Portfolio management and performance measures

Master 2: Financial Risk Management

Aix-Marseille Université 2021/20222 *Anthony Makarewicz*

Summary :

1) Introduction

This project aims at studying portfolio characteristics of 3 ETFs and to compare them with a suitable benchmark. Additionally, it expands the standard risk-adjusted performance measures to introduce alternative ones.

For that purpose, all further calculations are going to be realized in Rstudio that provide efficient packages for portfolio analysis. Some calculations are going to be done manually so as to break down the components of risk-adjusted performance measures such as the standard and extended Sharpe ratios, but some of them are already available in packages.

Nevertheless, sometimes we will need “black-boxes” functions such as the one used to fit a multivariate GARCH model that performs cumbersome computations and provide us only with the optimization’s results.

During the rest of this paper, we are going to go through the blocks of codes that require some explanation and present the output of our codes. Furthermore, we are going to show all the plots and comment them on.

2) ETF description

All our ETFs are available in Yahoo Finance, and we propose to describe some of their main important characteristics.

2.1) ETF IXC

The first ETF with the ticker symbol IXC is called iShares Global Energy ETF and seeks to track the investment results of an index composed of global equities in the energy sector. Its inception date is November 12, 2001 and is listed on the NYSEArca in US dollars. Moreover, it is issued by Blackrock financial management ETF list, and the brand is iShares, which is the most popular ETF’s brand in Europe. The main features can be seen below and originate from the yahoo finance summary page:

Previous Close	38.17	Net Assets	2.25B
Open	37.81	NAV	38.15
Bid	0.00 x 1000	PE Ratio (TTM)	3.54
Ask	0.00 x 1300	Yield	3.45%
Day's Range	37.32 - 38.02	YTD Daily Total Return	45.93%
52 Week Range	25.81 - 42.98	Beta (5Y Monthly)	1.30
Volume	612,253	Expense Ratio (net)	0.40%
Avg. Volume	774,979	Inception Date	2001-11-12

2.2) ETF IDV

The second ETF with ticker symbol IDV is named iShares International Select Dividend ETF and seeks to track the investment results of an index composed of relatively high dividend paying equities in non-U.S. developed markets. This one is listed on the BATS global markets in US dollars since June 11, 2007. Its main characteristics can be displayed below:

Previous Close	27.33	Net Assets	4.52B
Open	27.26	NAV	27.32
Bid	0.00 x 3100	PE Ratio (TTM)	N/A
Ask	0.00 x 900	Yield	7.07%
Day's Range	26.75 - 27.26	YTD Daily Total Return	-5.87%
52 Week Range	22.01 - 33.61	Beta (5Y Monthly)	1.20
Volume	1,142,752	Expense Ratio (net)	0.49%
Avg. Volume	1,374,858	Inception Date	2007-06-11

2.3) ETF SHY

The last ETF with ticker symbol SHY is called iSHares 1-3 Years Bond ETF, and since August 1, 2001, it has been trying to replicate the financial results of an index composed of US treasury

bills, with a maturity between 1 and 3 years, and is listed on the NasdaqGM in US dollars. We can find all the financial indicators in the table below:

Previous Close	**81.34**	Net Assets	**28.71B**
Open	**81.39**	NAV	**81.51**
Bid	**0.00 x 36200**	PE Ratio (TTM)	**3,697.73**
Ask	**0.00 x 900**	Yield	**0.92%**
Day's Range	**81.31 - 81.40**	YTD Daily Total Return	**-3.71%**
52 Week Range	**80.56 - 85.67**	Beta (5Y Monthly)	**0.22**
Volume	**6,163,276**	Expense Ratio (net)	**0.15%**
Avg. Volume	**6,359,203**	Inception Date	**2002-07-22**

From now on, we shorten the name of these ETFs by their ticker symbol in order to ease their description and we can move on to the data preparation section.

3) Data preparation

The data preparation step is the most important preliminary step in the data science world. Indeed, it consists in importing all the packages that contain optimized functions that perform all the computations for us and provide us with the most important results. Once we have imported the packages that we will be using, we need to import our dataset either directly from the internet or from a file from the disk. Afterwards, we must ensure that our data has been well imported and for that we perform a simple data exploration. However, the last step must not to be confounded with exploratory data analysis (EDA) that aims to gain insightful knowledge of the data structure and relationships between variables which we will achieve latter. Finally, we must pre-process our dataset in order to have the best quality as possible to not compromise the further analysis.

3.1) Data importation

First of all, we must first import the packages containing the functions that we will be using as well as our data from Yahoo Finance into weekly data. The quantmod package enables to directly import the data from Yahoo Finance without needing to save the data to disk, which is a good practice for reproducibility and update the data automatically. Thus, we download the weekly closing price for each ETF and store them in a matrix and merge them using an inner join, namely considering all common observations.

Therefore, we write the following lines of code:

```
# ****************
# Data Importation
# ****************

# load the packages
library(quantmod)
library(xts)
library(PerformanceAnalytics)


# specify the ticker symbols
tickers <- c("IXC", "IDV", "SHY","^GSPC")

# ---IXC---: iShares Global Energy ETF
# ---IDV---: iShares International Select Dividend ETF
# ---SHY---: iShares 1-3 Year Treasury Bond ETF
# ---GSPC---: S&P 500 Index
```

```
# instantiate an empty matrix matrix to store all weekly prices
mat_cl_prices <- matrix()

# download each closing price and join it to the existing matrix
for(ticker in tickers) {
  # import OHLC data from yahoo finance for all available observations
  price <- getSymbols(Symbols = ticker, auto.assign = FALSE, src = "yahoo")
  # take the closing price
  cl_price <- Cl(price)
  # convert into weekly frequency
  week_cl_price <- to.period(cl_price, period = "weeks", k = 1)[, 4]
  # merge using all common observations (inner join) for each new downloaded price
  mat_cl_prices <- merge.xts(mat_cl_prices, week_cl_price, join = "inner")
}
```

3.2) Data exploration

Now that we have imported our data, we can proceed to a simple data exploration so as to verify the integrity of our data and get some insights.
The following lines of code check the first and last rows, the dimensions, the column names, the periodicity, as well as the structure of our matrix.

All these operations can be performed in the following lines of code:

```
> head(mat_cl_prices)
           mat_cl_prices cl_price.Close cl_price.Close.1 cl_price.Close.2 cl_price.Close.3
2007-06-22            NA       43.00000            50.23            80.01          1502.56
2007-06-29            NA       43.11000            50.00            80.16          1503.35
2007-07-06            NA       44.65333            51.30            79.76          1530.44
2007-07-13            NA       45.98333            52.03            79.92          1552.50
2007-07-20            NA       45.62000            52.34            80.19          1534.10
2007-07-27            NA       42.60000            48.81            80.55          1458.95
> tail(mat_cl_prices)
           mat_cl_prices cl_price.Close cl_price.Close.1 cl_price.Close.2 cl_price.Close.3
2022-11-11            NA          41.71            26.18            81.24          3992.93
2022-11-18            NA          41.07            26.13            81.07          3965.34
2022-11-25            NA          41.44            26.83            81.19          4026.12
2022-12-02            NA          40.93            27.52            81.41          4071.70
2022-12-09            NA          38.03            27.52            81.32          3934.38
2022-12-15            NA          37.86            26.88            81.35          3895.75
>
> # check dimensions: rows for obs and cols for ETFs
> dim(mat_cl_prices)
[1] 809   5
```

```
> # check column names
> colnames(mat_cl_prices)
[1] "mat_cl_prices"    "cl_price.Close"   "cl_price.Close.1" "cl_price.Close.2" "cl_price.Close.3"
>
> # check the frequency and the start/end dates
> periodicity(mat_cl_prices)
Weekly periodicity from 2007-06-22 to 2022-12-15
>
> # check the structure of the data
> str(mat_cl_prices)
An 'xts' object on 2007-06-22/2022-12-15 containing:
  Data: num [1:809, 1:5] NA NA NA NA NA NA NA NA NA NA ...
 - attr(*, "dimnames")=List of 2
  ..$ : NULL
  ..$ : chr [1:5] "mat_cl_prices" "cl_price.Close" "cl_price.Close.1" "cl_price.Close.2" ...
  Indexed by objects of class: [Date] TZ: UTC
  xts Attributes:
List of 2
 $ src    : chr "yahoo"
 $ updated: POSIXct[1:1], format: "2022-12-16 14:10:11"
```

From the output, we can point out that the matrix contains 809 weekly observations for all our ETFs from June 22, 2007, to December 15, 2022, so it fits our requirements, that is we must have at least 10 years of weekly data.
Furthermore, we can observe the first not available (NA) column which is consistent since initially the matrix is empty, therefore the first ETF has been merged with nothing. Finally, the column names are clearly not indicative, so we are going to address these two flaws in the next section.

3.3) Data preprocessing

The data preprocessing step is a very important as it will condition the results obtained. Indeed, according to the well-known motto "garbage in garbage out" if the data we use for our analysis is of poor quality it will propagate to our final analysis.
As such, based on the previous section, we can already address two flaws of our matrix, that is the first NA column and the not indicative column names.
For that, we can simply drop the first NA column and replace the column names by the ticker symbols.
However, for S&P 500, we have defined the ticker symbol as ^GSPC which is not a valid column name (the ^ operator is the symbol for the power function). Therefore, we can simply define it apart:

```
> # ******************
> # Data Pre-processing
> # ******************
>
> # drop first NA column
> mat_cl_prices$mat_cl_prices <- NULL
> # replace not indicative column names
> colnames(mat_cl_prices) <- c(tickers[1:3], "GSPC")
> # now check the first rows
> head(mat_cl_prices)
                IXC   IDV   SHY    GSPC
2007-06-22 43.00000 50.23 80.01 1502.56
2007-06-29 43.11000 50.00 80.16 1503.35
2007-07-06 44.65333 51.30 79.76 1530.44
2007-07-13 45.98333 52.03 79.92 1552.50
2007-07-20 45.62000 52.34 80.19 1534.10
2007-07-27 42.60000 48.81 80.55 1458.95
```

Now that we have clear column names and non-NA columns, we must ensure that we have no missing values at all in our all matrix so as to study the statistical properties.
For that purpose, we must check if in any column we have at least one missing value and summing these missing values. To accomplish this in R, the is.na () function returns a Boolean mask of the original matrix with true (which is interpreted by the computer as 1) if it is a missing value and false otherwise. If any missing value is reported, we can address them with standard imputation for time series using the known observations between 2 or more missing values such as the last observation carried forward pattern (locf), or linear or spline interpolations. Here we report no missing values so we can pursue our analysis.

One requirement about our ETFs is that they must not be too much correlated. For that, we can compute the correlation matrix and considering that above 80% the two ETFs are too much correlated. Indeed, if they were too much correlated, we would have not benefited from the diversification principle based on the correlation that allows to reduce the global variance of our portfolio.
For this reason, if two or more ETFs are too much correlated, we can simply choose another ticker symbol from the ticker vectors defined in page 8 and re-run our code until the correlation matrix calculation without needing to do additional steps.

Now we can display these lines of code and the correlation matrix output in the console:

```
> # check if any missing values to impute them, if no continue the analysis
> sum(is.na(mat_cl_prices))
[1] 0
>
> # we can inspect the correlation matrix of ETFs to see if they are not too much correlated
> cor(mat_cl_prices[, 1:3])
           IXC        IDV        SHY
IXC  1.0000000  0.7066161 -0.6278667
IDV  0.7066161  1.0000000 -0.3751839
SHY -0.6278667 -0.3751839  1.0000000
```

Now that we have a well-defined dataset, we can transform the closing prices in order to obtain the returns.

As closing prices are not scale-free, we cannot use them directly for a statistical analysis. Instead, we consider the returns which additionally possess interesting statistical properties. As a matter of fact, returns are essential for financial data because they possess important statistical properties that can be extracted by econometric models.

For that, two approaches can be considered for computing the returns.

The first one is called the simple returns that are very useful for constructing the return of a portfolio as the return of the portfolio is the weighted average of the assets' returns.

Secondly, the log or continuously compounded returns enable straightforward time series aggregation and are the ones preferred in financial econometrics.

However, the two are similar and in fact the log-return is the first order's Taylor expansion for the simple return so when the returns are close to 0 (which is most of the time true for high frequency data such as intraday or daily) we can simply consider the log-returns as there is no indication about which one to use (and no indication about constructing a portfolio).

The formula for the log-returns can obtained as follows:

$$R_t = \log P_t - \log P_{t-1} = \log\left(\frac{P_t}{P_{t-1}}\right)$$

When computing the log-returns directly in R, we must ensure that we drop the first NA value, as the return at time t requires the price at time t-1 so for the first observation there is no return:

```
> # log-return calculation
> mat_returns <- Return.calculate(mat_cl_prices, method = "log")
> # see teh first NA value
> head(mat_returns)
                 IXC          IDV          SHY         GSPC
2007-06-22        NA           NA           NA           NA
2007-06-29  0.002554896 -0.004589452  0.001873035  0.000525576
2007-07-06  0.035173917  0.025667727 -0.005002537  0.017859302
2007-07-13  0.029350097  0.014129723  0.002003959  0.014311298
2007-07-20 -0.007932832  0.005940441  0.003372734 -0.011922661
2007-07-27 -0.068491989 -0.069825667  0.004479303 -0.050226909
> # remove the first NA value as there is no prices before the first
> mat_returns <- mat_returns[-1]
> # display the first returns
> head(mat_returns)
                 IXC          IDV          SHY         GSPC
2007-06-29  0.002554896 -0.004589452  0.001873035  0.000525576
2007-07-06  0.035173917  0.025667727 -0.005002537  0.017859302
2007-07-13  0.029350097  0.014129723  0.002003959  0.014311298
2007-07-20 -0.007932832  0.005940441  0.003372734 -0.011922661
2007-07-27 -0.068491989 -0.069825667  0.004479303 -0.050226909
2007-08-03 -0.037469470 -0.014237360 -0.001366557 -0.017904906
```

Now, we can move on to the statistical analysis section.

4) Statistical analysis

Now that we have high quality data, we can start a statistical analysis and emphasize some statistical properties about the returns. Firstly, we are going to gain more insight about the returns by comparing their summary statistics along with their central moments. Finally, we are going to provide a hypothesis test to test the normality assumption.

