Coursework - Advanced Statistical Finance

CID: 02091191

April 28, 2022

1 First part: extreme value theory

Consider in this first part a dataset containing daily closing prices of Europe Brent Spot Price FOB in dollars per barrel from 20/05/1987 to 01/03/2021 (source: U.S. Energy Information Administration). Increases in energy prices, especially oil prices, affect consumers, corporations, governments and have very important consequences in both economic activity but also our every day life. The aim of this part is to conduct conditional extreme value theory to compute risk measure estimates the oil market.

1.1 GARCH(1,1) model

As developed in the Lecture Notes [1], standard extreme value theory (EVT) applied directly to financial time series (returns, losses, etc.) is not efficient since the iid assumption is not realistic. This problem can me mitigated by first transforming or filtering the data in a suitable way, which leads to conditional EVT. For example, consider a GARCH(1,1) process $X = (X_t)_{t\in\mathbb{Z}}$ given by

$$X_t = \mu + \sigma_t Z_t$$

where $\mu \in \mathbb{R}$ is a constant and $Z = (Z_t)_{t\in\mathbb{Z}} \sim \mathrm{SWN}(0,1)$ and

$$\sigma_t^2 = \alpha_0 + \alpha_1 (X_{t-1} - \mu)^2 + \beta_1 \sigma_{t-1}^2$$

with parameters $\alpha_0 > 0$, $\alpha_1 \geq 0$ and $\beta_1 \geq 0$. Assuming standard normal errors, conditional EVT consists in applying the peak-over-threshold (POT) method to Z instead of X. In practice, Z is non observable and the POT method is applied to the standardised residuals

$$\widehat{Z_t} = \frac{X_t - \widehat{\mu}}{\widehat{\sigma_t}}$$

Here, one can fit a GARCH(1,1) model including a mean term and assuming standard normal errors to the returns (log-returns more precisely) on the entire sample. The parameter estimates (maximum likelihood estimates) of the model are reported in Table 1. Standard errors are relatively small compared to the estimates themselves which assess goodness of fit. Goodness of fit can also be seen on Figure 1.1 where fitted volatilities seem to follow relatively well the absolute returns on the considered period.

$\widehat{\alpha}_0$	$\widehat{\alpha}_1$	$\widehat{\beta}_1$	$\widehat{\mu}$	T
0.0547	0.0924	0.9022	0.0411	8576
(0.0092)	(0.0059)	(0.0060)	(0.0192)	

Table 1: Estimation results of the GARCH(1,1) model with $Z_0 \sim \mathcal{N}(0,1)$ fitted to Brent returns from 20/05/1987 to 01/03/2021. Standard errors are reported in parentheses.

One can now focus on the standardised residuals Z and plot how they vary with time as shown on Figure 1.2, which is consistent regarding the normal or strict white noise assumption. Figures 1.3 and 1.4 suggest that the residuals are consistent with the idd assumption based on the ACFs of the residuals and their absolute values. However, Figure 1.5 shows that their distribution is clearly non-normal given the spikes in the time series plot shown on Figures 1.2 and 1.1 and the heavier tails on the histogram and QQ-plot.

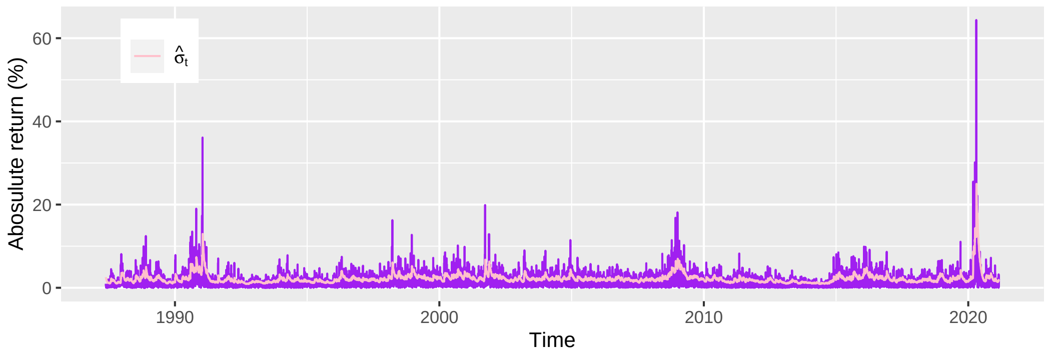


Figure 1.1: Fitted volatilities of the GARCH(1,1) process estimated wit Brent returns data from 20/05/1987 to 01/03/2021. `R` package `rugarch` was used.

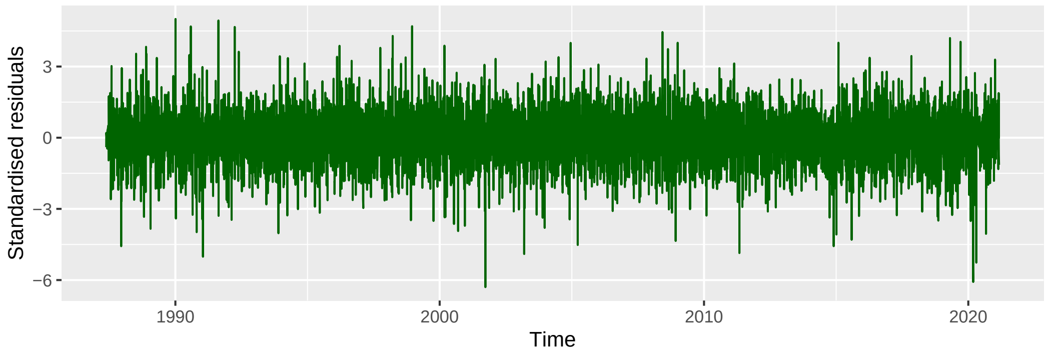


Figure 1.2: Standardised residuals of the GARCH(1,1) process estimated wit Brent returns data from 20/05/1987 to 01/03/2021.

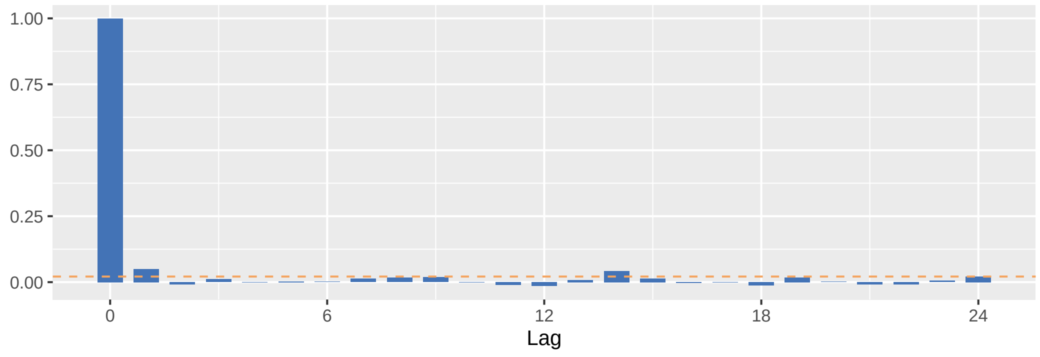


Figure 1.3: ACF plot of GARCH(1,1) standardised residuals.

