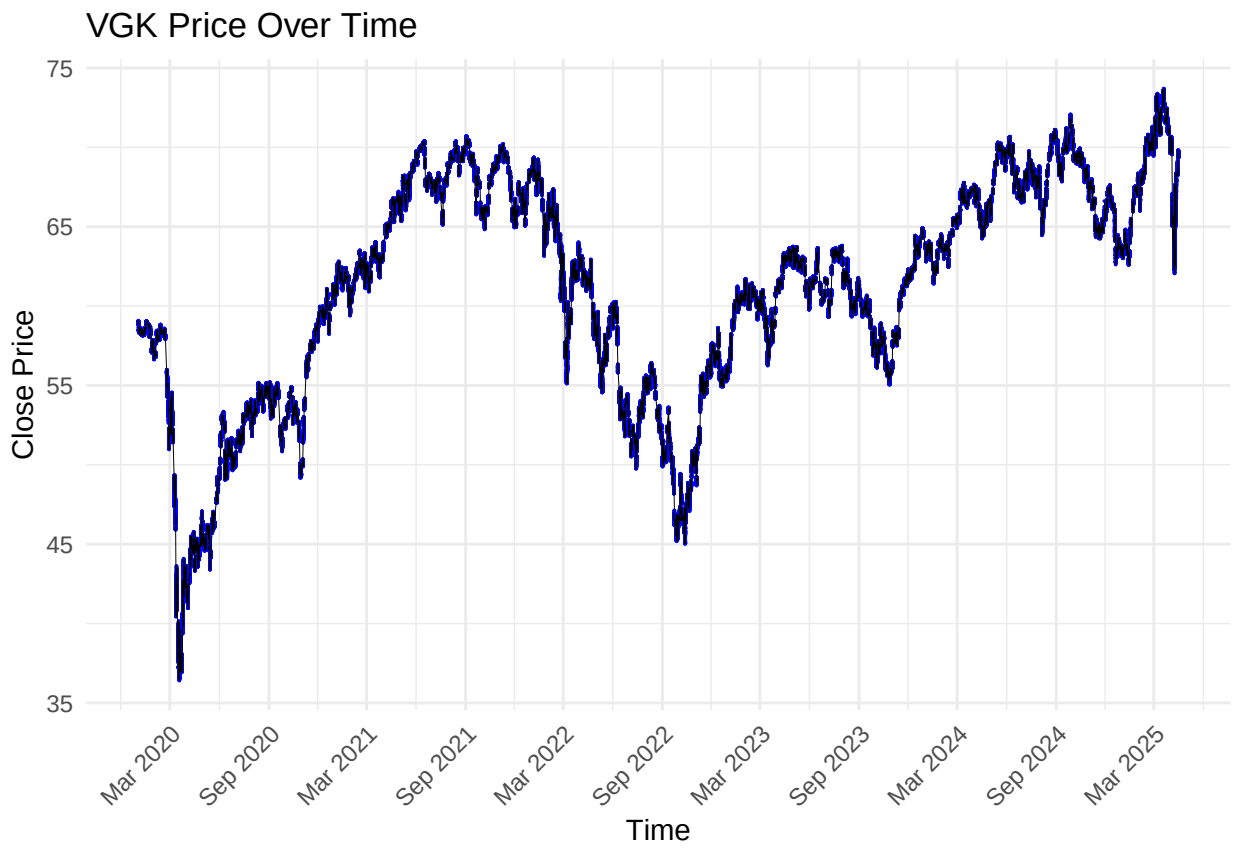
VGK Data Analysis

Contents

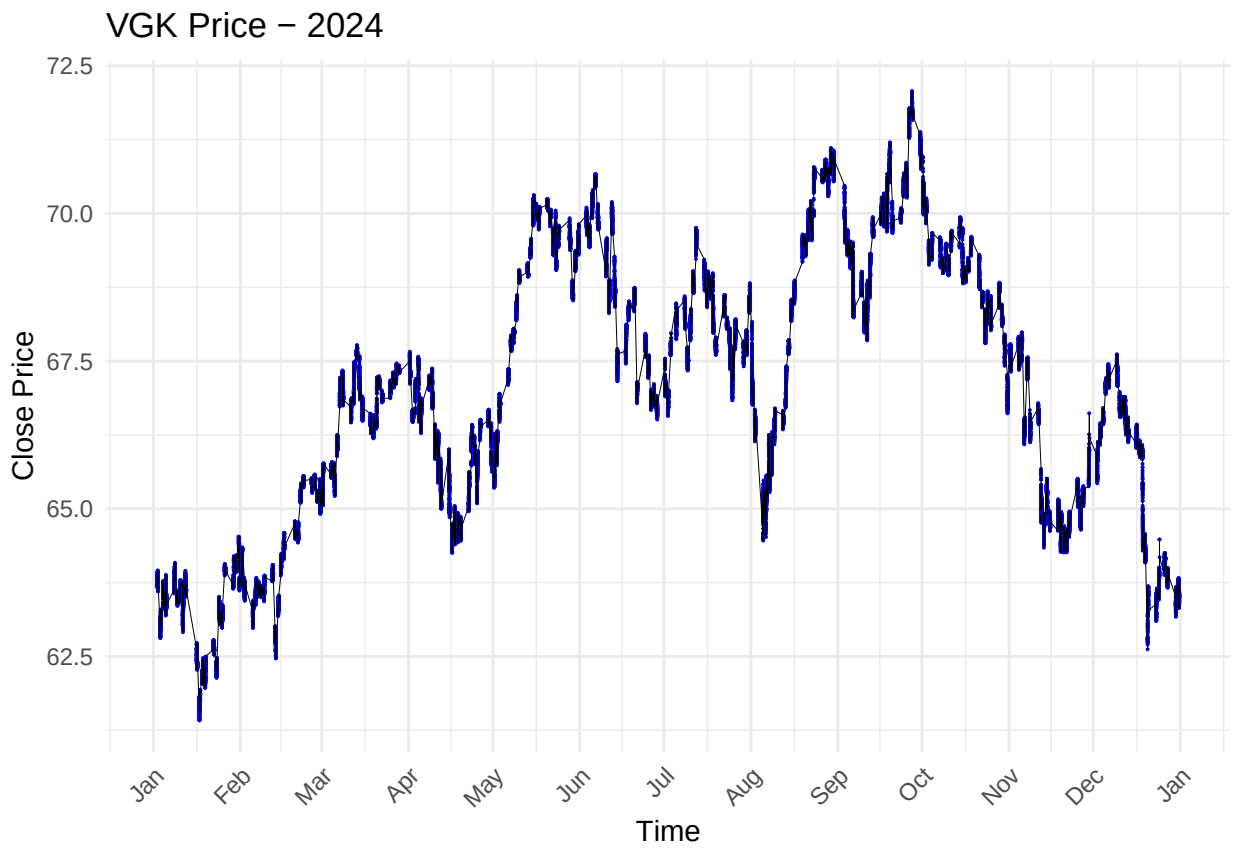
Price

Plots

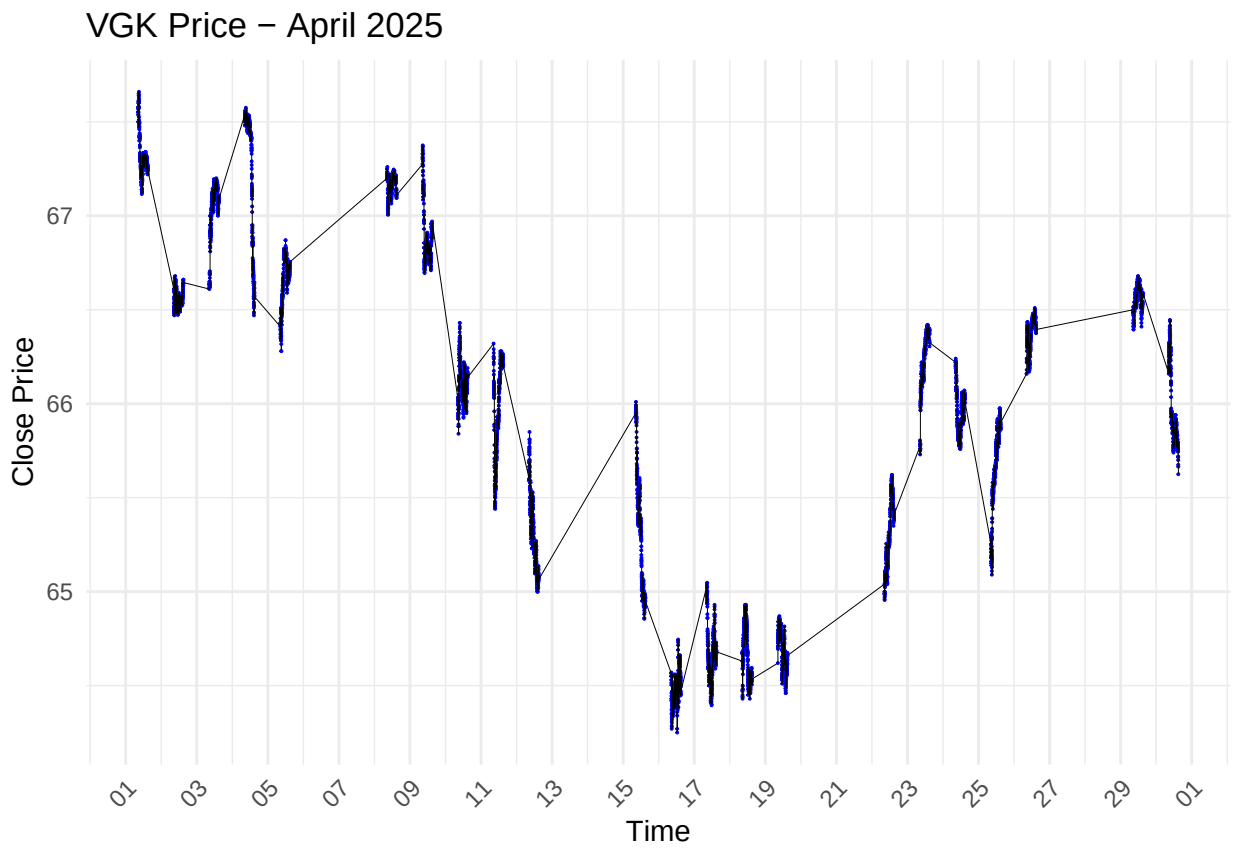
```
#All Time
price_plotter(raw_VGK,"VGK Price Over Time")
```




```
#2024
VGK_2024 = year_selector(raw_VGK,2024)
price_plotter_year(VGK_2024,"VGK Price - 2024")
```



