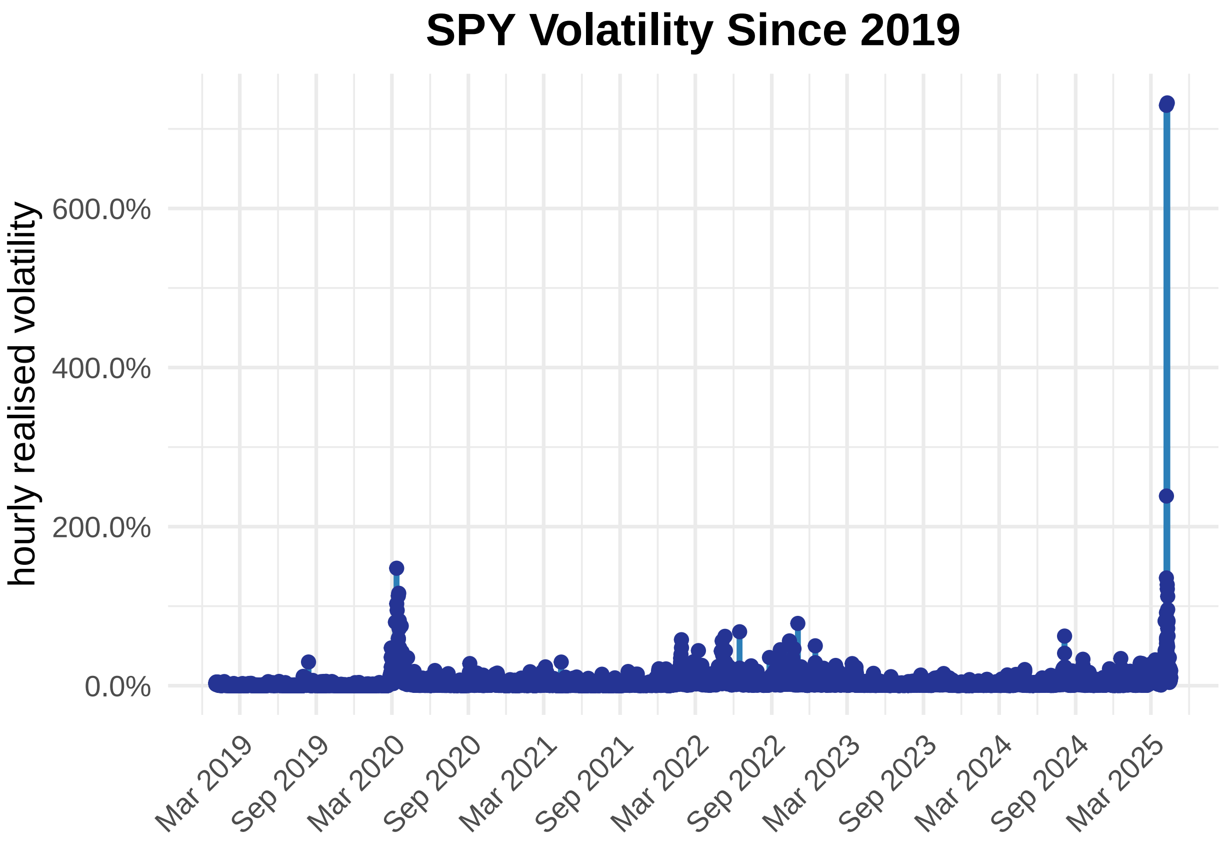
```
#avg per day volatility in a year
dvol_plotter(vol_24d,breaks="monthly",
             title="Realised Volatility - SPY 2024")
```



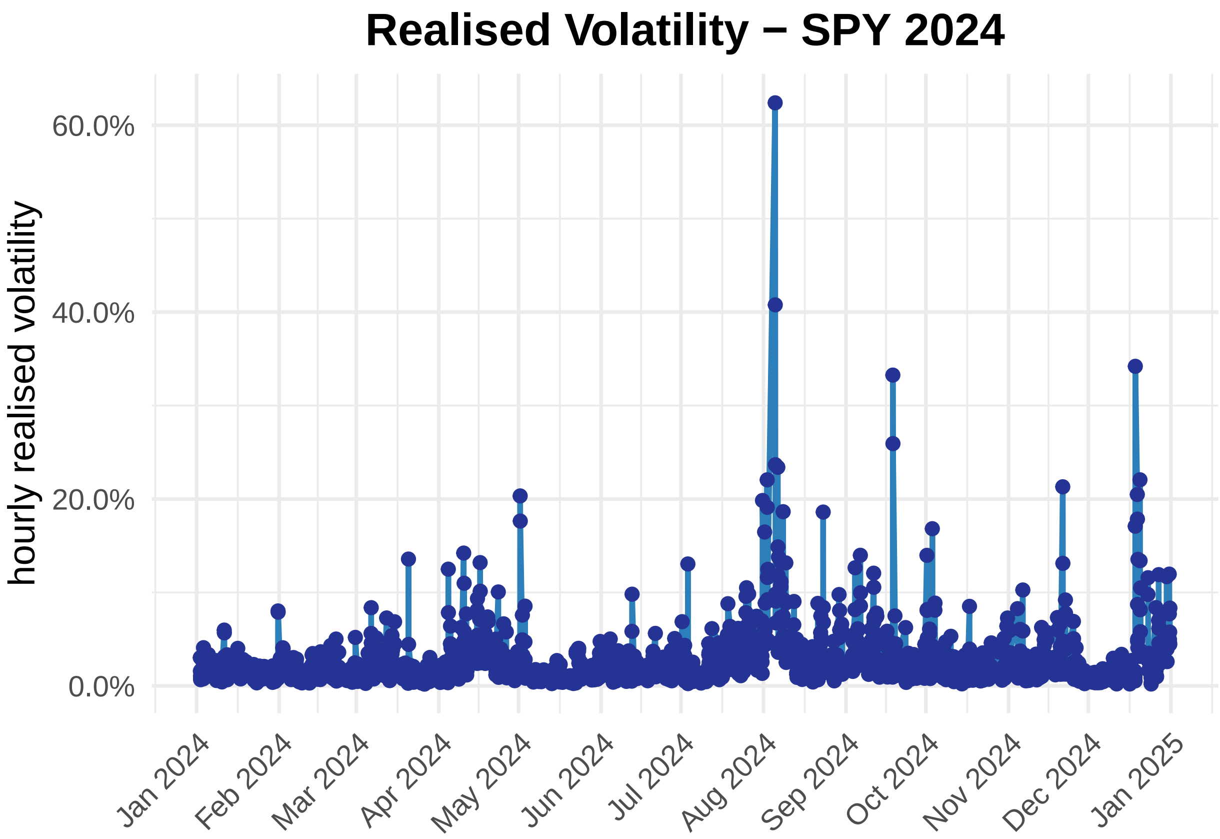
```
#avg per day volatility in a month
dvol_plotter(vol_24_08d,breaks="daily",
             title="Realised Volatility - SPY August 2024")
```