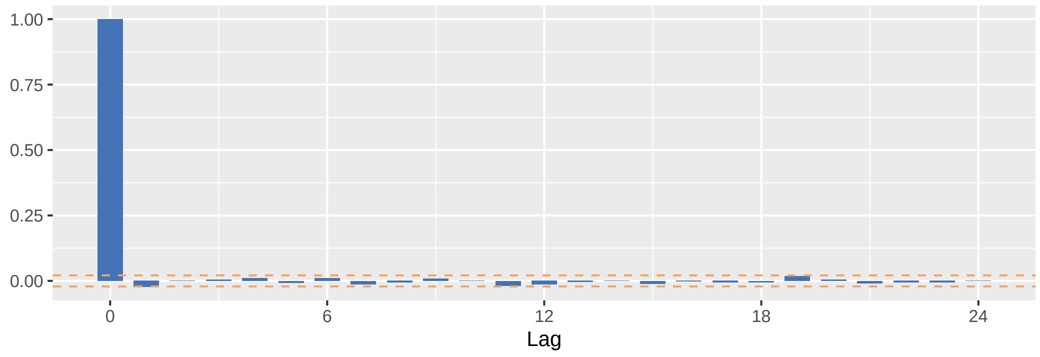
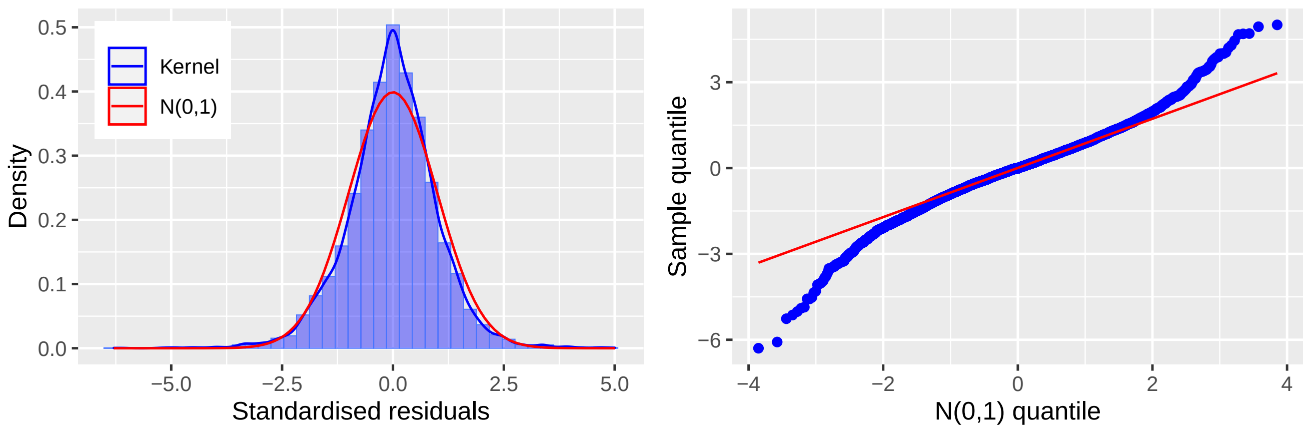


Figure 1.4: ACF plot of the absolute value of GARCH(1,1) standardised residuals.



(a) Histogram and density. (b) QQ-plot.

Figure 1.5: Diagnostic plot of the standardised residuals of the GARCH(1,1) process.

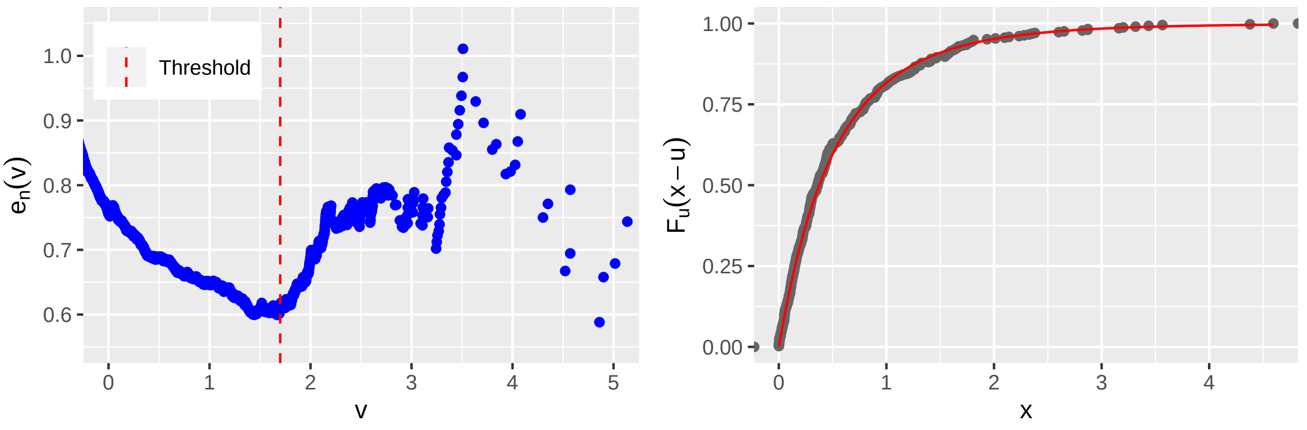
Therefore, the iid assumption of the residuals holds in our model: the correlation between $\widehat{Z}_t$ and $\widehat{Z}_{t-k}$ for any lag k seems to remain relatively low. However, residuals are not normally distributed.

1.2 Mean excess function

One may now model the left tail of the error distribution by applying the POT method to $-\widehat{Z}_t$, $t = 1, ..., T$. The sample mean excess function $e_n(v)$ is defined as

$$e_n(v) = \frac{\sum_{i=1}^{n}(Z_i - v)\mathbf{1}_{(Z_i > v)}}{\sum_{i=1}^{n}\mathbf{1}_{(Z_i > v)}}, \; v \in \mathbb{R}$$

Hence, one can plot the sample mean excess function for the obtained residuals Z from the GARCH(1,1) model as shown on Figure 1.6a. The choice of a suitable threshold will be discussed further in this report.



(a) Sample mean excess. (b) Fitted GPD.

Figure 1.6: Sample mean excess plot (left) of Brent losses and the fitted GPD (right) using threshold $u = 1.7$. `R` package `qrmtools` was used.

1.3 Generalised Pareto distribution (GPD)

Using the sample mean excess plot shown on Figure 1.6a, we identify $u = 1.7$ as a plausible threshold for a GPD fit even if a linear trend after this value is not clearly evident. The fitted GPD is shown in red on Figure 1.6b and seems to fit relatively well the data points. The maximum likelihood estimates of the fitted GPD distribution are reported in Table 2. The standard errors of the estimates are relatively low compared to the estimates themselves which shows a good fit the GPD distribution.

u	N_u	$\widehat{\xi}$	$\widehat{\beta}$	n	$1-\frac{N_u}{n}$
1.7	410	0.1575	0.5133	8576	0.9522
		(0.0612)	(0.0402)		

Table 2: Estimation results of the GPD distribution fitted to the residuals left tail. Standard errors are reported in parentheses.

The threshold is close to the 95% quantile of the data, so the GPD is fitted to the highest 5% of the losses. As shown in the Lecture Notes [1], for $\xi > 0$ the GPD reduces to the ordinary Pareto distribution with tail index $\alpha = \frac{1}{\xi} \simeq 6.3479$, which shows that the fitted GPD has finite 6th moments and thus is easily square integrable.

The choice of the threshold $u = 1.7$ is however somewhat subjective and it is wise to be prudent with the sensitivity of the result regarding changes in $\widehat{\xi}$ estimates. On Figure 1.7, ξ is estimated using maximum likelihood for u varying between 1.5 and 2. This plot shows that using $u = 1.7$ as a threshold is relatively robust: choosing lower values of u, the confidence interval around $\widehat{\xi}$ would be more narrow but the estimate $\widehat{\xi}$ would be biased, and choosing higher values of u, the estimate becomes increasingly noisy due to the paucity of observations. Choosing $u = 1.7$ seems to be a good bias-variance tradeoff.