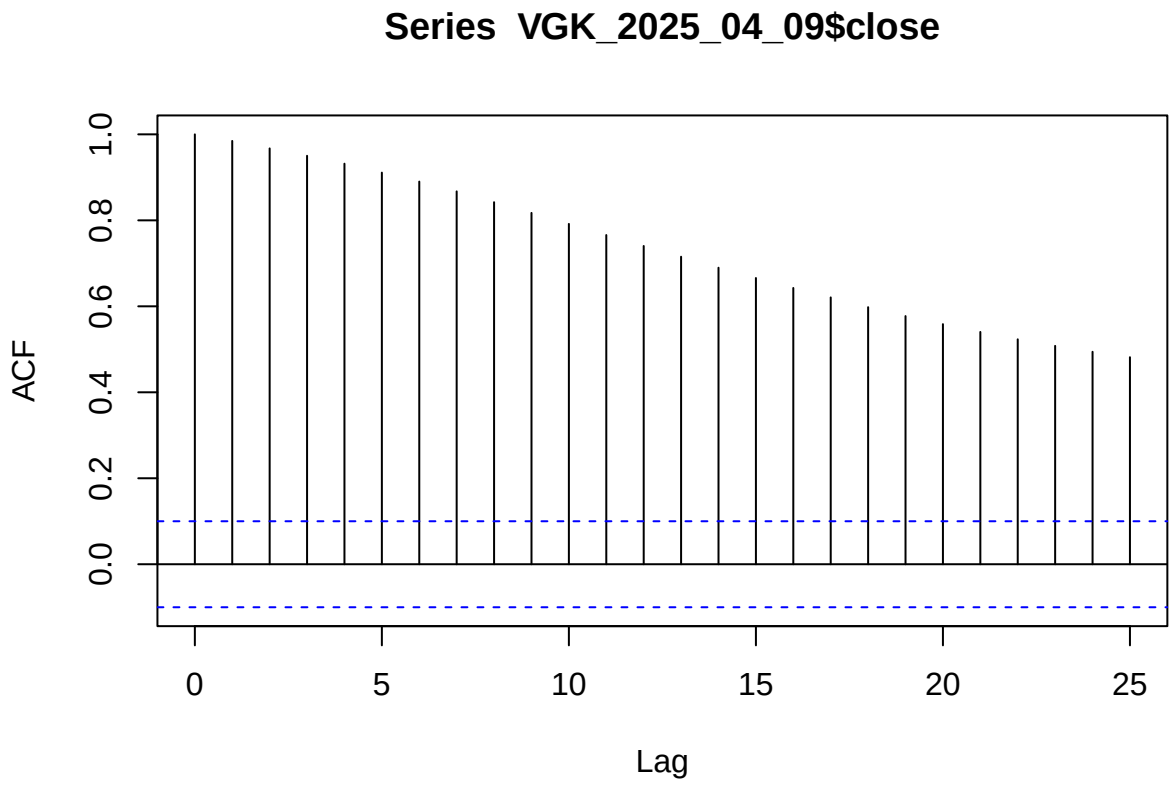

```
#April 2025
VGK_2025_04 = month_selector(raw_VGK,2024,04)
price_plotter_month(VGK_2025_04,"VGK Price - April 2025")
```



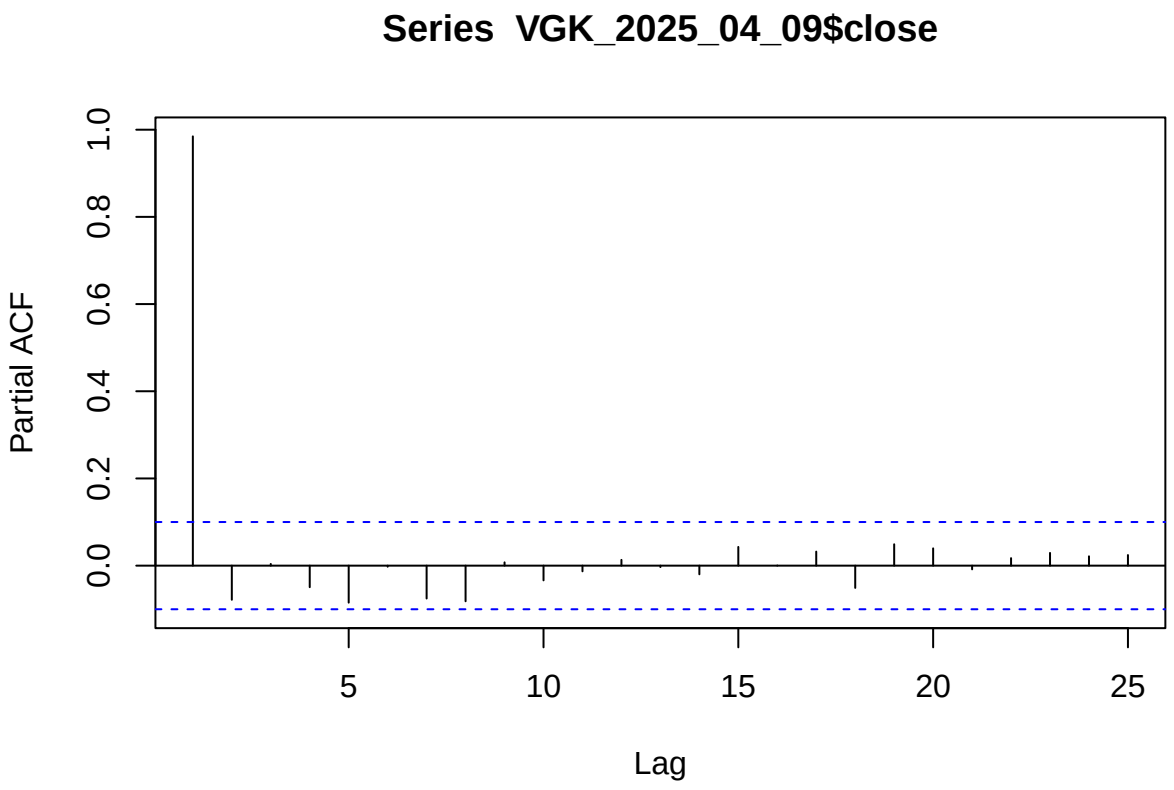

```
#9th of April 2025
VGK_2025_04_09 = day_selector(raw_VGK,2024,04,09)
price_plotter_day(VGK_2025_04_09,"VGK Price - 9th of April 2025")
```