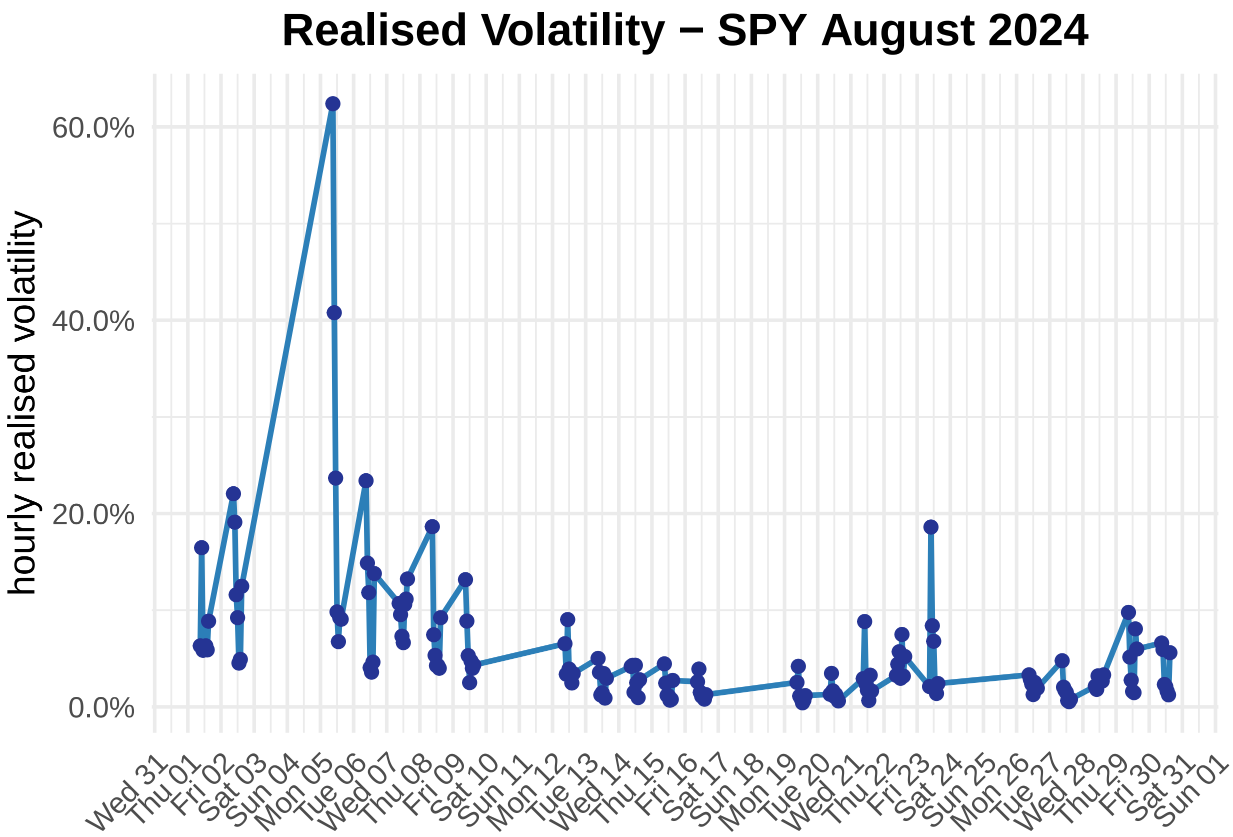
```r
#hourly volatility all time
hvol_plotter(vol_SPY_hourly,breaks="yearly",
             title="SPY Volatility Since 2019")
```



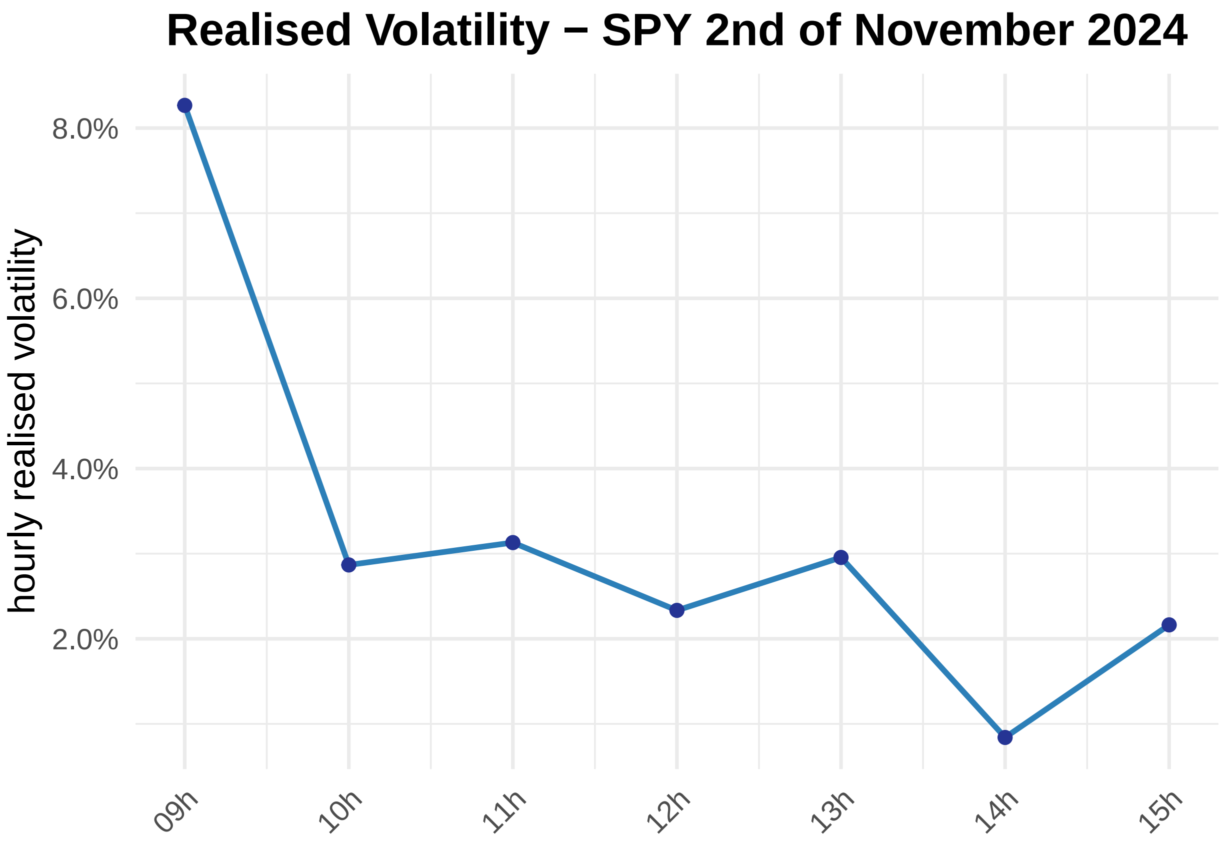
```
#hourly volatility in a year
hvol_plotter(vol_24h,breaks="monthly",
             title="Realised Volatility - SPY 2024")
```



```r
#hourly volatility in a month
hvol_plotter(vol_24_08h,breaks="daily",
             title="Realised Volatility - SPY August 2024")
```



```
#hourly volatility in a day
hvol_plotter(vol_24_11_04h,breaks="hourly",
             title="Realised Volatility - SPY 2nd of November 2024")
```



```
#vol_plotter(vol_VGK,breaks="yearly",title="VGK Volatility Since 2020")
```