4.1) Summary statistics and central moments

The fastest approach is to use summary statistics that we can find in most statistical programming languages with the summary function.
The function provides the most important statistical properties in a table but forget to incorporate useful empirical central moments such as the standard deviation, skewness, and excess kurtosis.

To that regard, we can display them and comment the most important ones:

```
> # summary statistics
> summary(mat_returns)
     Index                IXC                  IDV                 SHY                 GSPC          
 Min.   :2007-06-29   Min.   :-2.891e-01   Min.   :-0.2830374   Min.   :-1.049e-02   Min.   :-0.200838  
 1st Qu.:2011-05-18   1st Qu.:-1.803e-02   1st Qu.:-0.0145002   1st Qu.:-7.105e-04   1st Qu.:-0.010238  
 Median :2015-04-06   Median : 1.783e-03   Median : 0.0017982   Median : 1.176e-04   Median : 0.002798  
 Mean   :2015-04-06   Mean   :-8.384e-05   Mean   :-0.0006786   Mean   : 2.484e-05   Mean   : 0.001206  
 3rd Qu.:2019-02-23   3rd Qu.: 2.071e-02   3rd Qu.: 0.0165207   3rd Qu.: 8.280e-04   3rd Qu.: 0.015026  
 Max.   :2023-01-13   Max.   : 1.611e-01   Max.   : 0.1125634   Max.   : 8.199e-03   Max.   : 0.114237  
>
> # weekly mean and standard deviation
> week_mean <- colMeans(mat_returns)
> week_sd <- apply(mat_returns, 2, sd)
> print(week_mean)
          IXC           IDV           SHY          GSPC 
-8.384202e-05 -6.786234e-04  2.483704e-05  1.205538e-03 
> print(week_sd)
       IXC         IDV         SHY        GSPC 
0.039521600 0.032388769 0.001841118 0.026455089 
>
> # inter quartile range
> apply(mat_returns, 2, IQR)
       IXC         IDV         SHY        GSPC 
0.038743105 0.031020920 0.001538563 0.025264035 
```

The summary output displays several statistical properties.
First it shows central measures such as the mean and the second quartile (median) along with dispersion measures such as the minimum, maximum, the first quartile (Q1) and the third quartile (Q3). We propose to comment some of the results as the interpretations are the same.

We can thus say that in average, for an investment time horizon of one week, an investor could expect a return of – 0.015%, - 0.077%, 0.002%, and 0.11% for IXC, IDV, SHY, and GSPC

respectively. Thus, the highest profitable investment based on the expect return only is the S&P 500 index, and the worst is IXC. Additionally, these results highlight a stylized fact about financial returns calculated from a high frequency (intraday, daily, or weekly), that is the average return close is to 0. However, another popular central measure which is more robust in the presence of highly skewed distributions is the median. As a matter of fact, the median is not particularly affected by extreme deviations from the mean as it considers all the observations. Indeed, it sort them in ascending order, and choose the observation such as 50% of observations are below that number and the other 50% are above it. Here, we see that for the S&P 500 it is approximately more than 2 times higher than the mean which is a typical result in left-skewed distribution as the mean is more sensitive to extreme observations. Nonetheless, for the ETFS, the median is many times higher than the mean, highlighting potentially higher left tails.

Central measures provide a good statistical measure of the centralization of the distribution but forget to consider the deviation from the center. Accordingly, we need to consider dispersion measures which share the same pros and cons than the mean and median, namely the standard deviation and the interquartile range. The standard deviation is the most popular and is computed by taking the square root of the variance, namely the mean of all squared deviations from the mean which therefore penalize large deviations from the mean. Like the mean, the standard deviation is computed from a particular time horizon and is expressed in the same scale than the return, so here in percentage. In fact, we can argue that in average, on a weekly time horizon investment, an investor could expect that its return fluctuates around the mean by 3.86%, 3.24%, 0.1%, and 2.5% for IXC, IDV, SHY, and the S&P500 respectively. Accordingly, IXC is the riskiest as it has the highest standard deviation which negatively impact the investor's utility.

Secondly, we can consider the interquartile range which is an alternative to the standard deviation. It is nothing but the difference between the third and the first quartile and is therefore more robust to extreme observations (called outliers in statistics). Despite this measure is rarely used for financial returns as we care more about outliers, we can still compute it. Here, the IQRs are all slightly below the standard deviations, which fits what we could expect because the distributions are all left-skewed and fat-tailed.

Finally, even though they are no robust, the minimum and maximums are easy to interpret. Here, we see that IXC achieves the worst minimum return over the full sample, reaching a return of – 29.91% on its worst week, whereas SHY achieves the lowest one with only – 1% on its one. The maximum is also on the same order than the minimum, yet they appear lower in absolute value than the maximums, because financial assets exhibit more extreme negative returns than positive ones.

Then, we have the first as well as the third quartiles that are returns such as 25% and 75% of sorted returns are below the first and third respectively. Here the lowest first quartile is achieved by IXC, so we can say that 25% of returns are lower or equal than – 1.8%, or that

75% of returns are better than – 1.8%. Moreover, IXC also achieves the highest third quartile, so we can assert that 75% of returns are below 2% and that 25% of returns are better than 2%.

If we assume that the return distributions are normally distributed, we can construct confidence intervals at 95%, such as 95% of the time the true weekly returns would fall in that range. To that end, we first need to compute the 5th and the 95th quantile of the standard normal distribution but as it is symmetric, they are the same except their sign (the 5th quantile is - 1.64 and the 95th is 1.64). As a matter of fact, based on the results obtained above, we can affirm that for IXC there is 95% chance that the next weekly will fall between – 6.52% and 6.29%, whereas for SHY it will only be between – 0.2% and 0.3%.

```
> # standard normal 95th quantile
> qnorm_95 <- qnorm(p = 0.95, mean = 0, sd = 1)
>
> bounds <- c(week_mean - qnorm_95 * week_sd,
+                     week_mean + qnorm_95 * week_sd)
> bounds_lab <- c("lower_bound", "upper_bound")
>
> conf_interv_95 <- matrix(bounds, nrow = 2, ncol = 4,
+                          dimnames = list(bounds_lab, tickers), byrow = TRUE)
> print(conf_interv_95)
                IXC         IDV          SHY       ^GSPC
lower_bound -0.06509109 -0.05395341 -0.003003533 -0.04230921
upper_bound  0.06492340  0.05259616  0.003053207  0.04472029
```

When comparing investment decisions, we sometimes rather consider the annualized mean which is obtained by multiplying the higher frequency mean by the number of periods in a year, so in our case by 52 as there are about 52 trading weeks in a year. The annualized mean, that is the average return an investor could expect on an investment time horizon of 1 year is - 0.8%, - 3.95%, 0.1%, and 6.13% for IXC, IDV, SHY, and the S&P 500 respectively. Thus, we can apply the same principal for the standard deviation. Nevertheless, the factor that multiplies the weekly standard deviation is now the square root of the number of periods in a year so $\sqrt{52}$.
Therefore, we can argue that on an investment time horizon of 1 year, an investor could expect in average a deviation from the annualized mean of $28.5\%, 23.36\%, 1.03\%$, and 19.09% for IXC, IDV, SHY, and the S&P 500 respectively.

```
> # central moments:
> # annualized simple/arithmetic mean
> ann_mean <- week_mean * 52
> print(ann_mean)
         IXC          IDV          SHY         GSPC
-0.004359785 -0.035288418  0.001291526  0.062687953
> # weekly and annualized standard deviation
> ann_sd <- week_sd * sqrt(52)
> print(ann_sd)
       IXC        IDV        SHY       GSPC
0.28499431 0.23355874 0.01327649 0.19077036
```

The next statistical property is the asymmetry of the distributions, which is highlighted by the skewness. A positive skewness for the return distribution is characterized by a right-skewed distribution which highlight that large positives returns are more frequent than negative ones, which is rarely the case for financial returns. Instead, a negative skewness characterizes a left-skewed distribution highlighting that large negative returns are more frequent than positive ones. Here, we can argue that the return distributions are all skewed to the left, and the lowest and highest skewness are -1.45, and -0.44 that are achieved by IDV and SHY respectively. Furthermore, as extremes returns are more frequent than those implied by the normal distribution, the return distributions are all leptokurtic distribution. This is particularly the case for shorter horizons, then as the return horizon increases the kurtosis tends to decrease because the return distribution looks more than the normal distribution. On the one hand, the skewness affects positively the investor's utility (the higher the skewness, the higher the utility), whereas on the other hand the excess kurtosis affects it negatively (the lower the excess kurtosis, the higher the utility).

```
> # skewness
> skewness(mat_returns)
               IXC       IDV        SHY       GSPC
Skewness -1.077333 -1.453777 -0.4257869 -0.9395736
> # excess kurtosis
> kurtosis(mat_returns)
                     IXC      IDV      SHY     GSPC
Excess Kurtosis 8.532609 10.56815 4.996475 7.794144
```

As they are significantly different from 0, we can then perform a Jarque-Bera test.

4.2) Jarque-Bera test

The third and fourth central moments highlight the deviation from normality, as the normal distribution has a skewness of 0 (symmetric around the mean) and excess kurtosis of 0 (mesokurtic distribution).
Here, as they are different from 0, we can test if this result is statistically significant with a hypothesis test under the null hypothesis of normality using a Jarque-Bera test. This econometric test calculates a test statistic which is increasing in both skewness and excess kurtosis. Therefore, the higher the skewness and excess kurtosis, the less likely the null hypothesis is. The set of hypotheses along with formula can be obtained as follows:

$$H_0 : \mathbb{S}(R_t) = 0 \text{ and } \mathbb{K}(R_t) = 3$$
$$H_1 : \mathbb{S}(R_t) \neq 0 \text{ and } \mathbb{K}(R_t) \neq 3$$

$$JB = \frac{T}{6}\hat{S}_T^2 + \frac{T}{24}\left(\hat{K}_T - 3\right)^2$$

This test statistic is assumed to converge asymptotically (as the sample size goes to infinity) to a Chi-squared distribution with 2 degrees of freedom which correspond to the skewness and excess kurtosis.

From the result of this test, we are interested in the p-values which is probability of rejecting the null hypothesis whereas she is true.

```
> # check if skewness and kurtosis are statistically different from 0
> jb_tests <- apply(mat_returns, 2, jarque.bera.test)
>
> jb_tests$IXC

        Jarque Bera Test

data:  newX[, i]
X-squared = 2608.2, df = 2, p-value < 2.2e-16

> jb_tests$IDV

        Jarque Bera Test

data:  newX[, i]
X-squared = 4057.5, df = 2, p-value < 2.2e-16

> jb_tests$SHY

        Jarque Bera Test

data:  newX[, i]
X-squared = 892.85, df = 2, p-value < 2.2e-16

> jb_tests$GSPC

        Jarque Bera Test

data:  newX[, i]
X-squared = 2155.1, df = 2, p-value < 2.2e-16
```

Here, we report p-values less than 1% for all our ETFs and our benchmark, accordingly we have almost no chance of rejecting the null hypothesis whereas she is true, so we cannot accept it. Therefore, as the sample skewness and excess kurtosis are significant, a better modeling choice for returns' shocks would be a skewed student's distribution with two more parameters (d1 for the degree of fait-tails and d2 for the asymmetry) that we could estimate even though cumbersome with the Maximum Likelihood. Finally, we can add that for all our returns' ETF the closest to the normal distribution is SHY, as it has the lowest JB test statistics, meaning that is has the lowest skewness and excess kurtosis.

Now we can move on to the data visualization section that provides a useful visual diagnostic about the returns.

5) Data visualization

The previous section has allowed to provide insightful statistical information about the returns that can be confirmed with a visual representation. First, we are going to plot the normalized time-series prices in the same figure in order to compare our investments.

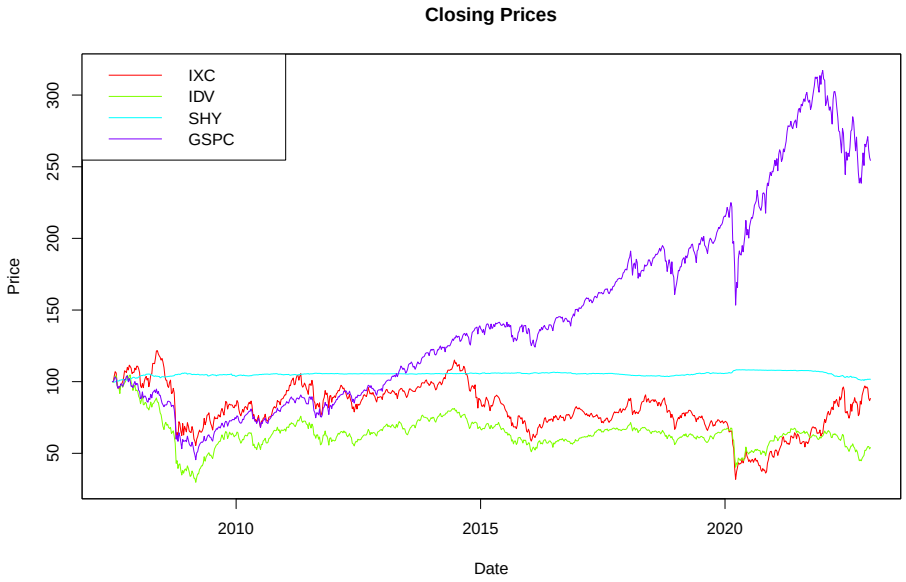
Afterwards, we are going to focus on the return distribution using 3 different approaches, namely the histogram, the boxplot, and the Q-Q plot. Ultimately, we will plot the autocorrelation function (ACF) and the partial autocorrelation function (PACF) so as to identify potential patterns.

5.1) Normalized overlayed ETFs

The first plot to consider for time-series is a line plot of the closing price that describes the evolution of the investment over time. Here, we are interested in displaying the ETFs and the benchmark, but as prices are not scale free, we cannot achieve this with more than 2 prices (with 2 time series we could have used two different y-axes on the same figure). Therefore, one preliminary step is to normalize our ETFs so that they start at the same value, namely $100. For that purpose, for each ETF (column) we must divide each price by the first one and multiply this ratio by 100. Thus, for the first normalized closing price, we divide the first price by itself and multiply by 100, so we obtain 100 accordingly.