Time Series Analysis

```
acf(VGK_2025_04_09$close)
```




```
pacf(VGK_2025_04_09$close)
```




```
AR1 = arima(VGK_2025_04_09$close,c(1,0,0),method="ML")
AR2 = arima(VGK_2025_04_09$close,c(2,0,0),method="ML")
```

```
## Warning in arima(VGK_2025_04_09$close, c(2, 0, 0), method = "ML"): possible
## convergence problem: optim gave code = 1
```

```
AR3 = arima(VGK_2025_04_09$close,c(3,0,0),method="CSS")
table1 = export_summs(AR1,AR2,AR3, model.names = c("AR1","AR2","AR3"), digits = 4)
```

```
## Warning in FUN(X[[i]], ...): tidy() does not return p values for models of
## class data.frame; significance stars not printed.
```

```
## Warning in FUN(X[[i]], ...): tidy() does not return p values for models of
## class data.frame; significance stars not printed.
## Warning in FUN(X[[i]], ...): tidy() does not return p values for models of
## class data.frame; significance stars not printed.
```

```
huxtable::caption(table1) <- "AR Estimations"
huxtable::set_width(table1, 0.8)
```

```
AR1res = as.numeric(AR1$residuals)
AR1res_lagged <- lag(AR1res, 1)
iidcheck1 = lm(AR1res ~ AR1res_lagged)
```

Table 1: AR Estimations

	AR1	AR2	AR3
ar1	0.9964	1.0562	1.0355
	(0.0034)	(0.0512)	(0.0510)
intercept	66.9997	66.9947	66.8277
	(0.1528)	(0.1454)	(0.0550)
ar2		-0.0602	-0.0247
		(0.0513)	(0.0731)
ar3			-0.0267
			(0.0503)
nobs	384	384	384
sigma	0.0166	0.0166	0.0163
logLik	1026.0693	1026.7507	
AIC	-2046.1385	-2045.5015	
BIC	-2034.2866	-2029.6989	
nobs.1	384.0000	384.0000	384.0000

*** p < 0.001; ** p < 0.01; * p < 0.05.

```
AR2res = as.numeric(AR2$residuals)
AR2res_lagged <- lag(AR2res, 1)
iidcheck2 = lm(AR2res ~ AR2res_lagged)
AR3res = as.numeric(AR3$residuals)
AR3res_lagged <- lag(AR3res, 1)
iidcheck3 = lm(AR3res ~ AR3res_lagged)
table2 = export_summs(iidcheck1,iidcheck2,iidcheck3,
            model.names = c("AR1 Residuals","AR2 Residuals","AR3 Residuals"),
            digits = 4)
huxtable::caption(table2) <- "Checking Residuals"
huxtable::set_width(table2, 0.8)
```

Table 2: Checking Residuals

	AR1 Residuals	AR2 Residuals	AR3 Residuals
(Intercept)	-0.0012	-0.0012	-0.0000
	(0.0008)	(0.0008)	(0.0008)
AR1res_lagged	0.0588		
	(0.0510)		
AR2res_lagged		-0.0030	
		(0.0511)	
AR3res_lagged			-0.0007
			(0.0513)
N	383	383	383
R2	0.0035	0.0000	0.0000

*** p < 0.001; ** p < 0.01; * p < 0.05.

Realised Volatility

Computations

```
#avg per day for each month of any dataset
vol_VGK_daily = r.vol_daily(raw_VGK,merge=F)
head(vol_VGK_daily)
```

timestamp	r__vol__d
2020-01-02	8.77e-05
2020-01-03	0.000126
2020-01-06	8.31e-05
2020-01-07	0.000101
2020-01-08	0.00013
2020-01-09	6.74e-05

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

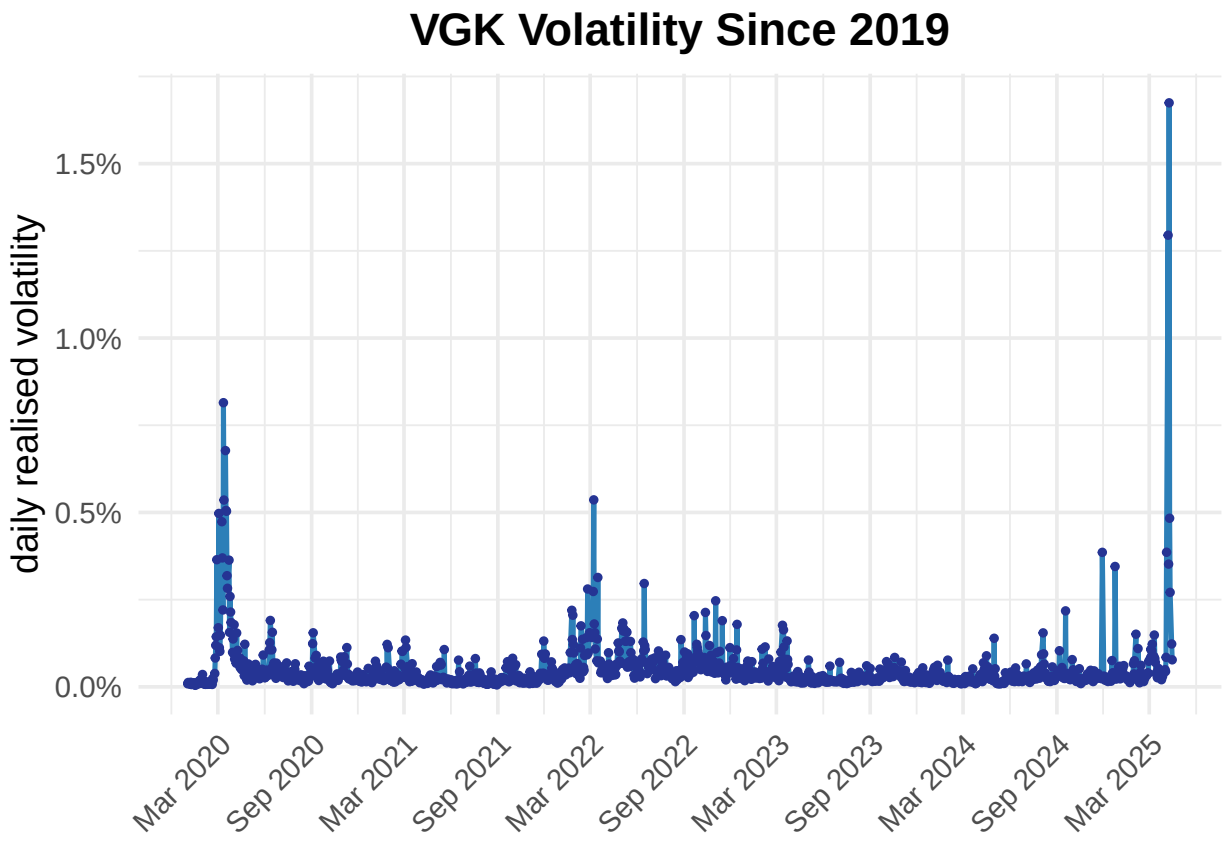
```
#avg per hour for each day of each month of any dataset
vol_VGK_hourly = r.vol_hourly(raw_VGK,merge=F)
head(vol_VGK_hourly)
```