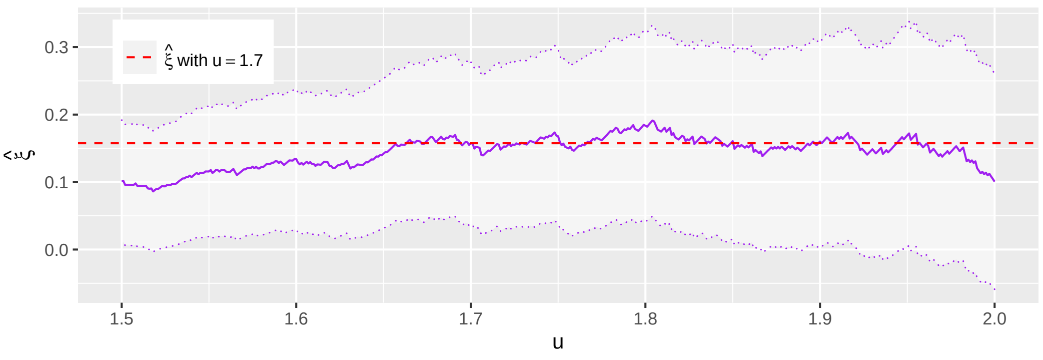


Figure 1.7: Sensitivity analysis of the choice of the threshold $u \in [1.5, 2]$. The light grey band indicates a 95 confidence interval for $\widehat{\xi}$.

1.4 VaR forecasts

Conditional EVT is often used for risk measure estimation. Assume that daily losses $L = (L_t)_{t\in\mathbb{Z}}$ follow the model $L_t = \mu + \sigma_t Z_t$ for $t \in \mathbb{Z}$. Given filtration $\mathbb{F}^L = (\mathcal{F}_t^L)_{t\in\mathbb{Z}}$ where $\mathcal{F}_t^L = \sigma\{L_t, L_{t-1}, ...\}$, the conditional approach to risk measurements uses the value-at-risk (VaR)

$$\mathrm{VaR}_\alpha(L_{t+1}|\mathcal{F}_t^L) = \mu + \sigma_{t+1} q_\alpha(Z_{t+1})$$

where q_α denotes the α-quantile of the distribution of Z. Under the assumption that the tail of the cdf F of Z_{t+1} is perfectly described by a GPD above the threshold u for some ξ and β, we have

$$q_\alpha(Z_{t+1}) = u + \frac{\beta}{\xi}\left(\left(\frac{1-\alpha}{\bar{F}(u)}\right)^{-\xi} - 1\right)$$

Therefore, in practice we use

$$\widehat{\mathrm{VaR}_\alpha(L_{t+1}|\mathcal{F}_t^L)} = \widehat{\mu} + \widehat{\sigma}_{t+1} \underbrace{\left(u + \frac{\widehat{\beta}}{\widehat{\xi}}\left(\left(\frac{1-\alpha}{1-\widehat{F}(u)}\right)^{-\widehat{\xi}} - 1\right)\right)}_{=:\widehat{q}_\alpha(Z_{t+1})}$$

where $\widehat{\xi}$, $\widehat{\beta}$ and $\widehat{F}(u)$ are estimators of each quantity from the previous GPD fit and $\widehat{\mu}$ and $\widehat{\sigma_{t+1}}$ are estimators obtained by the GARCH(1,1) model. When considering a standard normal distribution, the

α-quantile q_α is then simply replaced by the α-quantile of a standard normal distribution.

Hence, using a one-step volatility forecast $\widehat{\sigma_{n+1}}$ from the estimated GARCH(1,1) model and the fitted GPD, one can compute day-ahead VaR forecasts at 95% and 99% confidence levels where $n+1$ is the next business day index. Here, we use the fitted GPD parameters estimates $\widehat{\xi}$ and $\widehat{\beta}$ along with a GARCH(1,1) model whose parameters are re-estimated every business day using a rolling window of 500 days (as in the Lecture Notes [1]). The threshold u is fixed at $u = 1.7$.

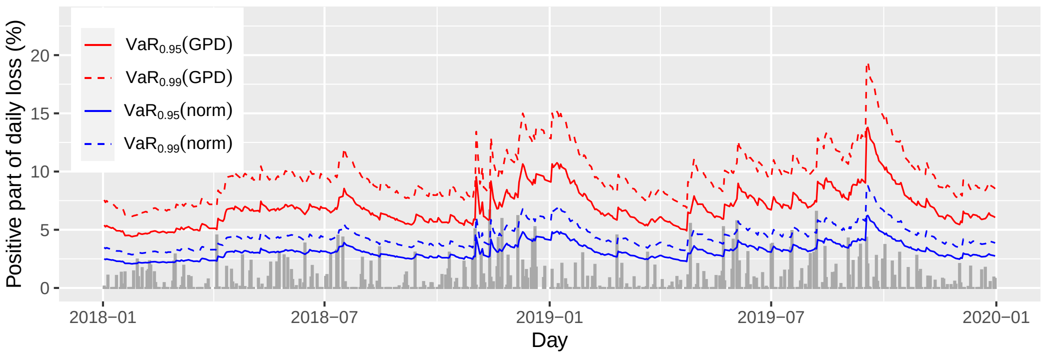


Figure 1.8: Daily $\mathrm{VaR}_{0.95}$ and $\mathrm{VaR}_{0.99}$ estimates on the Brent prices data using a GPD and a $\mathcal{N}(0,1)$ distribution. For convenience, the plot only displays forecasts between 01/01/2018 and 31/12/2019 even if the fit has been done on the entire sample. `R` package `SpatialExtremes` was used.

VaR forecasts displayed on Figure 1.8 compare the forecasts obtained by using a GPD and a standard normal distribution for the error distribution. One may note that the GPD-based method produces systematically higher estimates of VaR for a given confidence level since it captures better and more faithfully the heaviness of the tail. In particular, the normal-based method for both confidence levels seems to be not that efficient for multiple samples where the VaR forecasts were under-estimating the real daily loss. In comparison, the GPD-based method seems to fit better and give better estimates of VaR: for example on this period and using a GPD, the 95% confidence VaR forecasts were very rarely under-estimating the daily loss and the 99% confidence VaR forecasts has never been under-estimating it. However, one drawback could be a less accurate estimate of the daily loss since it appears (on this period at least) to over-estimate it quite often. In fact, the GPD-based method relies on the threshold u: choosing a smaller value of u would move down the VaR forecasts using GPD, that is lower these estimates. However, one may remain careful interpreting this plot since it represents only a small period of the whole sample and as a comparison, one may have a look to periods when volatility was higher than between 01/01/2018 and 31/12/2019, such as from 01/07/1990 to 01/07/1991 which corresponds to the 1990 oil price shock due to the Iraqi invasion of Kuwait.

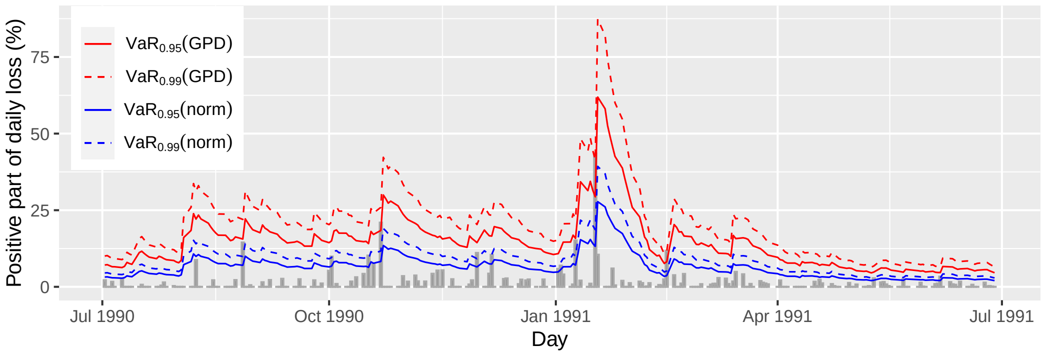


Figure 1.9: Daily $\mathrm{VaR}_{0.95}$ and $\mathrm{VaR}_{0.99}$ estimates on the Brent prices data using a GPD and a $\mathcal{N}(0,1)$ distribution. For convenience, the plot only displays forecasts between 01/07/1990 and 01/07/1991 even if the fit has been done on the entire sample.