We perform those operations in the following lines of code, and we can display the overlayed plot:

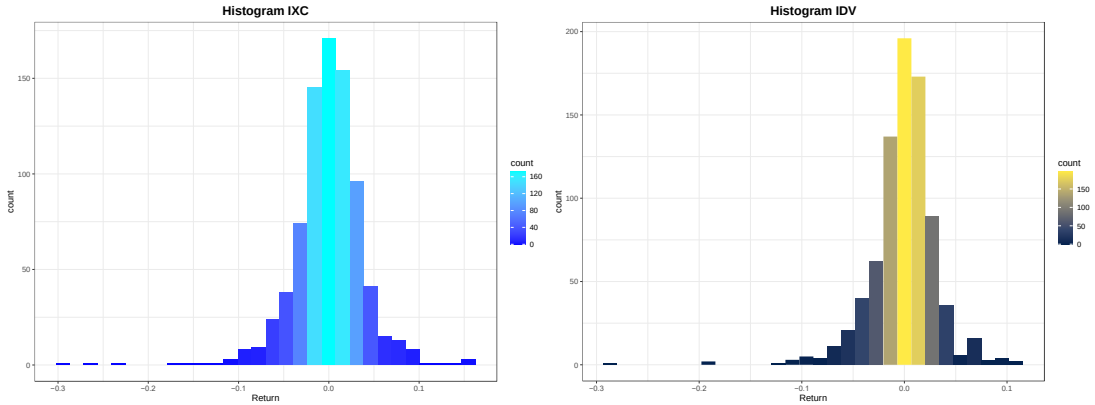
```
# ******************
# Data Visualization
# ******************

# package for data visualization
library(ggplot2)


# take the first row for normalization
first_row <- coredata(mat_cl_prices[1, ])
# normalize at $100
mat_cl_prices_norm <- as.xts(t(apply(mat_cl_prices, 1, function(x) (x / first_row) * 100)))
# choose appropriate column names
colnames(mat_cl_prices_norm) <- tickers
colnames(mat_cl_prices_norm)[4] <- "GSPC"
```

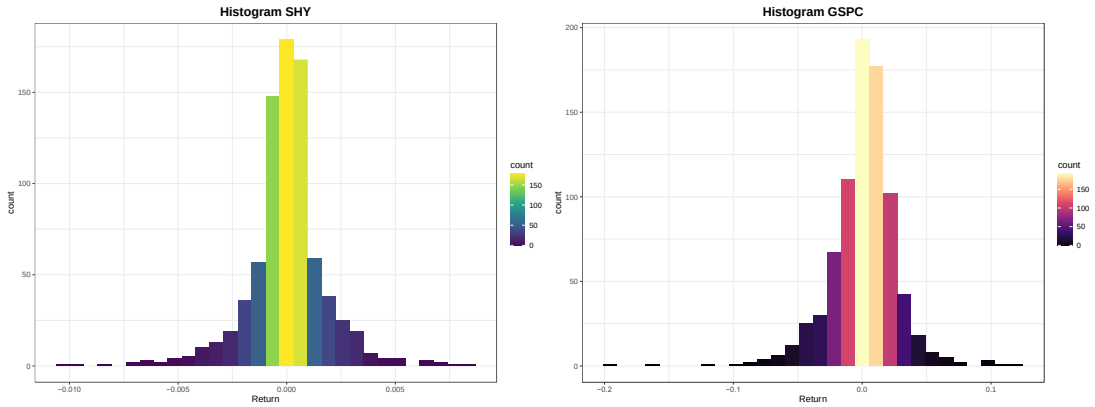


From this chart, we can see clearly that the S&P 500 outperforms our ETFs. Furthermore, we can notice the steady evolution of the ETF SHY which we saw earlier has the lowest standard deviation, skewness, and excess kurtosis so it is the less risky. The other ETFs, namely, IXC and IDV seems to be still underwater since June 22, 2007, that is an investor that has invested since June 22, 2007, and that have conserved is position, is currently losing money.

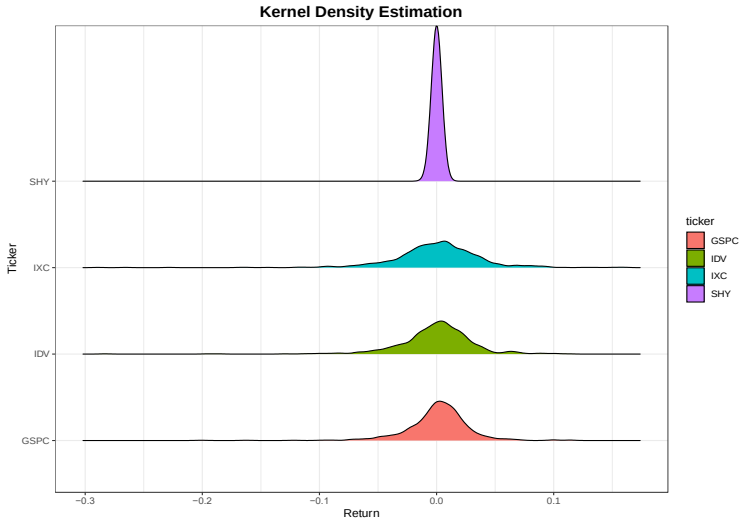
5.2) Histograms and boxplots

To compare the return distributions, we have at our disposal several graphical tools.
First, we can consider the most popular one, namely the histogram.
The histogram is a particular kind of plot that counts the number of observations in each bin of equal size, with each bin corresponding to a small range of the value of the variable to be explored (here the weekly return). As we have a lot of observations (802) we can display the histograms with 30 bins so as to describe the distributions more accurately:





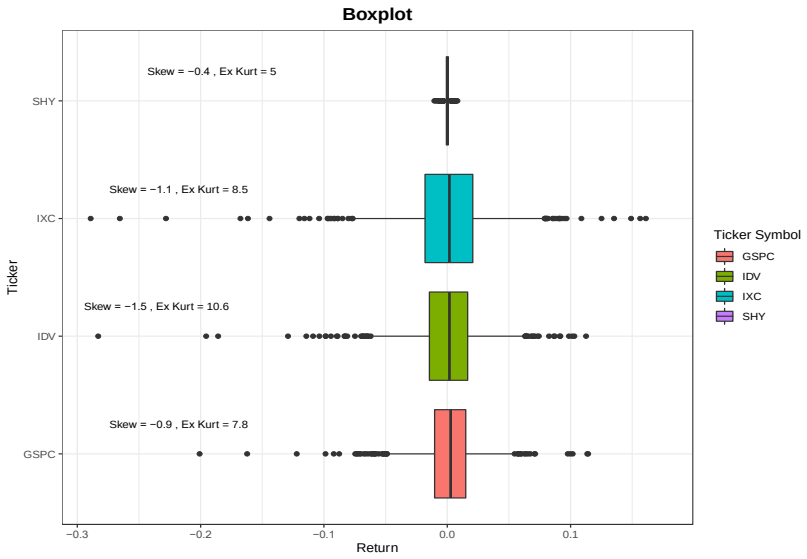
The return distributions for the 4 assets are all centered at 0 and they resemble the normal distribution in the center, but not in the tails as we observe an asymmetry and high tails. Additionally, we can display the kernel density estimate of the histogram in the same figure so as to highlight the main differences between the return distribution:



Here, by plotting the distribution on the x-axis scale, we see that the peak around 0 for SHY is more pronounced because the standard deviation is significantly lower than the other distributions.

Secondly, one good alternative to the histogram is the boxplot. Boxplots are useful visualization tools that allow us to easily highlight the quartiles as well as the outliers. The bottom of the box represents the first quartile (Q1), the band in the middle the median and the upper bound of the box the third quartile (Q3). The points correspond to outliers, and they are spotted outside the whiskers (the horizontal bars outside the box) which are computed as $+ 1.5 \times IQR$ for the top whisker and $- 1.5 \times IQR$ for the bottom where IQR denotes the Inter Quartile Range (Q3 – Q1).

Based on statistical properties described in page 18, we can bring some description to these boxplots:

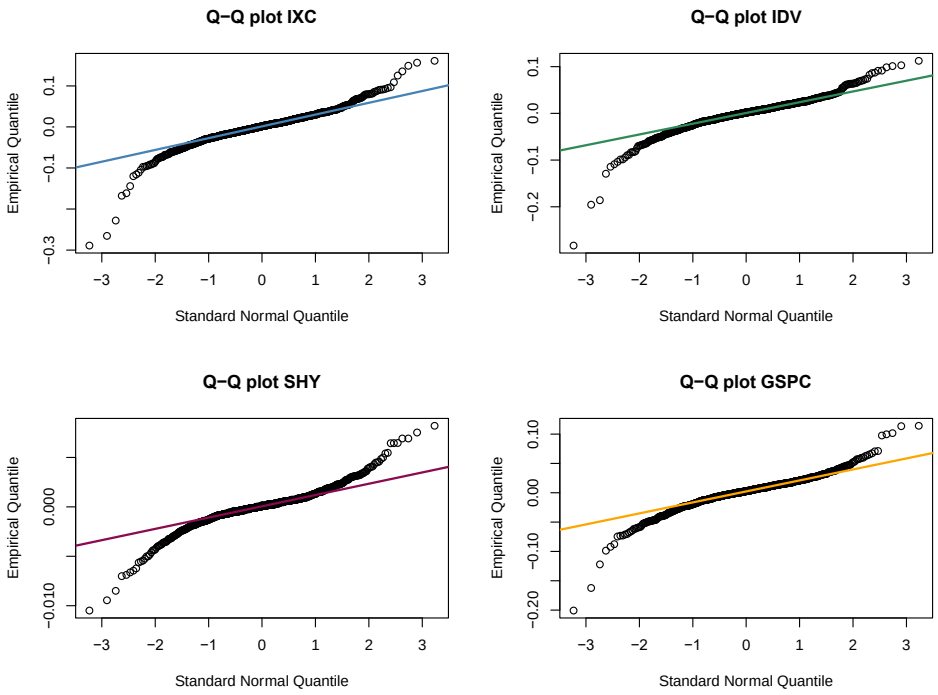


The ETFs IXC, IDV and the S&P 500 display a high negative skewness as we observe more outliers on the bottom of the whisker (here the left as they are stacked horizontally) rather than above. However, as there is almost no variation for SHY the x-axis scale is different from the others, so we do not see any information from this plot even though it as a skewness of -0.4 and excess kurtosis of 5. Additionally, the kurtosis property is described by the presence of excessive outliers which are easy to spot for IXC, IDV and the S&P 500.

Nevertheless, based on the boxplots alone, we cannot assert that the extreme deviations come from a leptokurtic distribution, as we could have simulated a large sample of normal random variables and have many outliers.

5.3) Quantile-Quantile plots

In this regard, a quantile-to-quantile plot allows to remedy the flaws illustrated from the boxplots so as to compare each empirical quantile to a theoretical one, such as the standard normal distribution. Now we can say that large deviations from the tails of the line highlight the deviation from normality.
If there is systematic deviation from the 45-degree line, then a statistical normality check can be performed to see if this result is statistically significant.
However, we saw previously that none of our ETFs and the S&P 500 follow a gaussian distribution which is highlighted in the Q-Q plots below.

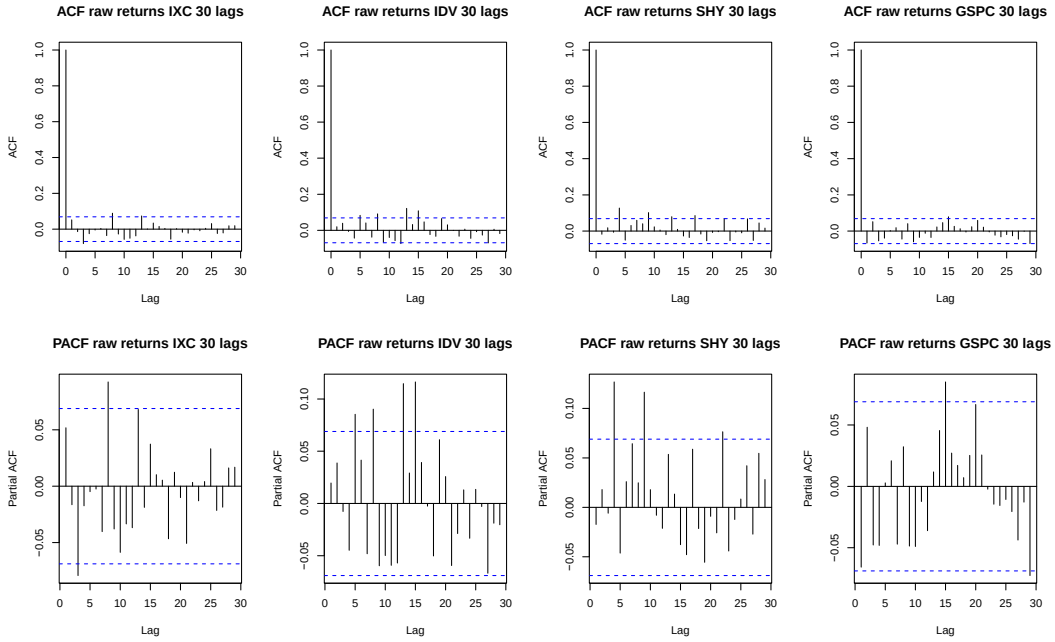


Here, we can see the inverse S-shaped that highlighting a negative skewness and excess kurtosis larger than the normal distribution.

5.4) Partial and simple Autocorrelation functions

The simple and partial autocorrelation functions (ACF/PACF) can be used to detect potential autocorrelations. However, an important stylized fact about financial returns is the lack of autocorrelation, particularly for small horizons such as daily, so on the surface a similar result should be expected.