timestamp	r__vol__h
2020-01-02 09:00:00	0.000165
2020-01-02 10:00:00	0.00014
2020-01-02 11:00:00	0.00018
2020-01-02 12:00:00	4.2e-05
2020-01-02 13:00:00	3.07e-05
2020-01-02 14:00:00	4.02e-05

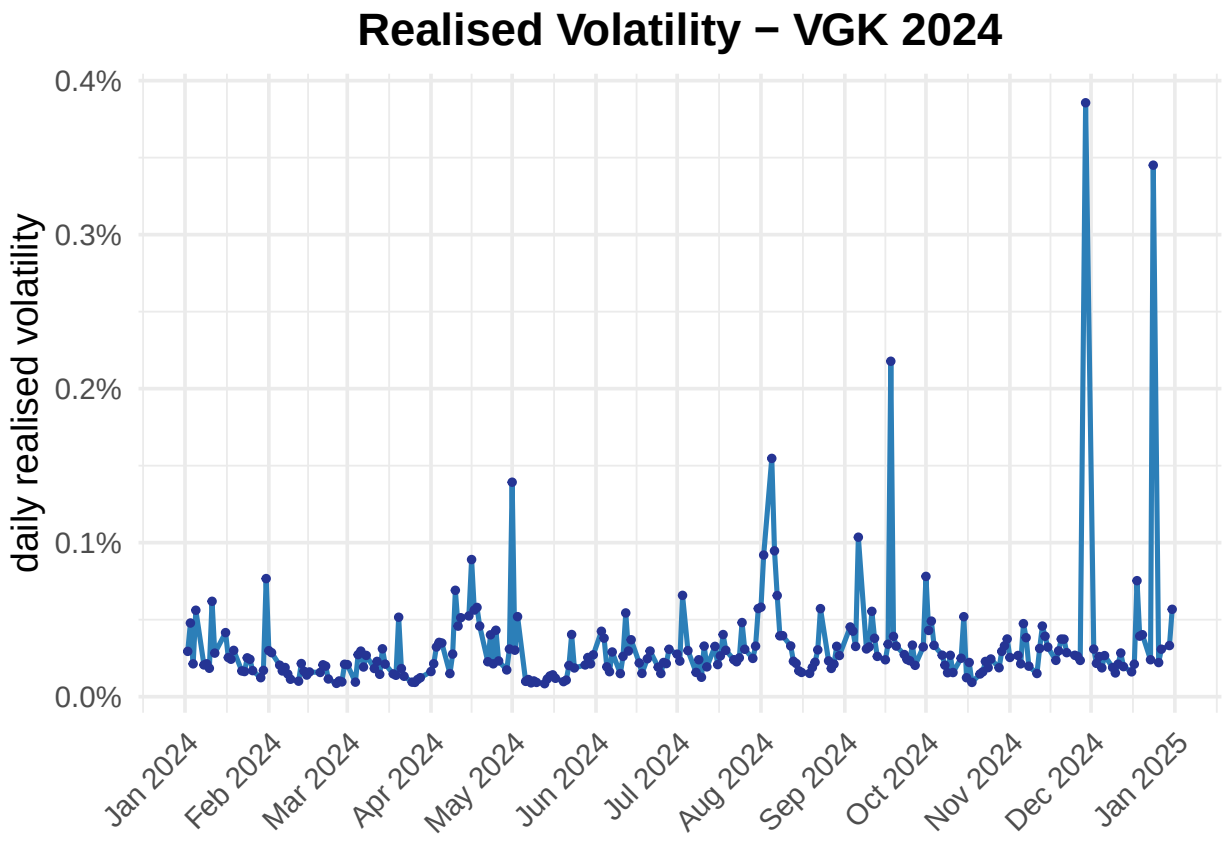
```
#can then filter out years, months, or days
vol_24h = year_selector(vol_VGK_hourly,2024)
vol_24_08h = month_selector(vol_VGK_hourly,2024,08)
vol_24_11_04h = day_selector(vol_VGK_hourly,2024,11,04) #vector
```