On Figure 1.9 are displayed the same VaR forecasts as previously but on a different time period, known for high volatility on the oil price. This plot shows how the oil price volatility made the VaR forecasts much higher than on Figure 1.8. Even using a 99% confidence GPD-based method, a few forecasts are below the actual daily losses. This emphasises the difficulty with setting a suitable threshold u: setting a lower value of u would lower the VaR estimates, which would make the forecasts more precise on low volatility periods but would under-estimate daily losses on high volatility periods. With regard to the estimates in Table 2 and their standard errors as well as the goodness of fit on Figure 1.6 and the forecasts obtained on Figures 1.8 and 1.9, the choice of $u = 1.7$ seems to be appropriate here.

1.5 Higher confidence levels

As suggested, one may focus on the accuracy of the forecasts using a GPD and a standard normal distribution and compare them for higher confidence levels than 99%. Indeed, it is often conjectured that EVT performs well in practice when the VaR confidence level exceeds 99%. This makes sense since EVT is somehow *tailored* for this kind of applications, especially in producing high quality and precise forecasts of extremely rare events. As discussed in the Lecture Notes [1] and in this report, a standard normal distribution does not fit well the standardised residuals especially in the tails. Therefore, when trying to forecast VaR for extremely rare events such as oil crisis here, using a GPD would provide more reliable upper bounds for the daily loss. The main drawback of using a very high confidence level would be the loss of accuracy on the forecast itself. But considering only that a VaR forecast for a given confidence level is a success when the actual daily loss is lower than the forecast we made for it, one can compare the performance of using a standard normal distribution and a GPD for confidence levels higher than 99%.

Hence, considering a grid of confidence levels $\alpha \in [0.99, 1-10^{-6}]$, one can compute the daily VaR forecasts using the same GARCH(1,1) rolling model as previously as well as the same GPD estimates, with the only difference in the quantiles levels. Then, for each confidence level α, we compare each one-day-ahead forecast of the VaR with the actual daily loss and consider the forecast is a success if it is greater than the actual daily loss. This gives a score metric reflecting the accuracy and defined as

$$p = \frac{1}{T}\sum_{t=1}^{T} \mathbf{1}_{(\text{VaR}_{\alpha,t} > L_t)}$$

Therefore, using this metric one can compare the GPD and the normal models for each confidence level α.

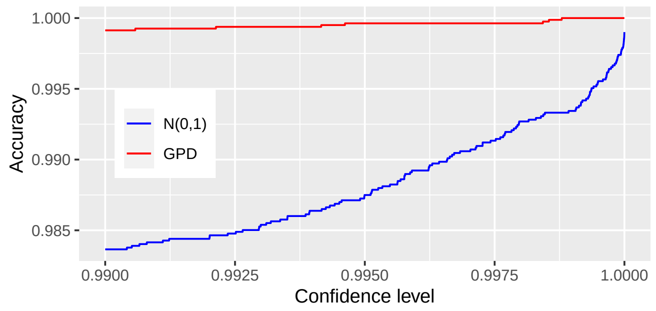


Figure 1.10: Comparison of the accuracy of forecasts between the $\mathcal{N}(0,1)$ model and the GPD model for confidence levels $\alpha \in [0.99, 1-10^{-6}]$.

Figure 1.10 shows that for all $\alpha \in [0.99, 1-10^{-6}]$, the GPD model has a higher accuracy than the $\mathcal{N}(0,1)$ model. Thus, it confirms that using an appropriate EVT approach to model extremely rare events is more efficient than modelling the residuals using a standard normal distribution. One may also note that even for very high confidence intervals (close to 1), the standard normal model struggle to reach 100%: this is due to the quantiles of the GPD being more appropriate to model the tail of $-\widehat{Z_t}$ distribution. However, note that using a confidence level very close to 1 such that $1-10^{-6}$ (upper bound of the interval considered here) is not ideal since it gives very high VaR forecasts which do not provide much information. However, without considering this aspect, Figure 1.10 shows that the GPD is always more accurate than the normal distribution on the entire sample, using the accuracy metric p defined above.

2 Second part: high frequency modelling and volatility estimation

In this part, we are interested in volatility estimation with the stock of Verizon Communications (VZ). The dataset used contains intraday midprice from 02/01/2019 to 31/12/2019, with at each row the time stamp measured as the number of seconds from midnight and the corresponding midprice. For seconds where there is no record of the price, we will assume it equal to the one of the closest timestamp before it in the data. No external `R` package is used for this part.

2.1 Daily realised variance

Practitioners often aim to measure the volatility modelled as a stochastic process $\sigma = (\sigma_t)_{t\in[0,T]}$ through the averaged quantity

$$IV_{[s,t]} = \int_s^t \sigma_u^2 du$$

which is defined as the integrated variance, taking $[s,t] = [0,T]$ where T is the number of seconds in a trading day. The integrated variance is not directly observable since we do not observe σ and therefore needs to be estimated. Suppose we observe the stock price S at regular frequencies $S_0, S_\Delta, S_{2\Delta}, ..., S_{n\Delta}$ where $n\Delta = T$. The realised variance of the observations is defined as

$$RV_n = \sum_{i=1}^{n} (\log S_{i\Delta} - \log S_{(i-1)\Delta})^2$$

In fact, one can show as in the Lecture Notes [1] that RV is a consistent estimator of IV as $n \to \infty$. Moreover, under some technical assumptions on μ and σ, we have

$$\frac{\sqrt{n}(RV_n - IV)}{\sqrt{\frac{2n}{3}RQ_n}} = \frac{(RV_n - IV)}{\sqrt{\frac{2}{3}RQ_n}} \xrightarrow[n\to\infty]{d} \mathcal{N}(0,1)$$

where RQ_n denotes the realised quarticity of the observations defined as

$$RQ_n = \sum_{i=1}^{n} (\log S_{i\Delta} - \log S_{(i-1)\Delta})^4$$

Hence, one can use this result to derive asymptotic confidence intervals for IV. Given a confidence level $\alpha \in (0,1)$, we have

$$\lim_{n\to\infty} \mathbb{P}\left[\Phi^{-1}\left(\frac{1-\alpha}{2}\right) \le \frac{(RV_n - IV)}{\sqrt{\frac{2}{3}RQ_n}} \le \Phi^{-1}\left(\frac{1+\alpha}{2}\right)\right] = \alpha$$

where Φ denotes the cdf of a $\mathcal{N}(0,1)$ distribution. Thus, an asymptotic confidence interval for IV at α level is

$$\left[RV_n - \Phi^{-1}\left(\frac{1+\alpha}{2}\right)\sqrt{\frac{2}{3}RQ_n},\ RV_n + \Phi^{-1}\left(\frac{1+\alpha}{2}\right)\sqrt{\frac{2}{3}RQ_n}\right]$$

since $\Phi^{-1}\left(\frac{1+\alpha}{2}\right) = -\Phi^{-1}\left(\frac{1-\alpha}{2}\right)$.

Then, using the realised variance estimator and the derived confidence interval, we compute the daily realised variance on each business day in three consecutive months from 01/07/2019 to 30/09/2019 using each of the frequencies $\Delta \in \{5\text{min}, 1\text{min}, 30\text{sec}\}$. We also compute a 95% confidence interval for the integrated variance for each business day.