The simple autocorrelation function is based on the linear correlation coefficient measured between today's date and a past date. The function is usually used with a statistical test to test if the autocorrelations are all equal zero. Graphically, horizontal dashed lines are generally represented in order to conclude or not to a lack of autocorrelation. If a value is taken out of this interval, then we reject the null hypothesis of the absence of autocorrelation for that particular lag. Therefore, we would conclude that there is a presence of autocorrelation that can be subsequently extracted by a statistical model.
The partial autocorrelation function, meanwhile, follows the same logic, but neutralizes the intermediate correlations occurring between two dates. These functions can therefore be represented for different lags from 1 to 30. Thus, we can display the ACF and PACF for the raw returns:



Nevertheless, the fact that we reject the absence of autocorrelation based on one lag is not a robust approach as usually we consider many lags up to 30.
Therefore, it would be preferable to test if all the autocorrelation from 1 to a particular lag are all equal to 0. To this end, we can perform a Ljung-Box test with the set of hypotheses associated to this test along and the test statistics which are:

$$H_0 : \rho_1 = \rho_2 = \cdots = \rho_k = 0$$
$$H_1 : \rho_j \neq 0 \text{ for some } j \in \{1, \ldots, k\}$$
$$Q_{LB}(k) = T(T+2) \sum_{j=1}^{k} \frac{\hat{\rho}_j^2}{T-j}$$

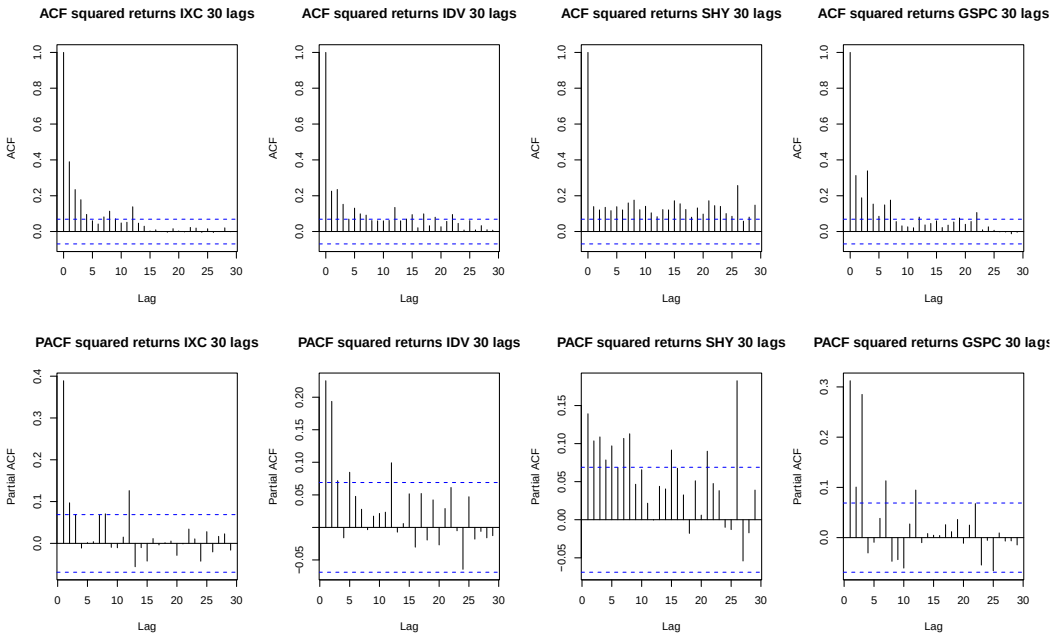
We can thus perform such as test with 30 lags for each of our ETFs and the S&P 500 index:

```
> # Ljung-Box test with 30 lags
> lj_box <- apply(mat_returns, 2, Box.test, lag = 30, type = "Ljung-Box")
> lj_box$IXC

        Box-Ljung test

data:  newX[, i]
X-squared = 33.812, df = 30, p-value = 0.2884

> lj_box$IDV

        Box-Ljung test

data:  newX[, i]
X-squared = 69.166, df = 30, p-value = 6.273e-05

> lj_box$SHY

        Box-Ljung test

data:  newX[, i]
X-squared = 61.224, df = 30, p-value = 0.0006528

> lj_box$GSPC

        Box-Ljung test

data:  newX[, i]
X-squared = 37.728, df = 30, p-value = 0.1569
```

By looking at the p-values, we cannot accept the null hypothesis for the ETFs IDV and SHY, yet we cannot reject it for IXC and GSPC as their p-values are larger than any confidence level, namely 1, 5 or 10%.

Lastly, we know that one stylized fact about financial returns is that of course there is usually no pattern in the ACF and PACF of the raw returns, however for the squared returns there is usually some persistence.

Consequently, we can display the correlograms of squared returns:



Here, we see that the ACFs are significant for all our ETFs and that SHY is the one with the most pattern in its squared return, as we could use squared returns from 30 weeks ago to forecast the next weekly squared return. These correlograms highlight the use of volatility models such as GARCH models that could enable to forecast the return volatility for many periods ahead.

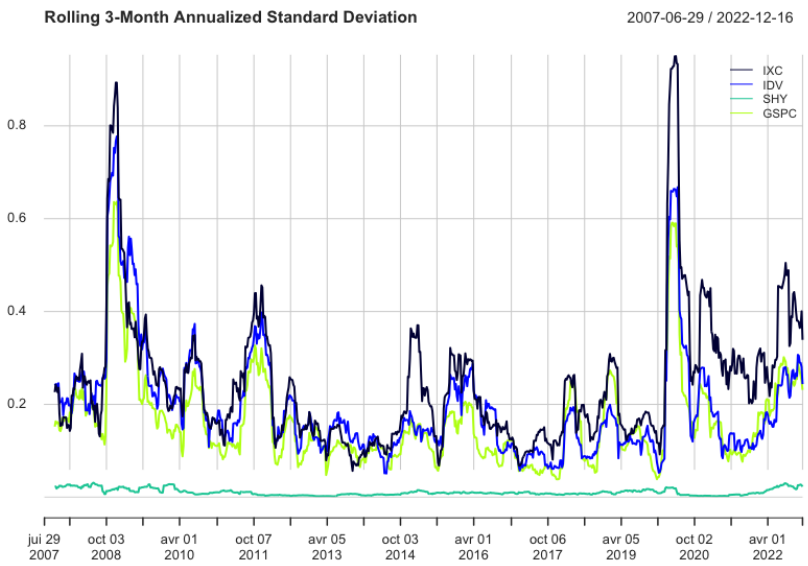
6) Data modelling

A good data modeling approach for financial returns would leverage stylized facts, namely the high correlation with a benchmark as well as the high persistence in the variance proxied by the expected value of squared returns. Indeed, as the weekly expected return is close to 0, most financial econometric approaches just omit it in the modeling process.

6.1) Jensen equation with GARCH errors

In the Jensen equation, we regress the return of a given stock or an ETF onto the return on a benchmark which is assumed to explained most of the variation. Here, we consider that each individual ETF gives its own model and the benchmark for the market return is the S&P 500 index. However, as we have seen in the previous section the variance that can be proxied by the sum of squared returns highlight a high persistence in the series.

We can illustrate this with a 3-month rolling window that computes the annualized standard deviation using only 3 months of data and adding the new one as well as discarding the last one:



Based on the rolling window estimates, we can notice the evolution of the annualized standard deviation, highlighting the non-stationarity of that one (but not for SHY). During periods of market stress when the volatility shoots up, the returns tend to decrease sharply, this is called the leverage effect.
Besides, the annualized standard deviation demonstrates 3 important features: the time variation, the persistence, and the mean reversion aspect.
Firstly, for the time variation aspect, we can unequivocally see that it is not a straight line, as there are periods when the volatility is many times higher than its long-term value (particularly during the busts in 2008 and 2020).
Secondly, the persistence is explained by volatility clusters, that is periods of low and high volatility tend to be followed by periods of low and high volatility respectively.
Lastly, we can see from this chart that when the volatility tends to be higher than its long run value of approximately 20% for IXC, IDV and the S&P 500, it is expected to return to this one.

According to these 3 features, it seems reasonable to consider using a simple econometric model such as GARCH (1,1) for the volatility that allow to capture most of the characteristics of the time-varying volatility. Instead, we could use more sophisticated models such as GJR-GARCH to model the leverage effect and/or skewed student's t distribution for the underlying distribution of the shocks.

6.1.1) Model fitting

To that end, we can use the rugarch package that enable to quickly estimate such a model by specifying the model's characteristics we want to use. A standard GARCH (1,1) model with constant mean (which gives the alpha coefficient) and normally distributed shocks is specified for each ETF. Additionally, we consider as external regressor the S&P 500 for the mean model.

The optimization performed under the hood by rugarch is a MLE routine which is an iterative search for calculating the conditional probability of observing the data sample given a probability distribution and distribution parameters.

Here, we do not have to bother about the implementation details, and we can just fit the model to the data in order to extract useful characteristics that we are going to comment on in the next section:

```
# -------------
# model fitting
# -------------

# specify the model (constant mean model with S&P 500 as external regressor)
general_garchspec <- ugarchspec(mean.model = list(armaOrder = c(0,0),
                                                  external.regressors = mat_returns$GSPC),
                          # simple GARCH(1,1) model for the variance model
                          variance.model = list(model = "sGARCH", garchOrder = c(1,1)),
                          # normal distribution for the GARCH shocks
                          distribution.model = "norm")

# fit the model to the data for each ETF
garchfit_ixc <- ugarchfit(spec = general_garchspec, data = mat_returns$IXC)
garchfit_idv <- ugarchfit(spec = general_garchspec, data = mat_returns$IDV)
garchfit_shy <- ugarchfit(spec = general_garchspec, data = mat_returns$SHY)
```

6.2.2) Feature extraction

From the fitted models obtained in the previous step, we can extract some insightful features about the optimization results.
From an econometrician point of view, the first thing that we care about is if the estimated coefficients are statistically significant from 0. Indeed, if they were not, we would retrieve them as it increases the complexity of our model without providing enough predictive capacity.
We can thus display the summary of the model coefficients as well as the p-values:

```
> show(garchfit_ixc)

*---------------------------------*
*          GARCH Model Fit        *
*---------------------------------*

Conditional Variance Dynamics
-----------------------------------
GARCH Model     : sGARCH(1,1)
Mean Model      : ARFIMA(0,0,0)
Distribution    : norm

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu     -0.002019    0.000649  -3.1090 0.001877
mxreg1  1.156328    0.032895  35.1519 0.000000
omega   0.000007    0.000005   1.5542 0.120142
alpha1  0.113121    0.021089   5.3639 0.000000
beta1   0.885576    0.020513  43.1705 0.000000


> show(garchfit_idv)

*---------------------------------*
*          GARCH Model Fit        *
*---------------------------------*

Conditional Variance Dynamics
-----------------------------------
GARCH Model     : sGARCH(1,1)
Mean Model      : ARFIMA(0,0,0)
Distribution    : norm

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu     -0.002091    0.000529  -3.9507  7.8e-05
mxreg1  1.046279    0.024508  42.6908  0.0e+00
omega   0.000009    0.000001   7.8278  0.0e+00
alpha1  0.077564    0.008056   9.6280  0.0e+00
beta1   0.891963    0.014399  61.9442  0.0e+00


> show(garchfit_shy)

*---------------------------------*
*          GARCH Model Fit        *
*---------------------------------*

Conditional Variance Dynamics
-----------------------------------
GARCH Model     : sGARCH(1,1)
Mean Model      : ARFIMA(0,0,0)
Distribution    : norm

Optimal Parameters
------------------------------------
        Estimate  Std. Error   t value Pr(>|t|)
mu     -0.000035    0.000037 -0.942317  0.34603
mxreg1 -0.012396    0.001873 -6.619682  0.00000
omega   0.000000    0.000000  0.035127  0.97198
alpha1  0.090467    0.014622  6.187274  0.00000
beta1   0.903202    0.013581 66.502860  0.00000
```

From these results, we can argue that the alpha given by the mu coefficients, namely the excess return above its theoretical price from the Capital Asset Pricing Model (CAPM), is -0.00219 for IXC and IDV, and – 0.000035 for SHY. Therefore, we can claim that our ETFs do

not generate alpha, that is they do not outperform their theoretical price, but SHY has the highest one. Moreover, the alphas are statistically significant for IXC and IDV as their p-values are less than 5%, yet SHY's alpha is not statistically significant as its p-value is 30%. Consequently, we could consider instead the standardized alpha which is given directly by the t-statistic, but even in this case SHY still has the highest one (- 0.94 compared to – 3.10 and – 3.95 for IXC and IDV respectively).

Additionally, the beta, namely the sensitivity of our ETFs to the market return is 1.15, 1.05, and – 0.01 for the ETFs IXC, IDV, and SHY respectively. Correspondingly, we can assert that when the market return increases by 1%, the return for IXC, IDV, and SHY also increases by 1.15%, 1.05, and – 0.01% respectively.
Thus, the higher the beta, the riskier our ETF is, so the riskiest ETF in our case would be IXC which amplifies movements in the market return by 0.15%.

To assess the performance of the GARCH model, we can display the persistence of the GARCH parameters that underscore whether the GARCH model is mean reverting. As a matter of fact, the key benefit of using a GARCH model is its mean reversion aspect which is a key feature in financial time series. For instance, the Vascisek model is widely known for modeling mean reverting interest rates, and conditional covariances for dynamic conditional covariance models. The mean reversion allows the conditional variance forecasts to converge to the unconditional one when forecasting large periods ahead. In this case, there is mean reversion when the persistence, namely the sum of the alpha and beta parameter/s, is strictly less than one which is the case for our ETFs as displayed below:

```
> # persistences (alpha + beta GARCH parameters)
> persistence(garchfit_ixc)
[1] 0.998055
> persistence(garchfit_idv)
[1] 0.9692379
> persistence(garchfit_shy)
[1] 0.9952289
>
> # unconditional variances
> uncvariance(garchfit_ixc)
[1] 0.003897951
> uncvariance(garchfit_idv)
[1] 0.000297144
> uncvariance(garchfit_shy)
[1] 3.305369e-06
```

Finally, a performance metric used for assessing the performance of linear models is the R-squared (R2), but the package rugarch does not provide such a metric. Instead, the Maximum Likelihood provide metrics such as information criterias that enable to consider both the goodness of fit and the complexity of the model. Additionally, the log-likelihood which is the sum of the log-likelihoods for each of our samples allows to assess the fitting but is not scale free and depends on the sample size.

Nevertheless, we know from the R-squared formula that we can compute it manually given 2 elements, the sum of squared of residuals and the total sum of squares.