Plots

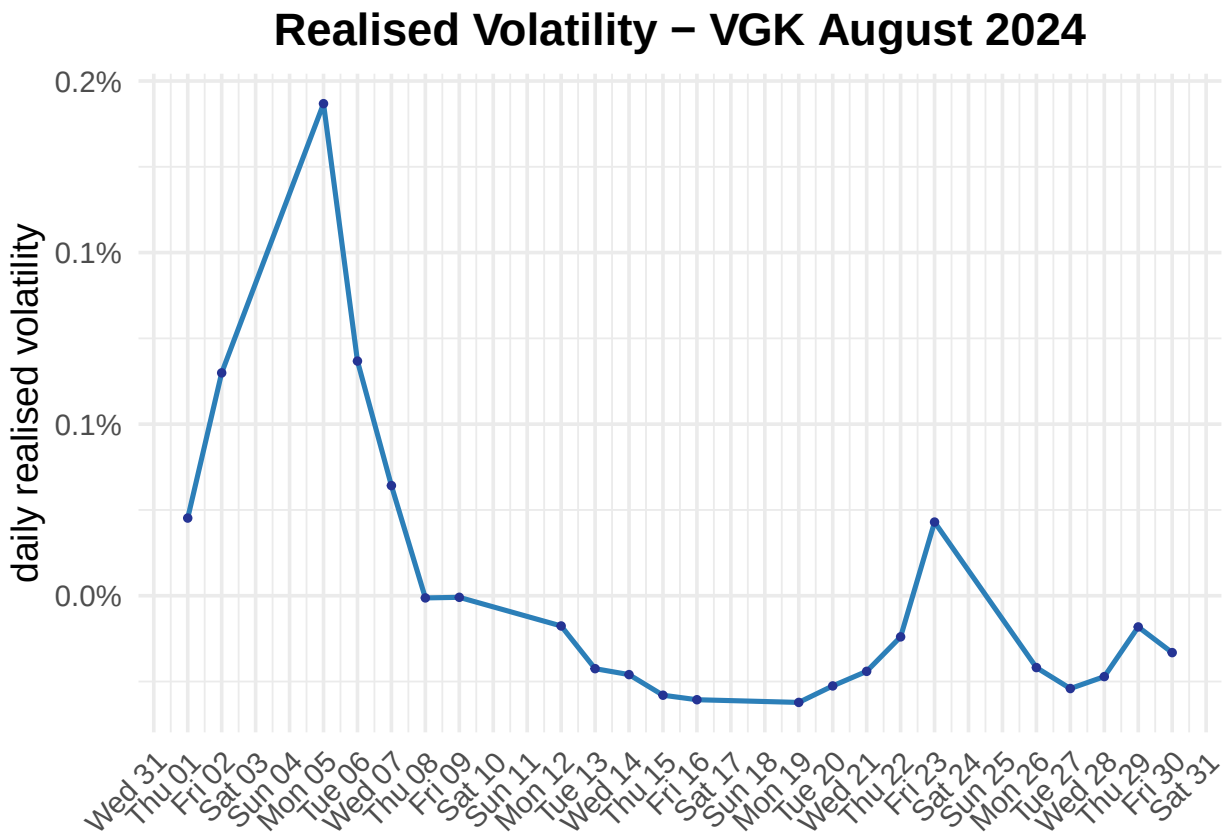
```
#avg per day volatility all time
dvol_plotter(vol_VGK_daily,breaks="yearly",
             title="VGK Volatility Since 2019")
```




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

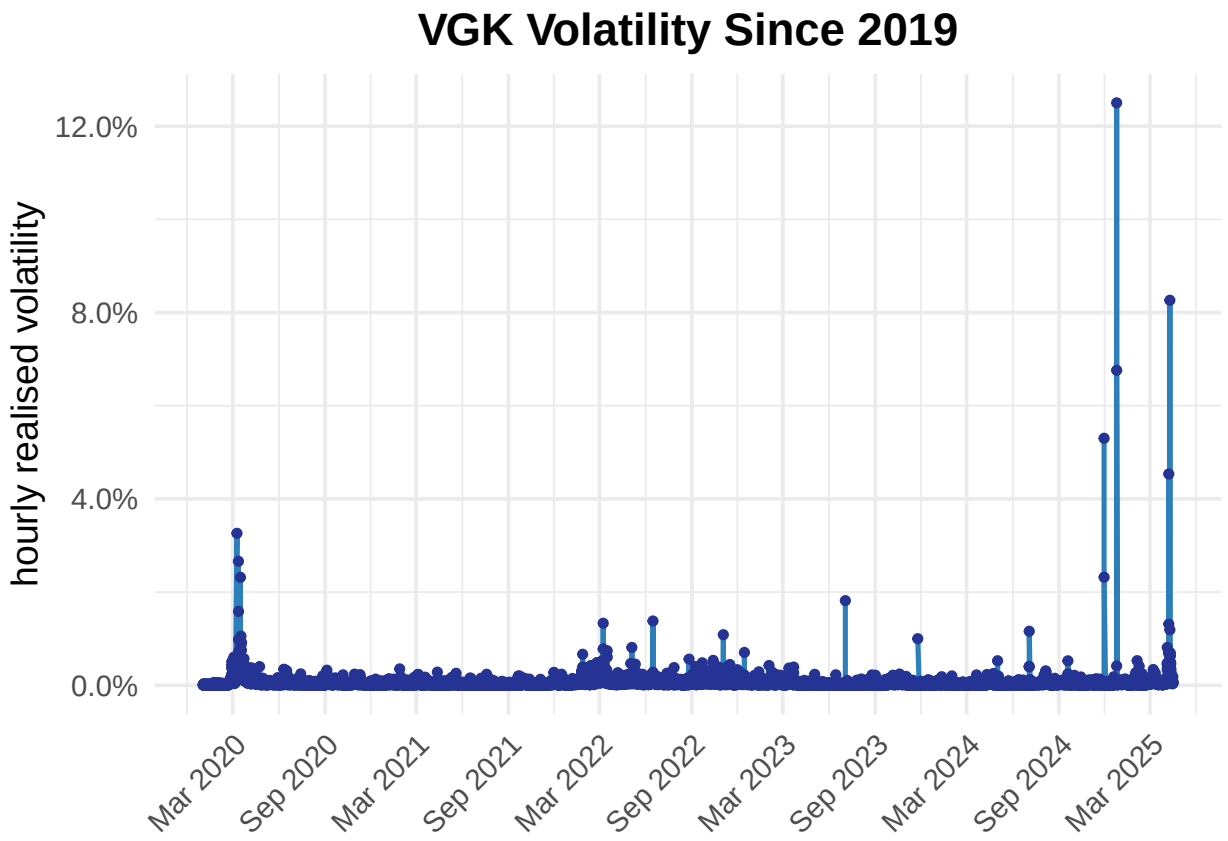



```
#avg per day volatility in a month
dvol_plotter(vol_24_08d,breaks="daily",
             title="Realised Volatility - VGK August 2024")
```