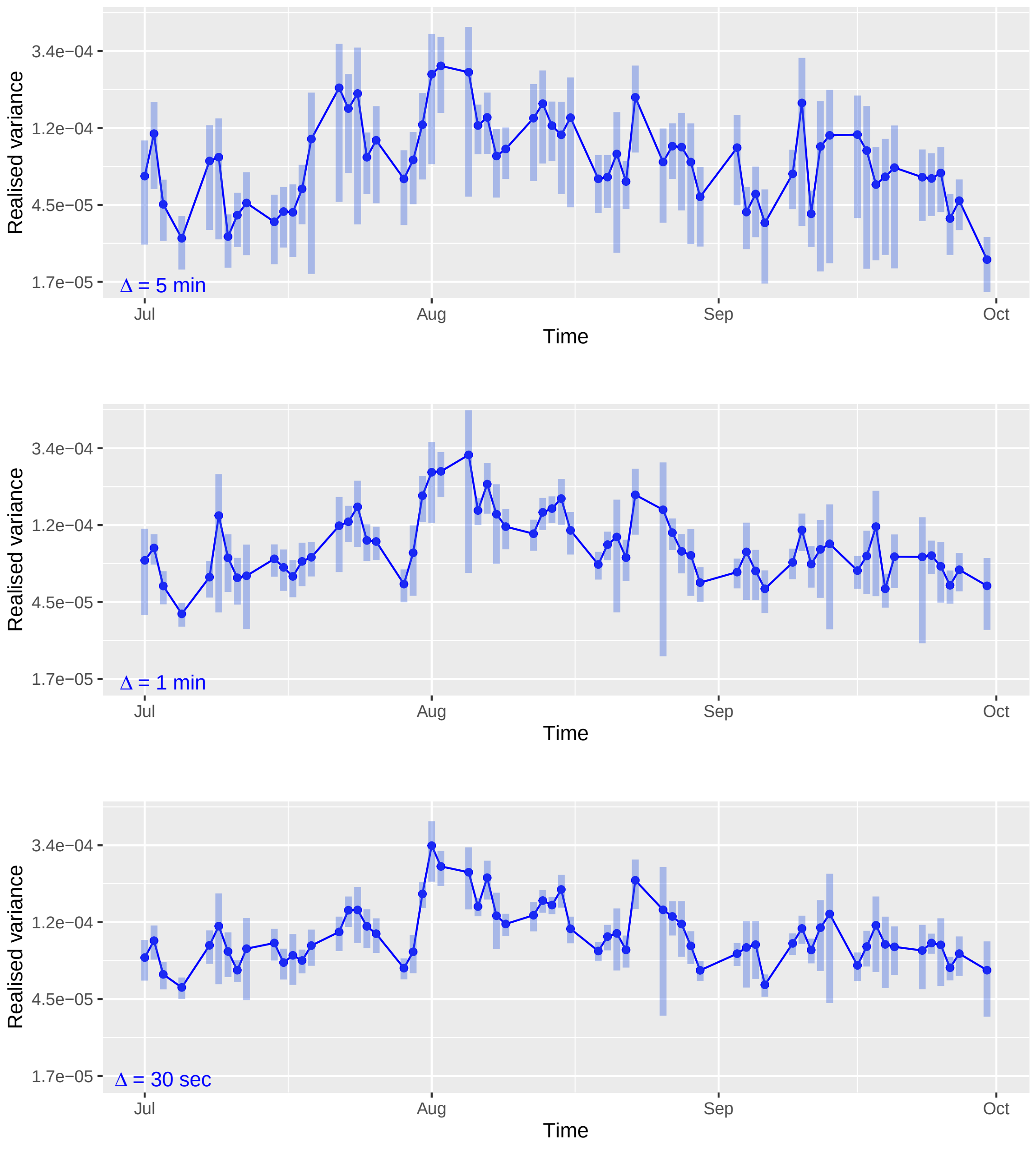


Figure 2.1: Daily realised variance estimates of Verizon Communications, using the midprice sampled with $\Delta \in \{5\text{min}, 1\text{min}, 30\text{sec}\}$, and their 95% confidence intervals. Note that y-axis uses log scale.

Figure 2.1 shows the daily realised variance on Verizon stock, from 01/07/2019 to 29/09/2019 along with their 95% confidence intervals sampled at different time frequencies Δ. This plot is in accordance with the asymptotic theorem stated above since we observe the shrinkage of confidence intervals while we go from $\Delta = 5$min to $\Delta = 30$sec. Some confidence intervals remain somehow wide as in the end of August for example, but they seem to be more and more narrow as Δ decreases. Note that the estimated levels of the realised variance remain roughly the same with $\Delta = 1$min and $\Delta = 30$sec but are slightly different using $\Delta = 5$min, although the global evolution of the realised variance is the same across all frequencies. The high levels of realised variance in August are the result of an escalation in U.S.-China trade relations (tweets of Donald Trump) and a recession signal being flashed by the bond market (source).

2.2 Signature plot of RV

Considering a market microstructure (MMS) noise term and at ultra high frequency, one can easily see that RV_n is no longer a consistent estimator of IV anymore. Indeed, the RV_n estimator is biased under

MMS noise (Hansen and Lunde, 2006)

$$\mathbb{E}[RV_n] = IV + 2n\nu^2$$

where $\mu = 0$, σ is non-random and $\varepsilon_1, ..., \varepsilon_n$ are iid independent of $\tilde{S}$ (latent price) and satisfy $\mathbb{E}[\varepsilon_i] = 0$, $\nu^2 = \mathbb{E}[\varepsilon_i^2] < \infty$. Moreover, Hansen and Lunde prove that under some assumptions, $\mathbb{V}[RV_n]$ is asymptotically proportional to n. Thus, $\mathbb{E}[RV_n] \xrightarrow[n\to\infty]{} \infty$ and $\mathbb{V}[RV_n] \xrightarrow[n\to\infty]{} \infty$.

The aim of this question is to assess the impact of MMS noise through signature plots of RV. We compute the daily realised variance $RV_n(d) = RV_{n_\Delta}(d)$ for each business day of 2019 in the dataset $d = 1, ..., D$ and frequencies $\Delta > 0$. We then calculate the average realised variance

$$ARV(\Delta) = \frac{1}{\Delta}\sum_{d=1}^{D} RV_{n_\Delta}(d)$$

for different values of Δ. In the absence of MMS noise, $ARV(\Delta)$ would stabilise as $\Delta \to 0$. We therefore compute the ARV on the whole year for $\Delta \in [1\text{sec}, 10\text{min}]$ with a time step of 1sec. More precisely, we compute the relative change from $ARV(\Delta)$ with $\Delta = 5\text{min}$ which is considered as the benchmark frequency and is unlikely to be affected with MMS noise. Hence, the plotted quantity is

$$DRV(\Delta) = \frac{ARV(\Delta) - ARV(5\text{min})}{ARV(5\text{min})}$$

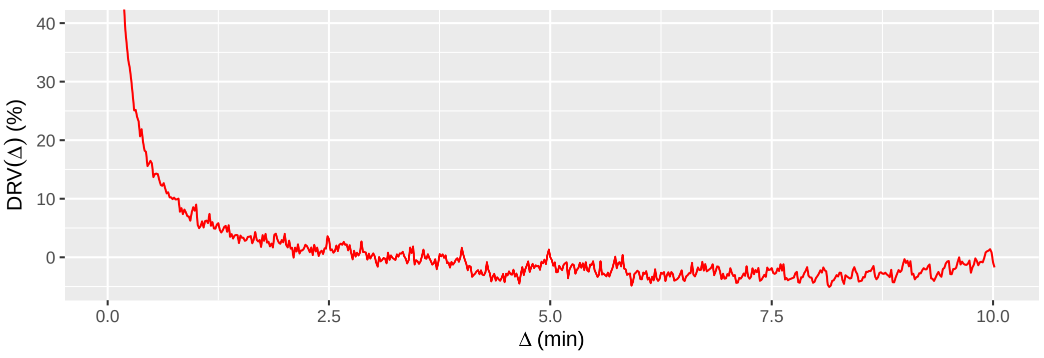


Figure 2.2: Signature plots of RV for Verizon (year 2019) using frequencies from $\Delta = 1\text{sec}$ to $\Delta = 10\text{min}$. The midprice is used to compute RV.

Figure 2.2 shows a relatively flat signature plot apart from random fluctuations and where $ARV(\Delta)$ increases rapidly as $\Delta \to 0$ which makes sense regarding the Hansen and Lunde property stated previously. Indeed, bias in RV seems to increase as $\Delta \to 0$. Considering $ARV(5\text{min})$ as a benchmark, the plot shows that $DRV(\Delta)$ is relatively constant and equal to 0 from $\Delta = 2.5\text{min}$. In fact, practitioners face a tradeoff: choosing Δ very close to 0 would be preferable in order to get more consistent and reliable IV estimates and confidence intervals (according to the asymptotic results stated previously), but it tends to make the estimator RV biased when considering the MMS noise. In the contrary, sticking to high values of Δ would be unsatisfactory from the point of view of efficient use of data. Here, a suitable value for Δ would be $\Delta = 2.5\text{min}$ since it is the lowest value of Δ where ARV is somehow stabilised around the benchmark $ARV(5\text{min})$.