Therefore, we can display the code the previous model pieces of information described above along with the output:

```
> # likelihoods
> likelihood(garchfit_ixc)
[1] 1943.4
> likelihood(garchfit_idv)
[1] 2201.983
> likelihood(garchfit_shy)
[1] 4181.217
>
> # residuals
> resid_ixc <- residuals(garchfit_ixc)
> resid_idv <- residuals(garchfit_idv)
> resid_shy <- residuals(garchfit_shy)
>
> # R2 intermediate calculations:
> # sum of residuals squared
> sum_squared_resid_ixc <- sum(resid_ixc^2)
> sum_squared_resid_idv <- sum(resid_idv^2)
> sum_squared_resid_shy <- sum(resid_shy^2)
> # total sum of squares
> total_sum_squares_ixc <- sum((mat_returns$IXC - mean(mat_returns$IXC))^2)
> total_sum_squares_idv <- sum((mat_returns$IDV - mean(mat_returns$IDV))^2)
> total_sum_squares_shy <- sum((mat_returns$SHY - mean(mat_returns$SHY))^2)
>
> # R2 final calculation
> R_squared_ixc <- 1 - (sum_squared_resid_ixc / total_sum_squares_ixc)
> R_squared_idv <- 1 - (sum_squared_resid_idv / total_sum_squares_idv)
> R_squared_shy <- 1 - (sum_squared_resid_shy / total_sum_squares_shy)
>
> print(R_squared_ixc)
[1] 0.5587421
> print(R_squared_idv)
[1] 0.7176218
> print(R_squared_shy)
[1] 0.05310953
```

Here we can point out that model that fits the data the best is the one for SHY as it has the highest log-likelihood. Indeed, SHY is much less complex to model because its return distribution is closer than the gaussian distribution which is the key assumption in MLE.

However, for the R-squared we obtain a completely different story. Indeed, SHY has now a R2 of only 5% compared to IXC, IDV which are 55%, 72% respectively. Therefore, we can assert that the model explains 55%, 72%, and 5% of the return's variations for IXC, IDV and SHY. Furthermore, the higher the R2 the better, so the ETF accounting for most of the variations is IDV. Lastly, one thing to notice is that if we had a different number of parameters for each ETF's returns, we would have computed the adjusted R-squared that take into account the goodness of fit as well as the complexity so as to obtain the best parsimonious model.

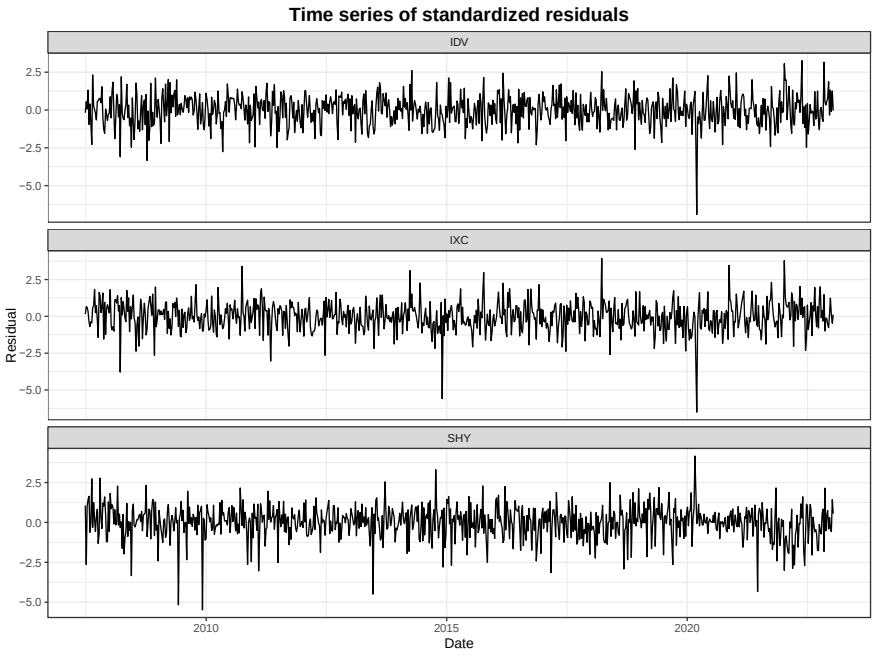
6.1.3) Residuals diagnostic

Previously, we have obtained metrics so as to emphasize the final performance of our model, however we do not know if we can improve it further.
Therefore, a residuals diagnostic allows to have an insight about which part of our model can be strengthened. Specifically, there are a few assumptions about the residuals that can be tested and that guarantee the validity of our model.

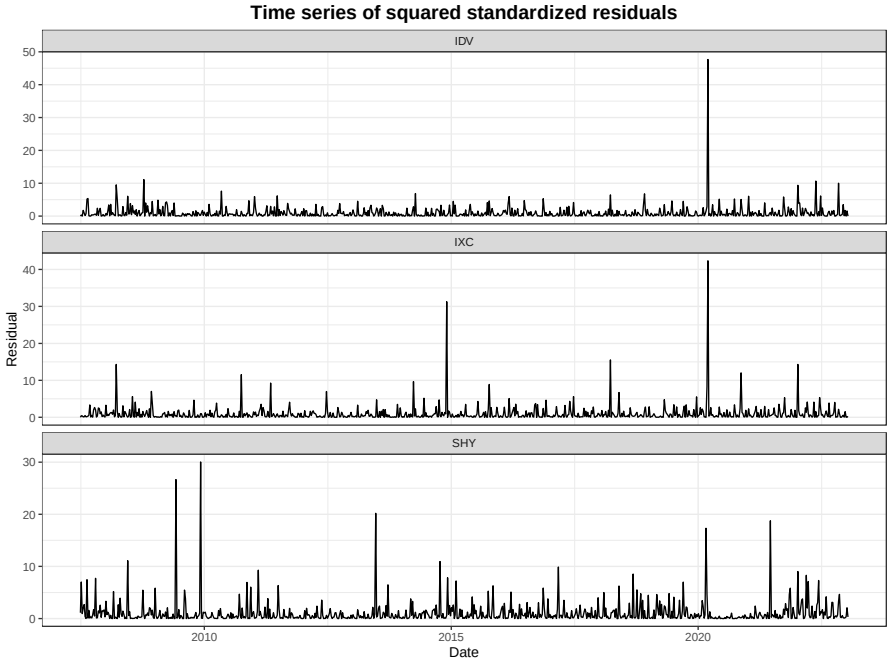
First, if the time series of residuals does not resemble a gaussian white noise process the model is probably wrong. For instance, it can be manifested by a clear trend (no stationarity of the mean) or no constant variations across the full time series (no stationarity of the variance). To that end, many tools can be used to address these shortcomings.
For the first, we would simply take the first difference of the series, namely subtracting each value by its predecessor, and see if the series is stationarity. If not, then we would re-iterate this procedure until we get a stationary process.
For the last, we would take power, inverse, or log transformations of the series depending on the non-constant variability.

Therefore, we can display a plot of residuals in order to emphasize our thought, but before that, we can standardize the series of residuals which is common practice, so that it has a 0 mean and a variance of 1:

```
# standardized residuals (deameaned and divided by their standard deviation)
sd_resid_ixc <- residuals(garchfit_ixc, standardize = TRUE)
sd_resid_idv <- residuals(garchfit_idv, standardize = TRUE)
sd_resid_shy <- residuals(garchfit_shy, standardize = TRUE)
```

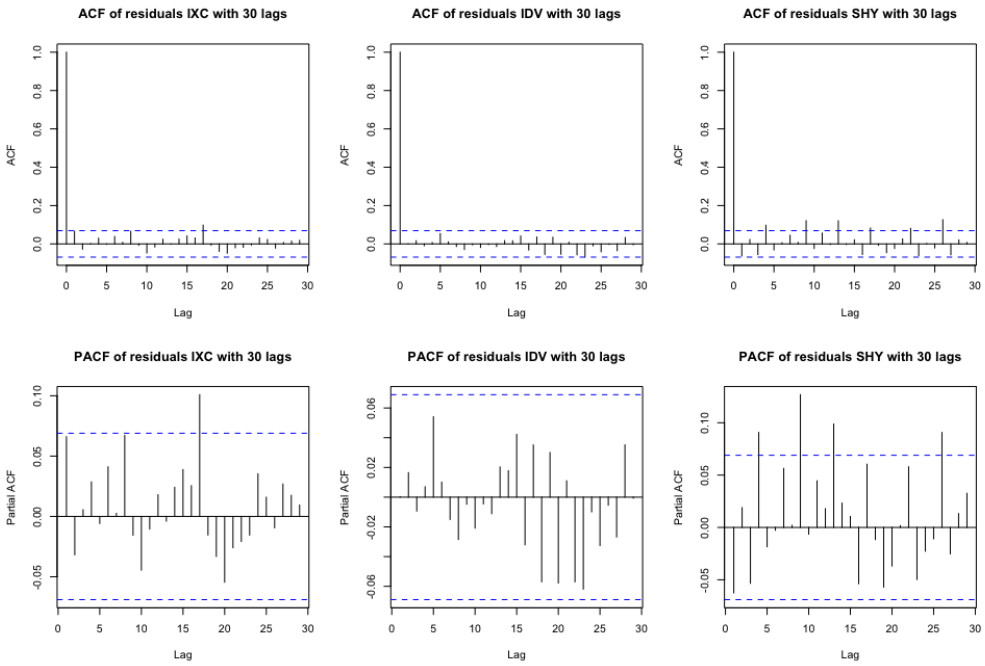


Here, we do not see a clear pattern, the unconditional mean is close 0, and we observe small variance clusters we can be seen more accurately by plotting the squared residuals which penalize higher values:



Here we see that for IDV except during ten covid-19 crash, the squared residuals are most of the time below 10 so it is stationary in variance. Therefore, the GARCH (1,1) model captures all the heteroskedasticity of the variance, so it is a good model. This observation is almost the same for IXC, but as the squared standardized residuals exceed the value of 10 many times. Nevertheless, SHY presents many clusters which highlight that the GARCH (1,1) model does not capture the correlation of residuals enough. For instance, from 2015 to 2020, the volatility is very high whereas from 2020 to 2021 the volatility is very low.

Secondly, we can focus on one important information for time series analysis, namely the ACF and PACF plots of the residuals. Indeed, if we find any strong pattern in the ACF and PACF, we can tap into this information so as to include AR and/or MA lags of returns in the mean model.



Here, we do not have a particular pattern in the residuals for IDV in both the ACF and PACF, which highlight that incorporating AR or MA lags would not improve the model, and even worse as it would increase the complexity of the model.
However, the ETF SHY indicate some pattern in both the ACF and PACF, which means that using AR and MA terms could potentially improve the predictive performance.
Here if we had to choose the perfect AR and/or MA lags, we would go from the lowest lags (1 in this case) and increasing the lags progressively to see if we obtain a lower information criterion (Akaike or Bayes are the most widely used).

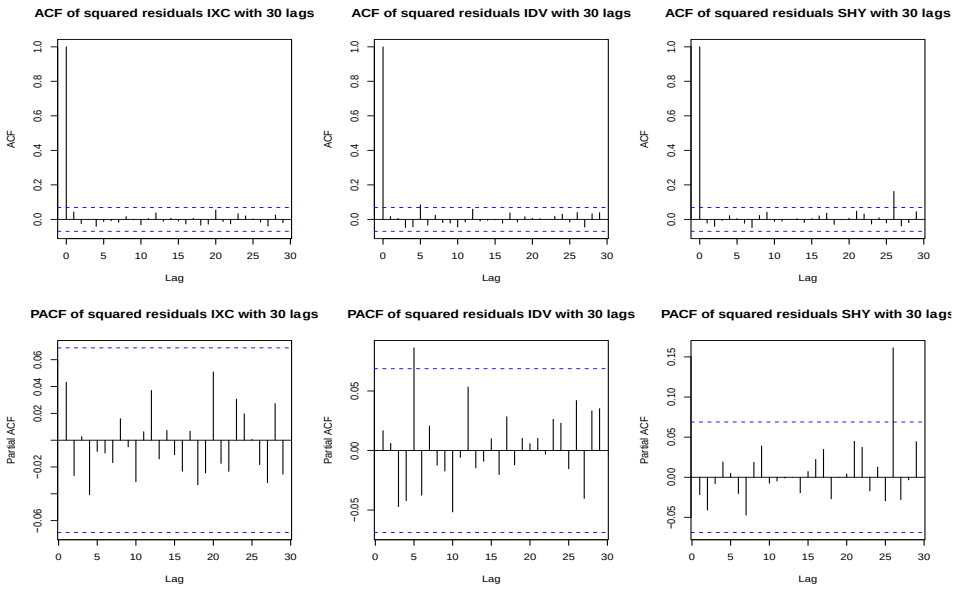
Finally, the ETF IXC does not have significant pattern in its correlograms, despite the 17th lag in both the ACF and PACF so maybe using the return 17 weeks ago could improve the predictive return for this week. Nevertheless, one lag may not be enough to include AR and MA lags, so a hypothesis test would be better for taking a decision.

Consequently, we can display the Ljung-Box statistics along with their p-values, and we reject the null hypothesis of all the autocorrelations from lag 1 to 30 to be jointly equal to 0 if the p-value is less than 5%.

```
> Box.test(sd_resid_ixc, lag = 30, type = "Ljung-Box")

        Box-Ljung test

data:  sd_resid_ixc
X-squared = 29.232, df = 30, p-value = 0.5054

> Box.test(sd_resid_idv, lag = 30, type = "Ljung-Box")

        Box-Ljung test

data:  sd_resid_idv
X-squared = 34.464, df = 30, p-value = 0.2627

> Box.test(sd_resid_shy, lag = 30, type = "Ljung-Box")

        Box-Ljung test

data:  sd_resid_shy
X-squared = 81.626, df = 30, p-value = 1.15e-06
```

Here the p-value for IXC is larger than all confidence levels, namely 10%, 5%, and 1%, so we cannot reject the null hypothesis of the absence of autocorrelation.
For IDV the p-value is half the one of IXC, but despite having 1 significant pattern in the 17th lag, we cannot reject the null hypothesis.
Ultimately, SHY has strong autocorrelation in its ACF with a p-value close to 0, so we have almost no chance of rejecting the null hypothesis whereas she is true.