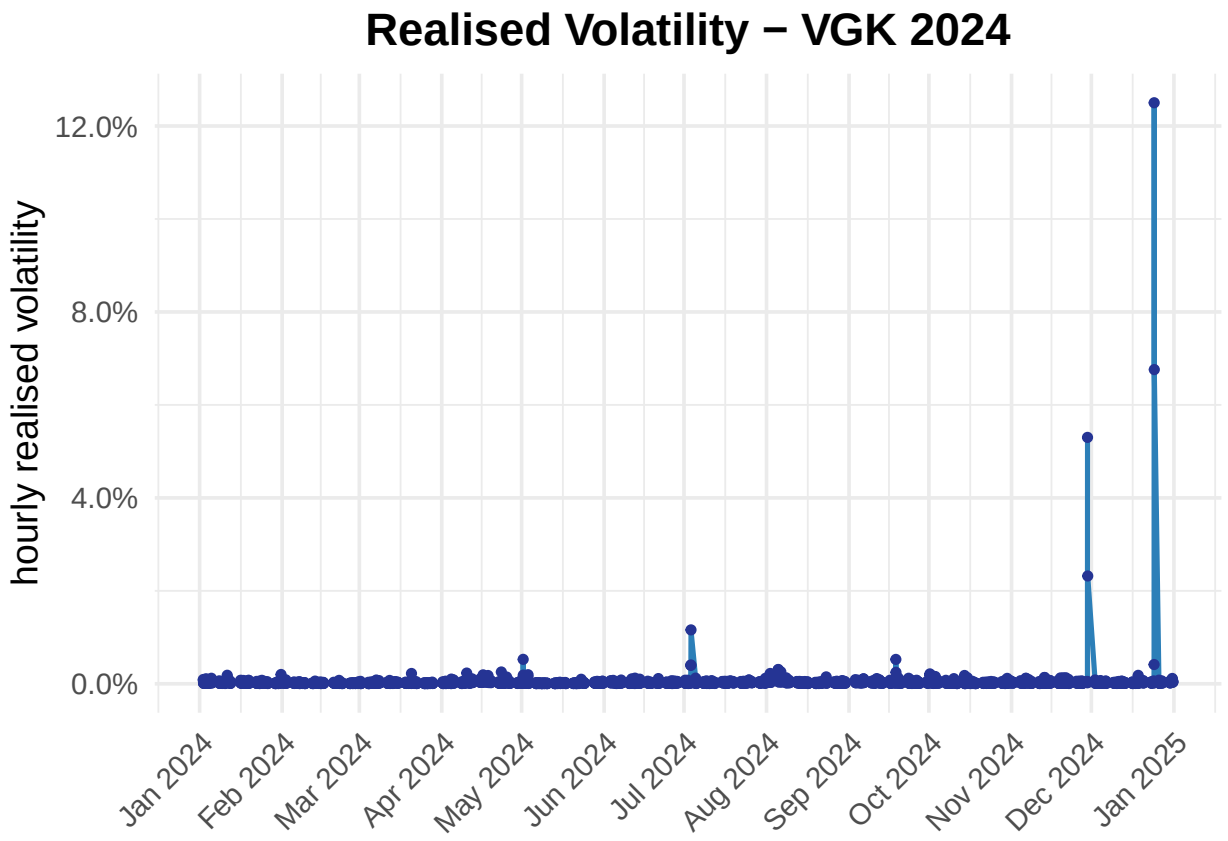

```
#hourly volatility all time
hvol_plotter(vol_VGK_hourly,breaks="yearly",
             title="VGK Volatility Since 2019")
```

```
## Warning: Removed 6 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

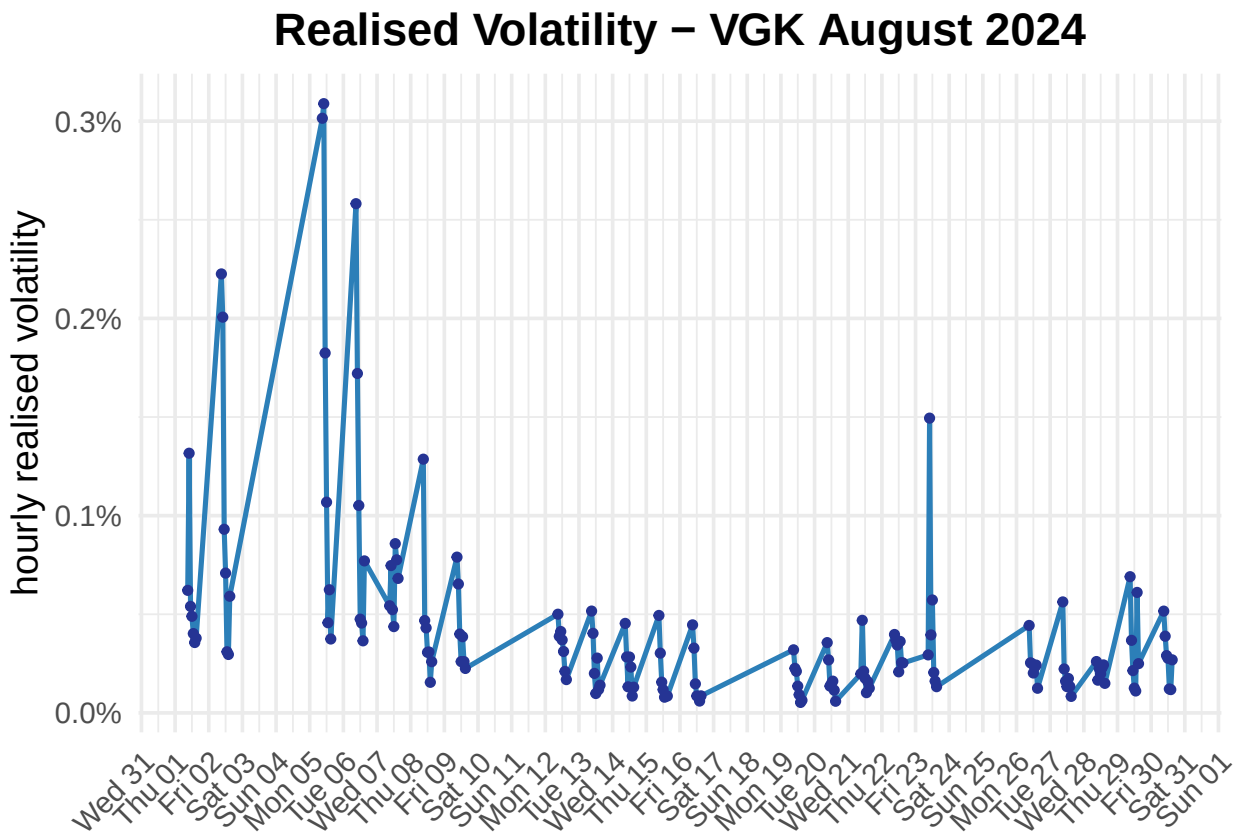