As suggested in the Lecture Notes [1], modifying the RV estimator would help to mitigate the impact of MMS noise. The autocorrelation corrected realised variance estimator (Zhou, 1996) provides a bias correction, although it remains not consistent due to its variance. The realised kernel estimator (Barndorff-Nielsen, 2008) is unbiased and consistent.

References

[1] Mikko Pakkanen, Nikolas Kantas (March 2022) *MATH70070 - Advanced Statistical Finance 2021-2022 Lecture Notes*, Imperial College London MSc Statistics resources

[2] Mikko Pakkanen, Axel Gandy, Ciara Pike-Burke (January 2022) *MATH70079 - Introduction to Statistical Finance 2021-2022 Lecture Notes*, Imperial College London MSc Statistics resources

R code

```r
library(ggplot2)
library(rugarch)
library(dplyr)
library(cowplot)
library(qrmtools)
library(latex2exp)

# Taken from https://rh8liuqy.github.io/ACF_PACF_by_ggplot2.html
ggplot.corr <- function(data, lag.max = 24, ci = 0.95, large.sample.size = TRUE,
                        horizontal = TRUE,...) {

  if(horizontal == TRUE) {numofrow <- 1} else {numofrow <- 2}

  list.acf <- acf(data, lag.max = lag.max, type = "correlation", plot = FALSE)
  N <- as.numeric(list.acf$n.used)
  df1 <- data.frame(lag = list.acf$lag, acf = list.acf$acf)
  df1$lag.acf <- dplyr::lag(df1$acf, default = 0)
  df1$lag.acf[2] <- 0
  df1$lag.acf.cumsum <- cumsum((df1$lag.acf)^2)
  df1$acfstd <- sqrt(1/N * (1 + 2 * df1$lag.acf.cumsum))
  df1$acfstd[1] <- 0
  df1 <- select(df1, lag, acf, acfstd)

  list.pacf <- acf(data, lag.max = lag.max, type = "partial", plot = FALSE)
  df2 <- data.frame(lag = list.pacf$lag,pacf = list.pacf$acf)
  df2$pacfstd <- sqrt(1/N)

  if(large.sample.size == TRUE) {
    plot.acf <- ggplot(data = df1, aes( x = lag, y = acf)) +
      geom_area(aes(x = lag, y = qnorm((1+ci)/2)*acfstd), fill = "#B9CFE7") +
      geom_area(aes(x = lag, y = -qnorm((1+ci)/2)*acfstd), fill = "#B9CFE7") +
      geom_col(fill = "#4373B6", width = 0.7) +
      scale_x_continuous(breaks = seq(0,max(df1$lag),6)) +
      scale_y_continuous(name = element_blank(),
                         limits = c(min(df1$acf,df2$pacf),1))

    plot.pacf <- ggplot(data = df2, aes(x = lag, y = pacf)) +
      geom_area(aes(x = lag, y = qnorm((1+ci)/2)*pacfstd), fill = "#B9CFE7") +
      geom_area(aes(x = lag, y = -qnorm((1+ci)/2)*pacfstd), fill = "#B9CFE7") +
      geom_col(fill = "#4373B6", width = 0.7) +
      scale_x_continuous(breaks = seq(0,max(df2$lag, na.rm = TRUE),6)) +
      scale_y_continuous(name = element_blank(),
                         limits = c(min(df1$acf,df2$pacf),1)) +
      xlab("Lag")
  }
  else {
    plot.acf <- ggplot(data = df1, aes( x = lag, y = acf)) +
      geom_col(fill = "#4373B6", width = 0.7) +
      geom_hline(yintercept = qnorm((1+ci)/2)/sqrt(N),
                 colour = "sandybrown",
                 linetype = "dashed") +
      geom_hline(yintercept = - qnorm((1+ci)/2)/sqrt(N),
                 colour = "sandybrown",
                 linetype = "dashed") +
      scale_x_continuous(breaks = seq(0,max(df1$lag),6)) +
      scale_y_continuous(name = element_blank(),
                         limits = c(min(df1$acf,df2$pacf),1)) +
      xlab("Lag")

    plot.pacf <- ggplot(data = df2, aes(x = lag, y = pacf)) +
      geom_col(fill = "#4373B6", width = 0.7) +
      geom_hline(yintercept = qnorm((1+ci)/2)/sqrt(N),
                 colour = "sandybrown",
                 linetype = "dashed") +
      geom_hline(yintercept = - qnorm((1+ci)/2)/sqrt(N),
                 colour = "sandybrown",
                 linetype = "dashed") +
      scale_x_continuous(breaks = seq(0,max(df2$lag, na.rm = TRUE),6)) +
      scale_y_continuous(name = element_blank(),
                         limits = c(min(df1$acf,df2$pacf),1)) +
      xlab("Lag")
  }
  plot.acf
}

data <- read.csv("Europe_Brent_Spot_Price_FOB.csv", sep=",")
colnames(data) <- c("Day", "Price")
data$Day <- as.Date(data$Day, "%m/%d/%Y")
data <- data[order(data$Day),]
data$Log_return <- c(NA, 100 * diff(log(data$Price), lag=1))
data$Abs_log_return <- abs(data$Log_return)
data <- data[-1,]

ggplot(data=data, aes(x=Day, y=Price)) +
  geom_line() +
  xlab("Time") + ylab("Price")

spec <- ugarchspec(variance.model=list(model="sGARCH", garchOrder=c(1, 1)),
                 mean.model=list(armaOrder=c(0, 0)), distribution.model="norm")
fit <- ugarchfit(spec=spec, data=data$Log_return)
fit

fitted_volatilities <- fit@fit$sigma
residuals_ <- residuals(fit, standardize=TRUE)
residuals <- c()
for(i in 1:length(residuals_)){
  residuals <- c(residuals, residuals_[[i]])
}


data$Residuals <- residuals
data$Abs_residuals <- abs(data$Residuals)
data$Fitted_volatilities <- fitted_volatilities

ggplot(data=data) +
  geom_line(aes(x=Day, y=Abs_log_return), colour="purple") +
  geom_line(aes(x=Day, y=Fitted_volatilities, colour="pink")) +
  scale_colour_manual(name=NULL, values=c('pink'='pink'),
                    labels = c(TeX(r'($\hat{\sigma}_t$)'))) +
  xlab("Time") + ylab("Abosulute return (%)") +
  theme(legend.position=c(0.1, 0.85))

# Plot of the residuals as a function of time
ggplot(data=data, aes(x=Day, y=Residuals)) +
  geom_line(colour="darkgreen") +
  xlab("Time") + ylab("Standardised residuals")

# ACF of residuals
ggplot.corr(data=data$Residuals, large.sample.size=FALSE, ci=0.95)

# ACF of absolute value of residuals
ggplot.corr(data=data$Abs_residuals, large.sample.size=FALSE, ci=0.95)

# Histogram of residuals
ggplot(data, aes(x=Residuals)) +
  geom_histogram(aes(y = ..density..), bins=40, fill="blue", alpha=0.4,
               colour="royalblue1", lwd=.2) +
  geom_density(lwd = .5, aes(colour = "blue")) +
  stat_function(fun = dnorm, args = list(mean=0, sd=1),
              aes(colour="red")) +
  labs(x="Standardised residuals", y="Density") +
  scale_color_manual(name=NULL, values=c("blue"="blue", "red"="red"),
                    labels=c("Kernel", "N(0,1)")) +
  theme(legend.position=c(0.15, 0.8))