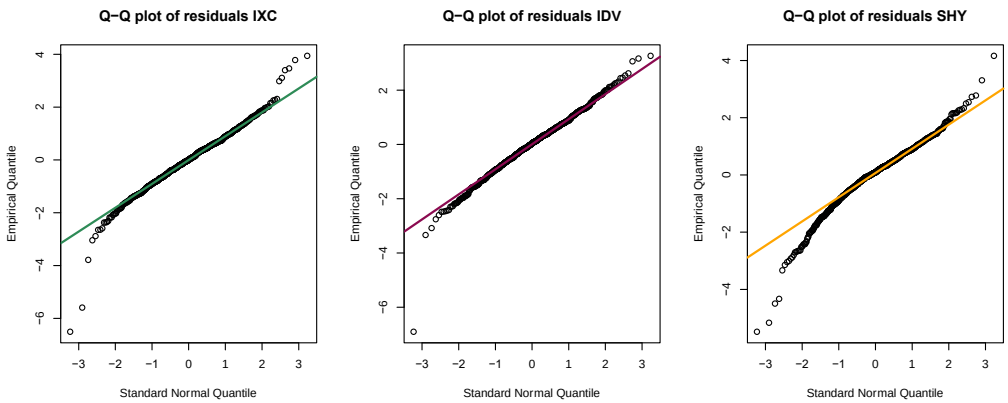
Furthermore, we can do the same with the squared standardized residuals to check if the variance model is correct. The diagnostic is the same, namely we want to asset that there is no autocorrelation in the squared residuals otherwise the variance model does not capture the persistence of volatility which is very high. Therefore, we can plot the correlograms along with the Ljung-Box test. Based on the previous observation of the time series of squared residuals, we should expect that IXC and IDV present almost no autocorrelation, whereas for SHY it should be the inverse.



```
> # statistical diagnostic
> Box.test(mat_sqrd_resid_merged[, "IXC"], lag = 30, type = "Ljung-Box")

        Box-Ljung test

data:  mat_sqrd_resid_merged[, "IXC"]
X-squared = 15.579, df = 30, p-value = 0.986

> Box.test(mat_sqrd_resid_merged[, "IDV"], lag = 30, type = "Ljung-Box")

        Box-Ljung test

data:  mat_sqrd_resid_merged[, "IDV"]
X-squared = 25.197, df = 30, p-value = 0.7154

> Box.test(mat_sqrd_resid_merged[, "SHY"], lag = 30, type = "Ljung-Box")

        Box-Ljung test

data:  mat_sqrd_resid_merged[, "SHY"]
X-squared = 38.106, df = 30, p-value = 0.1471
```

These results fit what we assumed from the time series of squared residuals, that is IXC and IDV have no autocorrelation whereas SHY even though as a p-value more than 5% has the highest autocorrelation but not enough to reject the null hypothesis.

Lastly, we can test the normality assumption of the residuals with a visual diagnostic to form some assumptions and validate them statistically with a Jarque-Bera test.
Consequently, we can display the Q-Q plots of residuals and see if the empirical residuals' quantiles comfort with the standard normal ones especially in the tails.



Here, we clearly see that the normality assumption is good for medium size residuals but fails in the tails especially the left one. Moreover, it seems that IDV's residuals are closer to the normal distribution compared to the other ones.

Nevertheless, this graphical approach does not conclude if we should reject the normality assumption or not.
Accordingly, a Jarque-Bera test can be used to test the normality assumption and we should expect that IDV fits the normal distribution the best, namely it should have at least the lowest JB test statistics:

```
> # statistical diagnostic
> jarque.bera.test(sd_resid_ixc)

        Jarque Bera Test

data:  sd_resid_ixc
X-squared = 519.48, df = 2, p-value < 2.2e-16

> jarque.bera.test(sd_resid_idv)

        Jarque Bera Test

data:  sd_resid_idv
X-squared = 282.34, df = 2, p-value < 2.2e-16

> jarque.bera.test(sd_resid_shy)

        Jarque Bera Test

data:  sd_resid_shy
X-squared = 330.98, df = 2, p-value < 2.2e-16
```

Here, we can assert that the Jarque test statistics are all significantly larger than the quantile from the chi-squared distribution with 2 degrees of freedom at 5% (which is equal to 5.99), which translates into p-values close to 0.

Therefore, we cannot accept the normality assumption of the residuals for the 3 ETFs. Furthermore, the residuals' distribution for IDV is the closest to the normal distribution as it has a JB test of 282, even tough it does not follow a normal distribution. Instead, we could consider an asymmetric student's t distribution that add two extra parameters in addition to the ones of the normal distribution. The first one controlling the degree of thickness in the distribution tails (d1), and another one for the asymmetry of the distribution (d2). However, such an assumption adds 2 additional parameters to optimize for the maximum likelihood which could become too cumbersome to estimate. To see if it would plausible, we first need to check if the skewness and excess kurtosis of residuals are significant, as if they were not, it would increase the complexity of the model by adding two additional parameters.

```
> # statistical properties of residuals
> skewness(resid)
                IXC       IDV        SHY       GSPC
Skewness -1.077333 -1.453777 -0.4257869 -0.9395736
> kurtosis(resid)
                     IXC      IDV      SHY     GSPC
Excess Kurtosis 8.532609 10.56815 4.996475 7.794144
```

Here, we notice that the skewness is close to one in absolute value, so the residual's distribution is likely to be asymmetric but only slightly for SHY. Furthermore, the excess kurtosis is largely greater than 0, so the residuals' distribution is likely to be leptokurtic.

Now can move on to the multivariate GARCH models in order to model the conditional beta.

6.2) Multivariate GARCH model

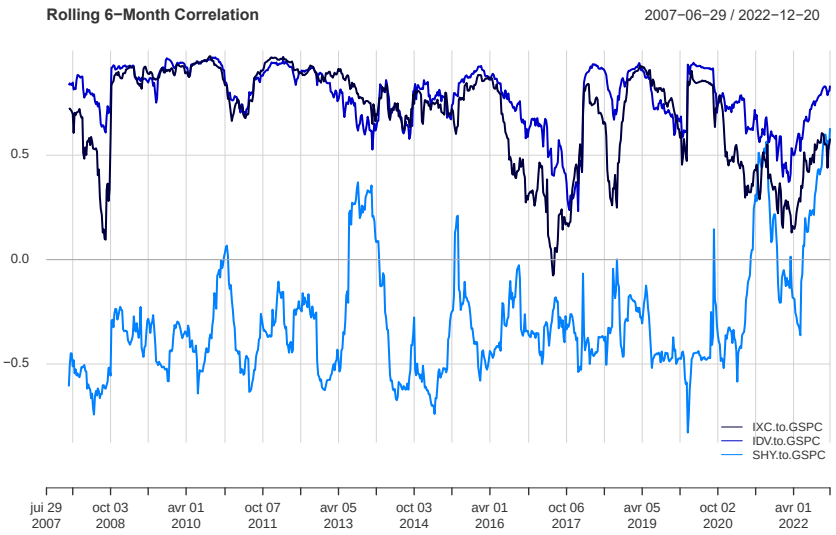
So far, we have been relying on setting the beta parameter of the Jensen equation to a constant, which we know is a simplified assumption. Indeed, the standard Jensen equation forget to consider the time variation of the covariances between each of our ETFs and the market returns as well as the volatility of the market return.

As a matter of fact, the classical beta can be expressed as the ratio of the unconditional covariance between an asset and the market returns divided by the unconditional variance of the market returns. Thus, the ratio is simply a constant, so it does not tap into the major innovation brought by time series models, namely using all the available information at t-1 to compute the metric at time t.

Here is the formula of the conditional CAPM that leverage all the information we have at our disposal before forecasting:

$$E\left[\left(r_{i,t}-r_{f,t}\right)\middle|\Omega_{t-1}\right]=\frac{E\left[\left(r_{m,t}-r_{f,t}\right)\middle|\Omega_{t-1}\right]}{Var\left[r_{m,t}\middle|\Omega_{t-1}\right]}Cov\left[r_{i,t},r_{m,t}\middle|\Omega_{t-1}\right]$$

Furthermore, we can display the moving correlation (which is the standardized version of the covariance) to corroborate our argument about the non-stationarity of the covariances:



Here we can observe the correlation's volatility and the mean-reversion of this one, which therefore supports our argument of estimating the Jensen equation with dynamic beta under a multivariate GARCH model.

6.2.1) Model fitting

Now we consider the rmgarch package for multivariate GARCH models using the same general specification for each of our ETFs. Namely, for the mean model, a constant mean with no ARMA orders. Yet, this time, we do not include the S&P 500 returns as external regressor because we want to model the covariances between our ETFs so as to compute the conditional beta. Besides, for the variance model and the distribution model, we consider the standard GARCH (1,1) model and the normality assumption for the GARCH shocks.
Afterwards, for the multivariate specification, we just simply replicate the general specifications 4 times for the 3 ETFs and the S&P 500 benchmark.

Nonetheless, when fitting the model, we do not incorporate a VaR model for the mean which is the default choice. From object-oriented programming knowledge, the dynamic conditional covariance object inherits and redefine all virtual methods from the multivariate GARCH class. In fact, the last is just a virtual class that provide the general architecture for designing a multivariate GARCH model and therefore cannot be instantiated directly. In our application, we prefer the DCC model rather a copula dynamic conditional covariance which is a much more advanced model, and the purpose is just to display the time variation of the beta.

Here, we showcase the lines of code that enable to perform such operations:

```
# -------------
# model fitting
# -------------

# same individual specification
general_spec <- ugarchspec(mean.model = list(armaOrder = c(0,0)),
                     variance.model = list(model = "sGARCH", garchOrder = c(1,1)),
                     distribution.model = "norm"
                     )

# coerce the same individual specifications as a multivariate specification
multiv_spec <- multispec( replicate(4, general_spec) )

# dynamic conditional covariance specification class but without a VaR model for the mean
dcc_spec <- dccspec(multiv_spec, VAR = FALSE)

# fitting the model to the data
multiv_fit <- dccfit(dcc_spec, data = mat_returns)
```

6.2.2) Feature extraction

Correspondingly, we can present some of the optimization outputs without needing to detail the purpose of each one which we have already done in the previous sections:

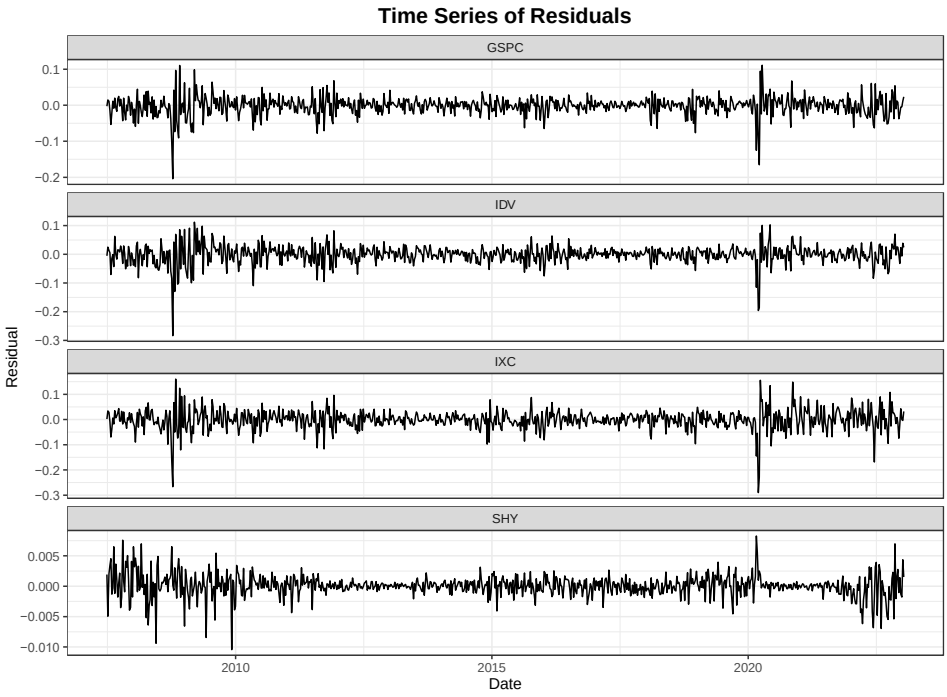
```
> # model summary
> show(multiv_fit)

*---------------------------------*
*          DCC GARCH Fit          *
*---------------------------------*

Distribution         :  mvnorm
Model                :  DCC(1,1)
No. Parameters       :  24
[VAR GARCH DCC UncQ] : [0+16+2+6]
No. Series           :  4
No. Obs.             :  809
Log-Likelihood       :  10349.55
Av.Log-Likelihood    :  12.79

Optimal Parameters
------------------------------------
               Estimate  Std. Error  t value Pr(>|t|)
[IXC].mu       0.000797    0.000954  0.83594 0.403190
[IXC].omega    0.000037    0.000017  2.21372 0.026848
[IXC].alpha1   0.195516    0.060383  3.23794 0.001204
[IXC].beta1    0.801993    0.042137 19.03314 0.000000
[IDV].mu       0.000400    0.000826  0.48412 0.628301
[IDV].omega    0.000051    0.000035  1.46690 0.142403
[IDV].alpha1   0.229457    0.093928  2.44290 0.014570
[IDV].beta1    0.732304    0.095485  7.66933 0.000000
[SHY].mu      -0.000055    0.000039 -1.41954 0.155741
[SHY].omega    0.000000    0.000000  0.04247 0.966124
[SHY].alpha1   0.106092    0.027952  3.79551 0.000147
[SHY].beta1    0.886515    0.029548 30.00298 0.000000
[GSPC].mu      0.003072    0.000633  4.85170 0.000001
[GSPC].omega   0.000040    0.000012  3.24232 0.001186
[GSPC].alpha1  0.334694    0.073135  4.57638 0.000005
[GSPC].beta1   0.641630    0.054279 11.82094 0.000000
[Joint]dcca1   0.036193    0.008612  4.20268 0.000026
[Joint]dccb1   0.944416    0.011098 85.09892 0.000000

Information Criteria
--------------------

Akaike       -25.527
Bayes        -25.387
Shibata      -25.528
Hannan-Quinn -25.473
```

Here, we notice that the only coefficients that are not statistically different from 0 are the mu and omega parameters which are the constants of the mean and variance models respectively. Nevertheless, non-significant constants are not problematic from an

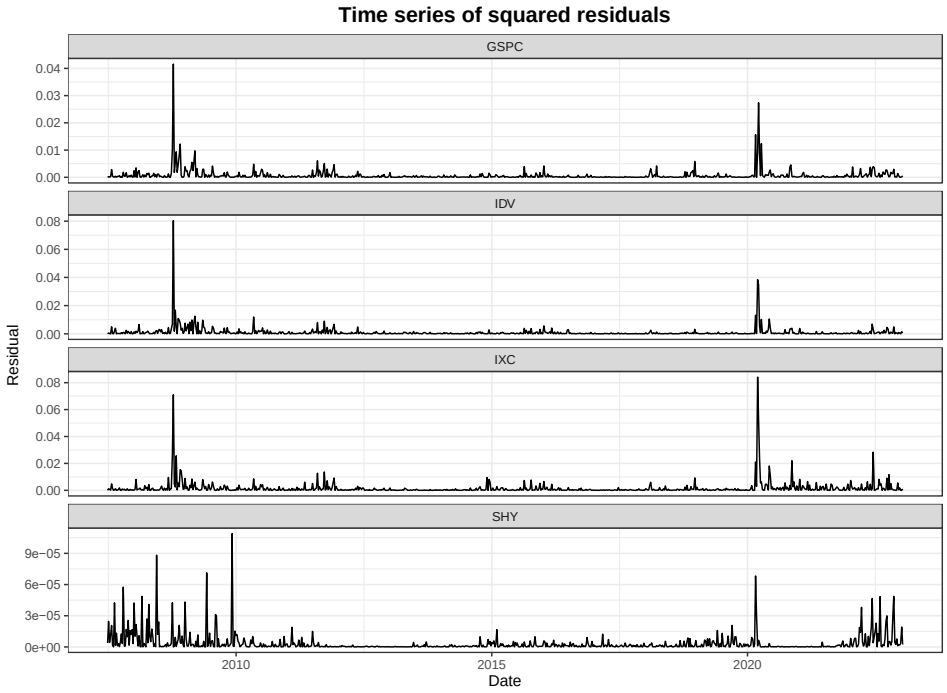
econometric point of view. On the other hand, all the other GARCH coefficients, namely alpha and beta are all statistically different from 0. Lastly, the two last joint parameters are also significant.