```
#hourly volatility in a year
hvol_plotter(vol_24h,breaks="monthly",
             title="Realised Volatility - VGK 2024")
```

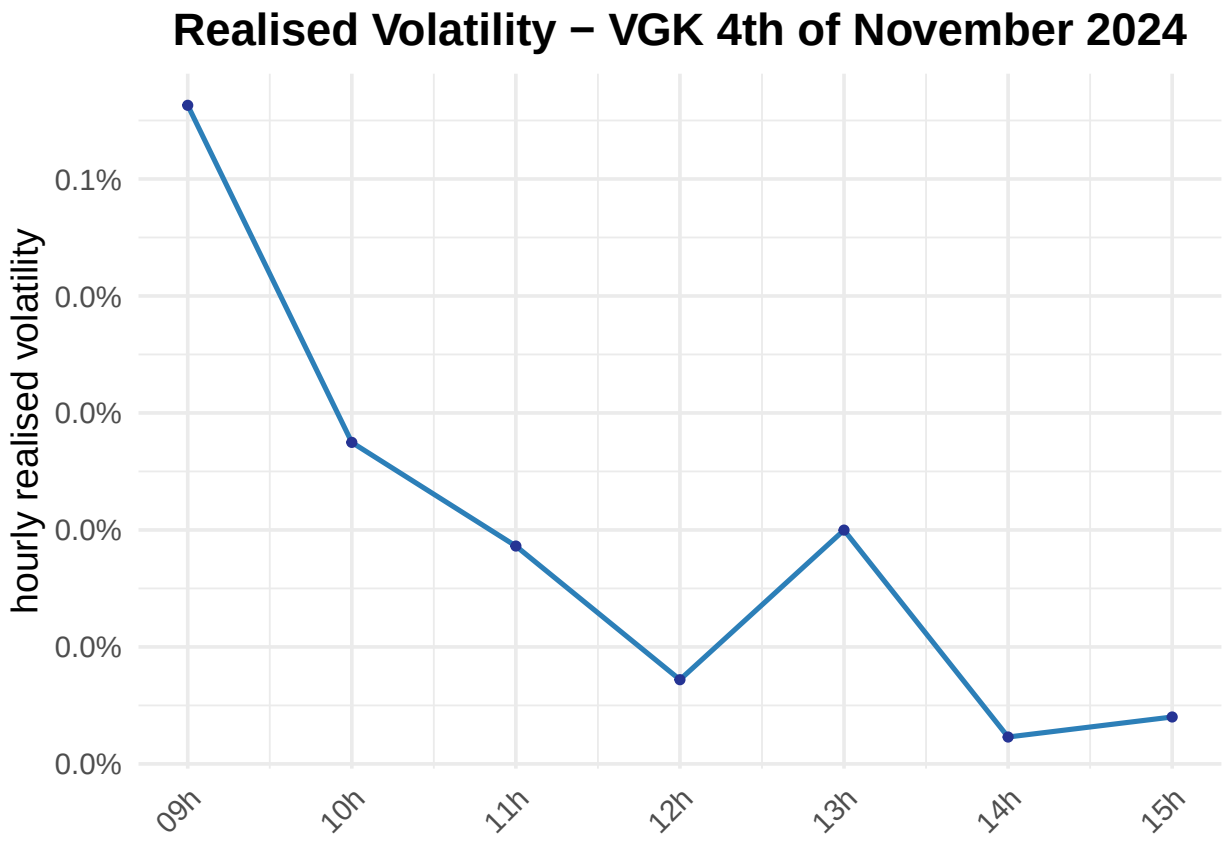
```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
```




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




```
#hourly volatility in a day
hvol_plotter(vol_24_11_04h,breaks="hourly",
             title="Realised Volatility - VGK 4th of November 2024")
```


Realised Volatility − VGK 4th of November 2024
hourly realised volatility
0.1%
0.0%
0.0%
0.0%
0.0%
0.0%
09h
10h
11h
12h
13h
14h
15h