# QQ-plot of residuals
ggplot(data, aes(sample=Residuals)) +
  stat_qq(colour="blue") +
  stat_qq_line(colour="red") +
  xlab("N(0,1) quantile") + ylab("Sample quantile")

# Mean excess function
mean_excess <- as.data.frame(mean_excess_np(-data$Residuals))
ggplot(mean_excess, aes(x=x, y=mean.excess)) +
  geom_point(colour="blue") +
  geom_vline(aes(colour="red", xintercept=1.7), linetype="dashed") +
  coord_cartesian(xlim=c(0, 5), ylim=c(0.55, 1.05)) +
  xlab("v") + ylab(expression(e[n](v))) +
  scale_color_manual(name=NULL, values=c("red"="red"), labels=c("Threshold")) +
  theme(legend.position = c(0.17, 0.85))

# GPD
shapes <- c()
lower <- c()
upper <- c()
us <- seq(1, 2, by=0.001)
for(u in us){
  fit_GPD <- fit_GPD_MLE(-data$Residuals[-data$Residuals > u] - u)
  shapes <- c(shapes, fit_GPD$par[1])
  lower <- c(lower, fit_GPD$par[1] - qnorm(0.975) * fit_GPD$SE[1])
  upper <- c(upper, fit_GPD$par[1] + qnorm(0.975) * fit_GPD$SE[1])
}
ggplot() +
  geom_ribbon(aes(x=us, ymin = lower, ymax = upper), alpha=0.5, fill="white",
             linetype="dotted", colour="purple", size=0.4) +
  geom_line(aes(x=us, y=shapes), colour="purple") +
  geom_hline(aes(yintercept=shapes[which(us == 1.5)], colour="h"), linetype="dashed",
            size=0.5) +
  scale_color_manual(name=NULL, values=c("h"="red"),
                    labels=c(expression(hat(xi) ~ "with" ~ u == 1.7))) +
  theme(legend.position=c(0.12, 0.85)) +
  xlab("u") + ylab(expression(hat(xi)))

u <- 1.7
fit_GPD <- fit_GPD_MLE(-data$Residuals[-data$Residuals > u] - u)
shape <- fit_GPD$par[1]
scale <- fit_GPD$par[2]
ggplot() +
  stat_ecdf(aes(x=-data$Residuals[-data$Residuals > u] - u),
           geom="point", colour="gray40") +
  geom_function(fun=pGPD, args=list(shape=shape, scale=scale), colour="red") +
  xlab("x") + ylab(expression(F[u](x-u)))
fit_GPD$par
fit_GPD$SE
length(data$Residuals[-data$Residuals > u])
1 - length(data$Residuals[-data$Residuals > u])/length(data$Residuals)

# VaR estimation
spec <- ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1,1)),
                 mean.model = list(armaOrder = c(0, 0)),
                 distribution.model = "norm")

garch_roll <- ugarchroll(spec=spec, data=data$Log_return, n.start=500,
                      refit.every=1, refit.window='moving', window.size=500,
                      n.ahead=1)

VaR_99_GPD <- garch_roll@forecast$density[,1] +
  garch_roll@forecast$density[,2] *
  SpatialExtremes::qgpd(0.99, loc=u, scale=scale, shape=shape)
VaR_95_GPD <- garch_roll@forecast$density[,1] +
  garch_roll@forecast$density[,2]*
  SpatialExtremes::qgpd(0.95, loc=u, scale=scale, shape=shape)

VaR_99_norm <- garch_roll@forecast$density[,1] +
  garch_roll@forecast$density[,2] * qnorm(0.99, mean=0, sd=1)
VaR_95_norm <- garch_roll@forecast$density[,1] +
  garch_roll@forecast$density[,2]* qnorm(0.95, mean=0, sd=1)

new_data <- data[,c(1, 2)]
new_data$Loss <- sapply(
  (lag(new_data$Price, 1) - lag(new_data$Price, 0))/lag(new_data$Price, 0) * 100,
  function(x) max(x, 0))
new_data$VaR_99_GPD <- c(rep(NA, 500), VaR_99_GPD)
new_data$VaR_99_norm <- c(rep(NA, 500), VaR_99_norm)
new_data$VaR_95_GPD <- c(rep(NA, 500), VaR_95_GPD)
new_data$VaR_95_norm <- c(rep(NA, 500), VaR_95_norm)
new_data <- na.omit(new_data)

ggplot(data=new_data, aes(x=Day, y=Loss)) +
  geom_line(colour="darkgrey") +
  geom_line(aes(x=Day, y=VaR_95_GPD), colour="red", size=0.2) +
  geom_line(aes(x=Day, y=VaR_99_GPD), colour="red", size=0.2, linetype="dashed") +
  geom_line(aes(x=Day, y=VaR_95_norm), colour="blue", size=0.2) +
  geom_line(aes(x=Day, y=VaR_99_norm), colour="blue", size=0.2, linetype="dashed")

new_data_2018_2020 <- new_data[new_data$Day >= "1990-07-01" & new_data$Day < "1991-07-01",]

ggplot(data=new_data_2018_2020, aes(x=Day, y=Loss)) +
  geom_col(colour="darkgrey") +
  geom_line(aes(x=Day, y=VaR_95_GPD, colour="0.95, GPD"), size=0.4) +
  geom_line(aes(x=Day, y=VaR_99_GPD, colour="0.99, GPD"), linetype="dashed", size=0.4) +
```

```r
  geom_line(aes(x=Day, y=VaR_95_norm, colour="0.95, norm"), size=0.4) +
  geom_line(aes(x=Day, y=VaR_99_norm, colour="0.99, norm"), linetype="dashed", size=0.4) +
  scale_colour_manual(name=NULL, values=c('0.95, GPD'='red',
                                          '0.99, GPD'='red',
                                          '0.95, norm'='blue',
                                          '0.99, norm'='blue'),
                    labels = c(expression(VaR[0.95] (GPD)),
                               expression(VaR[0.99] (GPD)),
                               expression(VaR[0.95] (norm)),
                               expression(VaR[0.99] (norm))),
                    guide = guide_legend(override.aes = list(
                      linetype = c("solid", "dashed", "solid", "dashed")))) +
  #ylim(0, 23) +
  ylab("Positive part of daily loss (%)") +
  theme(legend.position=c(0.1, 0.72))

# Higher than 99%
data_cl <- new_data[,c(1, 2, 3)]
confidence_levels <- seq(0.99, 0.999999, length.out=10000)
mean_norm <- c()
mean_GPD <- c()
for(cl in confidence_levels){
  VaR_GPD <- garch_roll@forecast$density[,1] +
    garch_roll@forecast$density[,2] *
    SpatialExtremes::qgpd(cl, loc=u, scale=scale, shape=shape)
  VaR_norm <- garch_roll@forecast$density[,1] +
    garch_roll@forecast$density[,2]* qnorm(cl, mean=0, sd=1)
  data_cl[paste("VaR_GPD", cl, sep="_")] <- VaR_GPD
  data_cl[paste("VaR_norm", cl, sep="_")] <- VaR_norm
  mean_norm <- c(mean_norm, mean(VaR_norm > data_cl$Loss))
  mean_GPD <- c(mean_GPD, mean(VaR_GPD > data_cl$Loss))
}

ggplot() +
  geom_line(aes(x=confidence_levels, y=mean_norm, colour="norm")) +
  geom_line(aes(x=confidence_levels, y=mean_GPD, colour="gpd")) +
  scale_color_manual(name=NULL, values=c("norm"="blue", "gpd"="red"),
                   labels=c("N(0,1)", "GPD")) +
  theme(legend.position=c(0.15, 0.5)) +
  xlab("Confidence level") + ylab("Accuracy")

# Question 2
library(zoo)
library(dplyr)
library(ggplot2)

round_scientific <- function(x){