6.2.3) Residual analysis

Without presenting the purpose of each diagnostic which are the same than the previous section, we can comment on the results we have obtained:

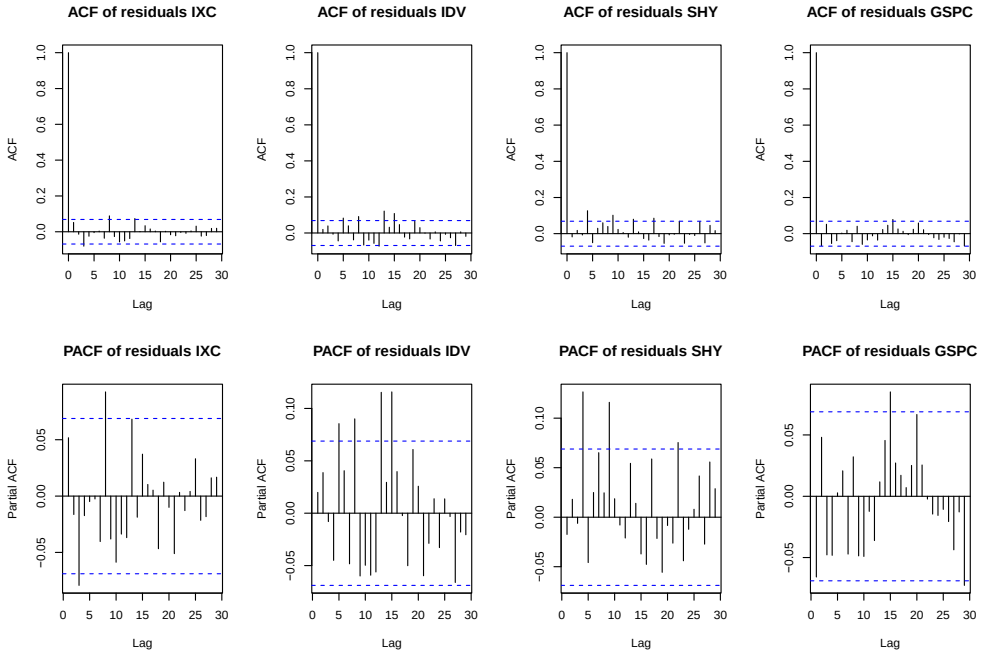


Here, we see that the residuals are more clustered than in the previous section, namely the variance is not stationary at all, and this can be emphasized by considering the time-series of squared residuals instead.



Here we can see different volatility regimes with periods of high volatilities and ones of very low volatility. For instance, we can see that from 2007 to 2010, SHY displays a high volatile regime whereas from 20212 to 2015 its volatility is extremely low.

The ACF and PACF can also be displayed and based on the previous plots, we can expect much more autocorrelation than in the previous section:



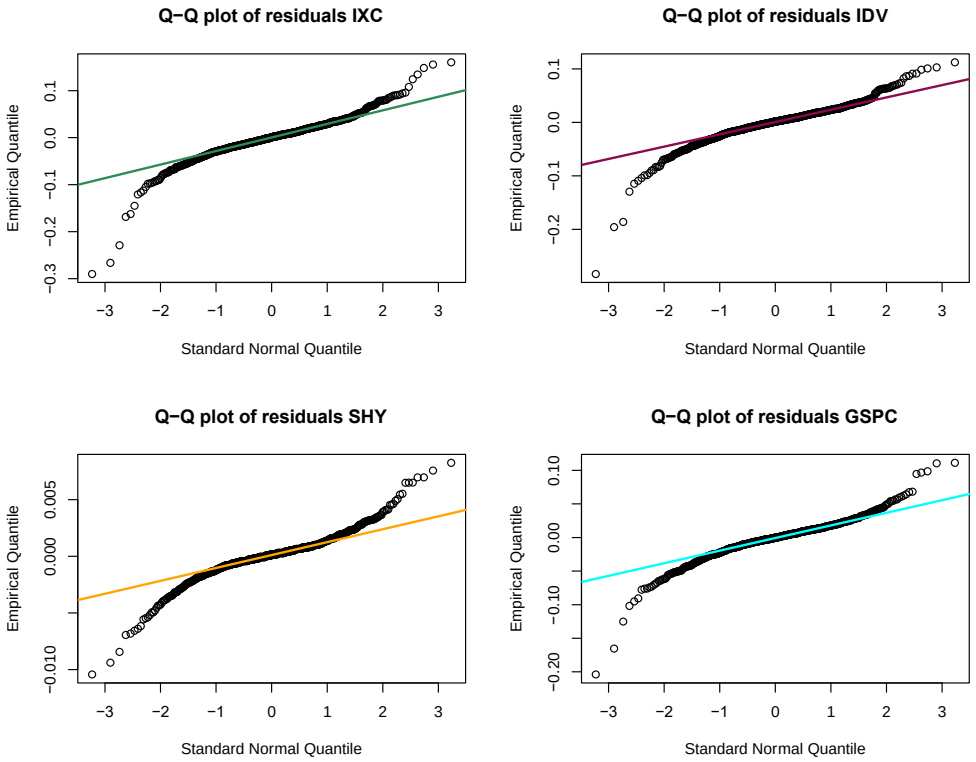
Here, we see that this time SHY has strong autocorrelations, with for the lags 4 and 9 the ACF and PACF surpasses the 95% confidence bands. The same with IDV which has significant autocorrelation coefficients different from 0 in both its ACF and PACF. However, IXC still does not have any pattern in both its ACF and PACF and the same with GSPC.

Therefore, a Ljung-Box test can be used to validate statistically the visual diagnostic:

```
> # statistical diagnostic
> lj_box <- apply(resid, 2, Box.test, lag = 30, type = "Ljung-Box")
> lj_box$IXC

        Box-Ljung test

data:  newX[, i]
X-squared = 34.185, df = 30, p-value = 0.2735

> lj_box$IDV

        Box-Ljung test

data:  newX[, i]
X-squared = 69.154, df = 30, p-value = 6.295e-05

> lj_box$SHY

        Box-Ljung test

data:  newX[, i]
X-squared = 60.058, df = 30, p-value = 0.0009059

> lj_box$GSPC

        Box-Ljung test

data:  newX[, i]
X-squared = 37.701, df = 30, p-value = 0.1576
```

The Ljung-Box test statistic for IXC is relatively low compared to the 2 next ones, which highlight that the autocorrelations are not significant. Therefore, this translates into a p-value is larger than any standard confidence levels, namely 1%, 5%, and 10%, which lead to not reject the null hypothesis.
In contrast, IDV and SHY both exhibit p-values less than 1%, which therefore lead to reject the null hypothesis of the absence of autocorrelation up to the 30^{th} lag. Lastly, our benchmark GSPC has a p-value larger than any confidence level so we cannot reject the null hypothesis and it comforts our assumptions from the ACF and PACF plots.

Finally, the last residual diagnostic consists in validating the normality assumption.
For that, a first visual check with Q-Q plots enable to give a first assumption about the normality distribution of the empirical residuals' quantiles.



Here we see that the normality assumption is good for the center of the distribution, but the tails are much higher than the ones of the standard normal distribution, so we should expect the rejection of the normality assumption from a statistical test.

To corroborate our observation, we can test for normality with the same Jarque-Bera test used previously:

```
> for(ticker in tickers) {
+   print(ticker)
+   print(jarque.bera.test(resid[, ticker]))
+ }
[1] "IXC"

        Jarque Bera Test

data:  resid[, ticker]
X-squared = 2620.3, df = 2, p-value < 2.2e-16

[1] "IDV"

        Jarque Bera Test

data:  resid[, ticker]
X-squared = 4064.7, df = 2, p-value < 2.2e-16

[1] "SHY"

        Jarque Bera Test

data:  resid[, ticker]
X-squared = 869.18, df = 2, p-value < 2.2e-16

[1] "GSPC"

        Jarque Bera Test

data:  resid[, ticker]
X-squared = 2174.8, df = 2, p-value < 2.2e-16
```

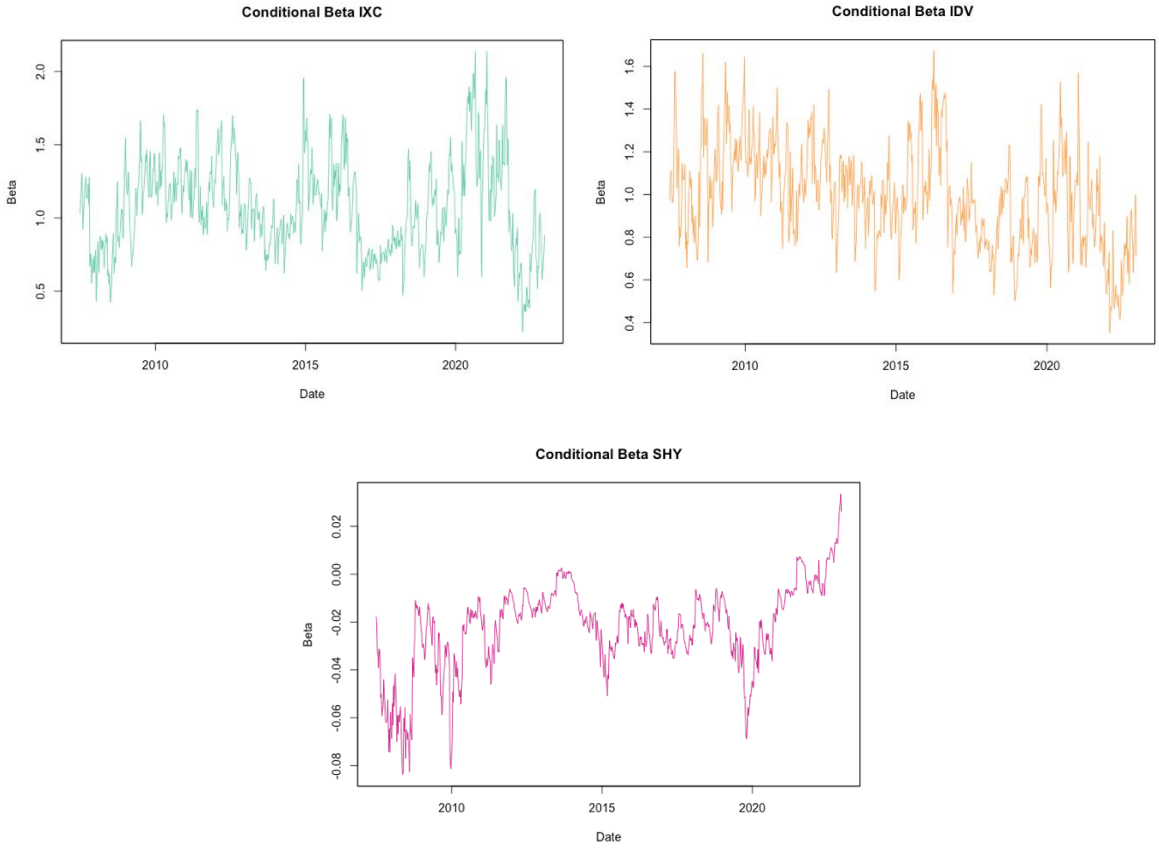
With p-values close to 0, we cannot accept the null hypothesis of normality for all our ETFs and our benchmark.

Now, let us move on the construction of the conditional beta based on the features of the multivariate GARCH model.

6.2.4) Conditional beta

One new feature provided the multivariate GARCH model is the estimated conditional covariance matrix.
From this conditional covariance matrix, we can extract the conditional covariances between each of our ETFs and the S&P 500 that we can divide with the conditional variance of the S&P 500 in order to obtain the conditional betas.
Then, we can plot the time series of the betas:

```r
# ----------------
# conditional beta
# ----------------

# filtered dynamic conditional covariance array (the third dimensions gives the time index)
cov_matrix <- rcov(multiv_fit)

# time-varying betas (extract a time series of beta for each time index)
cond_var_gspc <- cov_matrix["GSPC", "GSPC", ]
tv_beta_ixc <- cov_matrix["IXC", "GSPC", ] / cond_var_gspc
tv_beta_idv <- cov_matrix["IDV", "GSPC", ] / cond_var_gspc
tv_beta_shy <- cov_matrix["SHY", "GSPC", ] / cond_var_gspc

# go back to xts to have dates in the x-axis
tv_beta_ixc <- as.xts(tv_beta_ixc, index(mat_returns))
tv_beta_idv <- as.xts(tv_beta_idv, index(mat_returns))
tv_beta_shy <- as.xts(tv_beta_shy, index(mat_returns))
```



Here we can see that the beta fluctuates between 0.5 and 2.0 for IXC, and between 0.4 and 1.5 for IDV. In contrast, the beta for SHY fluctuates only between – 0.08 and 0.02 which indicates that it is relatively insensitive to the S&P 500 and that these changes are relatively small compared to the other ETFs. Moreover, the beta for SHY is most of the time in negative territory as bond prices are most of the time negatively correlated to stock prices.