  function(x) formatC(signif(x, digits=2), format = "e", digits = 1)
}

files_names <- list.files(path="./vz_data/")
rv_rq_list <- c()

for(i in 1:length(files_names)){

  # Progress
  print(paste(i, "/", length(files_names), sep=""))

  # Create the file path
  file_path <- paste("./vz_data/", files_names[i], sep="")
  # Read the file
  data <- read.csv(file_path)
  # Create an additional data frame to complete missing values
  alt_data <- data.frame(list(Var1=34200:57599, Var2=NA))
  # Merge both data frames
  new_data <- merge(alt_data, data, by="Var1", all.x=TRUE)
  new_data$Var2.x <- NULL
  # Impute missing values by the last observed value
  new_data <- na.locf(new_data, fromLast=FALSE, na.rm=TRUE)
  # Small correction for the very first time stamps
  #new_data <- na.locf(new_data, fromLast=TRUE, na.rm=FALSE)
  colnames(new_data) <- c("Seconds", "Midprice")

  # Filter for 3 time delta
  data_5mn <- new_data %>%
    slice(which((row_number() - 1) %% (5 * 60) == 0))
  data_1mn <- new_data %>%
    slice(which((row_number() - 1) %% (1 * 60) == 0))
  data_30s <- new_data %>%
    slice(which((row_number() - 1) %% 30 == 0))

  # Daily realized variance for each delta
  rv_5mn <- sum(diff(log(data_5mn$Midprice), lag=1)^2)
  rv_1mn <- sum(diff(log(data_1mn$Midprice), lag=1)^2)
  rv_30s <- sum(diff(log(data_30s$Midprice), lag=1)^2)

  # Daily realized quarticity for each delta
  rq_5mn <- sum(diff(log(data_5mn$Midprice), lag=1)^4)
  rq_1mn <- sum(diff(log(data_1mn$Midprice), lag=1)^4)
  rq_30s <- sum(diff(log(data_30s$Midprice), lag=1)^4)

  rv_rq_list <- cbind(rv_rq_list, c(rv_5mn, rq_5mn, rv_1mn, rq_1mn, rv_30s, rq_30s))
}

# Select starting and stopping dates (3 months interval)
first <- "VZ_2019-07-01.csv"
last <- "VZ_2019-09-30.csv"
idx_first <- which(files_names == first)
idx_last <- which(files_names == last)
rv_rq <- as.data.frame(rv_rq_list[,idx_first:idx_last])

reshape_name <- function(string){
  strsplit(strsplit(string, split="_")[[1]][2], split=".csv")[[1]]
}
reshaped <- unlist(lapply(files_names[idx_first:idx_last], reshape_name))
reshaped <- as.Date(reshaped, format="%Y-%m-%d")
colnames(rv_rq) <- reshaped
rownames(rv_rq) <- c("RV_5mn", "RQ_5mn", "RV_1mn", "RQ_1mn", "RV_30s", "RQ_30s")

rv_rq <- t(rv_rq)

rv_rq_5mn <- as.data.frame(rv_rq[,1:2])
rv_rq_5mn$lower <- rv_rq_5mn$RV_5mn - qnorm(0.975) * sqrt(2/3 * rv_rq_5mn$RQ_5mn)
rv_rq_5mn$upper <- rv_rq_5mn$RV_5mn + qnorm(0.975) * sqrt(2/3 * rv_rq_5mn$RQ_5mn)

ggplot() +
  geom_line(aes(x=reshaped, y=rv_rq_5mn$RV_5mn), colour="blue") +
  geom_point(aes(x=reshaped, y=rv_rq_5mn$RV_5mn), colour="blue") +
  geom_linerange(aes(x=reshaped, ymin=rv_rq_5mn$lower, ymax=rv_rq_5mn$upper),
                 colour="royalblue", alpha=0.4, size=1.7) +
  scale_y_continuous(trans="log", labels=round_scientific()) +
  coord_cartesian(ylim=c(1.6e-5, 5e-4)) +
  ylab("Realised variance") + xlab("Time") +
  annotate("text", x=as.Date("2019-07-03"), y = 1.6e-5,
          label = expression(Delta ~ "= 5 min"), colour="blue")

rv_rq_1mn <- as.data.frame(rv_rq[,3:4])
rv_rq_1mn$lower <- rv_rq_1mn$RV_1mn - qnorm(0.975) * sqrt(2/3 * rv_rq_1mn$RQ_1mn)
rv_rq_1mn$upper <- rv_rq_1mn$RV_1mn + qnorm(0.975) * sqrt(2/3 * rv_rq_1mn$RQ_1mn)

ggplot() +
  geom_line(aes(x=reshaped, y=rv_rq_1mn$RV_1mn), colour="blue") +
  geom_point(aes(x=reshaped, y=rv_rq_1mn$RV_1mn), colour="blue") +
  geom_linerange(aes(x=reshaped, ymin=rv_rq_1mn$lower, ymax=rv_rq_1mn$upper),
                colour="royalblue", alpha=0.4, size=1.7) +
  scale_y_continuous(trans="log", labels=round_scientific()) +
  coord_cartesian(ylim=c(1.6e-5, 5e-4)) +
  ylab("Realised variance") + xlab("Time") +
  annotate("text", x=as.Date("2019-07-03"), y = 1.6e-5,
          label = expression(Delta ~ "= 1 min"), colour="blue")

rv_rq_30s <- as.data.frame(rv_rq[,5:6])
rv_rq_30s$lower <- rv_rq_30s$RV_30s - qnorm(0.975) * sqrt(2/3 * rv_rq_30s$RQ_30s)
rv_rq_30s$upper <- rv_rq_30s$RV_30s + qnorm(0.975) * sqrt(2/3 * rv_rq_30s$RQ_30s)

ggplot() +
  geom_line(aes(x=reshaped, y=rv_rq_30s$RV_30s), colour="blue") +
  geom_point(aes(x=reshaped, y=rv_rq_30s$RV_30s), colour="blue") +
  geom_linerange(aes(x=reshaped, ymin=rv_rq_30s$lower, ymax=rv_rq_30s$upper),
                colour="royalblue", alpha=0.4, size=1.7) +
  scale_y_continuous(trans="log", labels=round_scientific()) +
  coord_cartesian(ylim=c(1.6e-5, 5e-4)) +
  ylab("Realised variance") + xlab("Time") +
  annotate("text", x=as.Date("2019-07-03"), y = 1.6e-5,
          label = expression(Delta ~ "= 30 sec"), colour="blue")



rv_list <- c()

for(i in 1:length(files_names)){

  # Progress
  print(paste(i, "/", length(files_names), sep=""))

  # Create the file path
  file_path <- paste("./vz_data/", files_names[i], sep="")
  # Read the file
  data <- read.csv(file_path)
  # Create an additional data frame to complete missing values
  alt_data <- data.frame(list(Var1=34200:57599, Var2=NA))
  # Merge both data frames
  new_data <- merge(alt_data, data, by="Var1", all.x=TRUE)
  new_data$Var2.x <- NULL
  # Impute missing values by the last observed value
  new_data <- na.locf(new_data, fromLast=FALSE, na.rm=TRUE)
  # Small correction for the very first time stamps
  #new_data <- na.locf(new_data, fromLast=TRUE, na.rm=FALSE)
  colnames(new_data) <- c("Seconds", "Midprice")

  # Define a delta grid
  delta_grid <- seq(1, 601, by=1)
  rv_that_day <- c()
  for(delta in delta_grid){
    data_delta <- new_data %>%
      slice(which((row_number() - 1) %% delta == 0))
    rv_delta <- sum(diff(log(data_delta$Midprice), lag=1)^2)
    rv_that_day <- c(rv_that_day, rv_delta)
  }

  rv_list <- cbind(rv_list, rv_that_day)
}

arv_delta <- rowMeans(rv_list)
arv_5mn <- arv_delta[length(arv_delta)%/%2]
drv <- (arv_delta - arv_5mn)/arv_5mn

ggplot() +
  geom_line(aes(x=delta_grid/60, y=drv*100), colour="red") +
  coord_cartesian(ylim=c(min(drv*100), 40)) +
  xlab(expression(Delta ~ "(min)")) + ylab(expression(DRV(Delta) ~ "(%)"))
```