7) Sharpe-based performance measures

Here we define Sharpe-based performance measures as measures that use as a blueprint the standard Sharpe ratio but try to fix some of its limits. Furthermore, these performance measures are computed based on the return distribution. In contrast, draw-dawn based performance measures are calculated based on the drawdown distribution. Nevertheless, the similarity of these Sharpe ratio's extensions is that they try to address some of its shortcomings. Therefore, we are going to discuss each one in the following sections and comment on to what extent they correct the limits of the Sharpe ratio.

In the rest of this section, we assume a risk-free rate of 0% as there is no indication as well as a confidence level of 5%. However, we provide a modular code so that if we had to test for other values, we would just have to modify one line of code.

7.1) Standard Sharpe ratio

The standard Sharpe ratio is the most widely known risk adjusted performance measure, which is key in finding the market portfolio in the CAPM which is also known as the maximum Sharpe ratio portfolio.

The formula for computing it is obtained as follows:

$$Sharpe\ Ratio = \frac{E(R_p) - R_f}{\sigma_p}$$

$where: E(R_p)\ is\ the\ expected\ portfolio\ return,$

$R_f\ is\ the\ risk - free\ rate,$

$and\ \sigma_p\ is\ the\ portfolio's\ standard\ deviation$

In R, we can compute it manually or using built-in functions, but in this case, we want to detail all the implementation details, so we do it manually. Hence, we compute the expected return as the simple arithmetic mean for each ETFs and subtract the risk-free rate in order to obtain the excess return. Then, we must divide each excess return by the volatility of each ETFs to finally obtain the Sharpe ratio.
Additionally, we can compute its annualized version which is often used for comparing between portfolios. For that, we must multiply the excess return by the number of periods in a trading year, while for the portfolio's volatility the square root of the number of periods. In our case, as we have weekly returns and that we have approximately 52 trading weeks in a

year, we must multiply the numerator by 52 and the denominator by $\sqrt{52}$ so the whole ratio by $\sqrt{52}$.

The code for computing the weekly as well as the annualized Sharpe ratios is provided as follows:

```
> # *********************
> # Standard Sharpe ratio
> # *********************
>
> # weekly Sharpe ratio
> expected_returns <- colMeans(mat_returns)
> excess_returns <- expected_returns - mean(risk_free_rate)
> st_dev <- apply(mat_returns, 2, sd)
> sharpe_ratio <- excess_returns / st_dev
> print(sharpe_ratio)
         IXC          IDV          SHY         GSPC
-0.003896081 -0.023456893  0.011264218  0.043574456
>
> # annualized Sharpe ratio assuming 52 trading weeks in a year
> ann_expected_returns <- expected_returns * 52
> ann_risk_free_rate <- risk_free_rate * 52
> ann_st_dev <- sqrt(52) * st_dev
> ann_sharpe_ratio <- (ann_expected_returns - mean(ann_risk_free_rate)) / ann_st_dev
> print(ann_sharpe_ratio)
        IXC         IDV         SHY        GSPC
-0.02809504 -0.16915006  0.08122743  0.31421987
```

The relative order between our ETFs and the S&P 500 is the same as we could expect, but the performance is worse for the annualized version for IXC and IDV. Indeed, as they have a negative expected return, accordingly their weekly Sharpe ratio is negative, and as we multiply the whole by $\sqrt{52}$ the result is worse. Nonetheless, for SHY and GSPC, the annualized ones are larger than the weekly ones as we have positive weekly Sharpe ratios. As a consequence, we can argue that the best investment is our benchmark GSPC so none of our ETFs outperform it. However, we can still classify our ETF from the best to the worst, namely, SHY followed by IXC, and the worst one being IDV.

Nevertheless, even though the standard Sharpe ratio is the most widely used risk adjusted performance measure, more powerful ratios that leverage the stylized facts about financial returns can be considered instead.

7.1) Smart Sharpe ratio

So far, we have been using the standard deviation but without considering the autocorrelation of returns. Indeed, for small lags, it seems that taking into account the autocorrelations would provide a more accurate measure of the total variation of our ETFs. As a matter of fact, especially during periods of market stress, periods of high volatility tend to be followed by periods of high volatility and vice-versa. Therefore, from the

autocorrelation function (ACF) we can take the first lag and add it to the standard deviation to obtain Sharpe ratio corrected for the autocorrelation, aka the smart Sharpe ratio.
The general formula is provided as follows where q denotes the maximum lag autocorrelation order to consider:

$$SRq = SR \times \frac{q}{\sqrt{q + 2\sum_{l=1}^{q-1}(q-l)\rho_l}}$$

To accomplish this in R, from the acf function we can take the first lag, extracting the corresponding first order autocorrelations and plug it into the formula. The code for achieving this is provided as follows:

```
> # ******************
> # Smart Sharpe ratio
> # ******************
>
> # autocorrelation array
> array_acf <- acf(mat_returns, lag.max = 1, plot = FALSE)$acf
>
> # understand the structure to extract the first order autocorrelation
> print(array_acf)
, , 1

           [,1]       [,2]       [,3]         [,4]
[1,] 1.00000000 0.79326575 -0.2018662  0.748381551
[2,] 0.05177067 0.05724035 -0.0957552 -0.009945838

, , 2

             [,1]       [,2]        [,3]        [,4]
[1,] 0.7932657532 1.00000000 -0.21800394  0.84712986
[2,] 0.0008136605 0.01956925 -0.03040468 -0.03354436

, , 3

            [,1]        [,2]        [,3]         [,4]
[1,] -0.20186625 -0.21800394  1.00000000 -0.243832214
[2,] -0.04128561 -0.03316771 -0.01729431  0.006857107

, , 4

            [,1]        [,2]        [,3]        [,4]
[1,]  0.74838155 0.847129857 -0.24383221  1.00000000
[2,] -0.01311375 0.009604148 -0.06540801 -0.06584453

> # 3-dimensional array
> dim(array_acf)
[1] 2 4 4
>
> # instantiate an empty matrix to store the results
> rho <- matrix()
> # extract the first order autocorrelation for each ETF
> for(ticker in 1:4) {
+   # extract the second row (the first row is order 0) for each ETF
+   rho[ticker] <- array_acf[2, ticker, ticker]
+ }
>
```

From the acf function, we obtain a 3-dimensional array where the rows correspond to the autocorrelation's order with the first row being the order 0 so the ACF equals to 1, and the second being the first order. Furthermore, the columns are the auto and cross

autocorrelations and the third dimension corresponding to each ETF. Therefore, the first matrix gives us the auto and cross autocorrelations for the ETF SHY, with the first column being the autocorrelation and other being the cross ones. We can repeat this reasoning for the second matrix which is for IDV, so the second columns are the autocorrelation and the other are the cross and so on. Thus, based on this observation, we can extract the corresponding first order ACF for each ETF and compute the smart Sharpe ratio given by the formula.

```
> # compute the smart Sharpe ratio aka the long-run variance sharpe ratio
> smart_sharpe_ratio <- sharpe_ratio * 2 / (2 + rho)
> print(smart_sharpe_ratio)
         IXC          IDV          SHY         GSPC
-0.002068459 -0.020743472  0.013606473  0.047110851
> # is the smart Sharpe ratio larger than the standard one ?
> print(smart_sharpe_ratio > sharpe_ratio)
 IXC  IDV  SHY GSPC
TRUE TRUE TRUE TRUE
> # show the relative difference
> print((smart_sharpe_ratio / sharpe_ratio) - 1)
         IXC          IDV          SHY         GSPC
-0.024966266 -0.009972982  0.008619720  0.033830720
```

To see if our results look plausible, we compare it with the standard Sharpe ratio and as we could have expected they are almost the same, as we have seen in previous sections that the ACF is very low.
It is important to note that we do not see significant differences because the autocorrelations are only significant for very small intraday frequencies (below 20 minutes) due to microstructure effects.

7.3) Sharpe-VaR ratio

So far, we have been focusing on the standard deviation as a risk measure, which is the standard metric used to analyze risk, but this measure is sometimes inappropriate.

Indeed, the standard deviation penalizes equally deviations above and below the mean, and it is not focused on the left tail of the return distribution which is what investors care about. One popular tail risk measure is the value at risk (VaR) which is reported for a given horizon and confidence level.
The VaR is defined as the minimum loss in the p% worst case scenarios or as the maximum loss an investor would stomach in the (1-p) % best cases.
As we have weekly returns and specified a confidence level of 5%, we therefore consider the 1-week 5% VaR.

The general formula for the VaR is defined as follows, where X denotes a loss in monetary unity:

$$\mathrm{VaR}_{1-\alpha}(X) := \inf_{t \in \mathbf{R}} \{t : \Pr(X \leq t) \geq 1 - \alpha\}$$

Lastly, we provide different methods for computing the VaR which produce different results.

7.3.1) Gaussian method

First, we can assume that the return distribution follows a normal distribution and compute the gaussian VaR from the quantiles of the standard normal distribution. Here, the 5th percentile is - 1.69, and based on the corresponding expected return and standard deviation we can compute the 5% VaR accordingly.

The general formula for the gaussian VaR is as follows:

$$Gaussian\ VaR^{\alpha} = -(\mu + z_{\alpha}\sigma^2)$$

$where: \mu\ is\ the\ expected\ return,$

$z_{\alpha} the\ \alpha\ \%\ quantile\ from\ the\ standard\ normal\ dsitribution,$

$\sigma^2 the\ variance$

This formula can be easily replicated in R as the only remaining component to compute is the 5th quantile which can be computed straightforwardly:

```
> # ****************
> # Sharpe-VaR Ratio
> # ****************
>
> # ---------------
> # gaussian method
> # ---------------
> # compute the standard normal 5th percentile
> std_norm_quantile <- qnorm(p = conf_level, mean = 0, sd = 1)
> # - sign in front of the VaR as it represents a loss
> VaR_gauss_95 <- - (expected_returns + st_dev * std_norm_quantile)
> sharpe_VaR_gauss_95 <- excess_returns / VaR_gauss_95
> print(VaR_gauss_95)
       IXC         IDV         SHY        GSPC
0.065212417 0.054055411 0.002999467 0.042412754
> print(sharpe_VaR_gauss_95)
         IXC          IDV          SHY         GSPC
-0.002363052 -0.014060268  0.006895379  0.027212279
```

Here, we obtain Sharpe ratios that are lower than the previous ones, but this is because the VaR is much higher than the weekly standard deviation.

However, the relative order between our investments is still the same with the best being our benchmark followed by SHY and IXC, and the worst being IDV.

Nevertheless, the main drawback of this approach is that it relies on the normality assumption which we saw in previous sections is wrong. Therefore, we can consider a model-free approach such as the historical method.

7.3.1) Historical method

The downside of the gaussian or parametric approach, is the normality assumption for the return distribution. Consequently, a model-free approach could be more valuable.

The historical method does not compute the quantile from a particular distribution, but instead simply computes the quantiles form the return distribution directly. One of the main advantages of such an approach is that it is easy to implement, namely it does not require any numerical optimization. Furthermore, this method is considered model-free, namely it does not rely on any parametric model.
Hence, the method consists in sorting all returns in ascending order from the lowest to the highest, taking the number of observations and multiplying it with the corresponding confidence level. For instance, if we have 100 observations and a confidence level of 5%, the 5% VaR will be the 5th worst return. Notice that it is the 5% quantile so we could directly use the quantile function but detailing how the function operates under the hood is interesting.

```
> # ------------------
> # historical  method
> # ------------------
> # number of observations
> nb_obs <- nrow(mat_returns)
> # sort the returns in ascending order
> sorted_returns <- apply(mat_returns, 2, sort)
> # take the appropriate index
> idx_largest_losses <- nb_obs * conf_level
> # extract the corresponding VaR
> VaR_hist_95 <-  - sorted_returns[idx_largest_losses, ]
> sharpe_VaR_hist_95 <- excess_returns / VaR_hist_95
>
> print(VaR_hist_95)
        IXC         IDV         SHY        GSPC
0.059969693 0.051880709 0.002955959 0.043175511
> print(sharpe_VaR_hist_95)
         IXC          IDV          SHY         GSPC
-0.001398073 -0.013080458  0.008402363  0.027921790
> # is the Sharpe-hist VaR higher than the gaussian one ?
> print((sharpe_VaR_hist_95 / sharpe_VaR_gauss_95) - 1)
        IXC         IDV         SHY        GSPC
 0.08539973  0.03995124  0.01609443 -0.02006459
```

Here, the results are very similar to the gaussian VaR, with a VaR which is slightly higher for GSPC, but lower for the 3 ETFs. Consequently, the Sharpe VaR computed with the historical method is very similar to the previous Sharpe VaR with the gaussian method.

However, the major downside of such an approach, is that the model puts equal weights on past returns treating the return 10 years ago as having the same impact as the return of last week. Therefore, the VaR does not adapt quickly to a market crash and provides excessive cautious when the volatility in the market has plummeted. To remedy this drawback, the weighted historical simulation could be an alternative as it allows to respond more quickly to large losses. But we can still correct the gaussian method by considering higher central moments that we are going to see in the next section.

7.3.1) Cornish-Fisher method

One of the main issues of the gaussian method, namely the underlying normality assumption, can be corrected by adding the sample skewness and excess kurtosis that enable to add additional information on the left tail of the return distribution.

The method consists in correcting the standard normal quantile by the Cornish fisher extension that incorporates the skewness and excess kurtosis multiplied by some coefficients that depend on given confidence level.
The aim of the Cornish fisher extension is to penalize the standard normal quantile when the skewness is negative and/or the kurtosis is positive. Indeed, a negative and a positive excess kurtosis decrease the utility of the investor and so would lead to a higher risk. The formula as well as the coefficients table are presented as follows:

$$\text{VaR modifiée} = -V \times (\mu_T + z_\alpha^{\text{Cornish-Fisher}} \sigma_T)$$

$$z^{\text{Cornish-Fisher}} = z + a.S + b.(K-3) + c.S^2$$

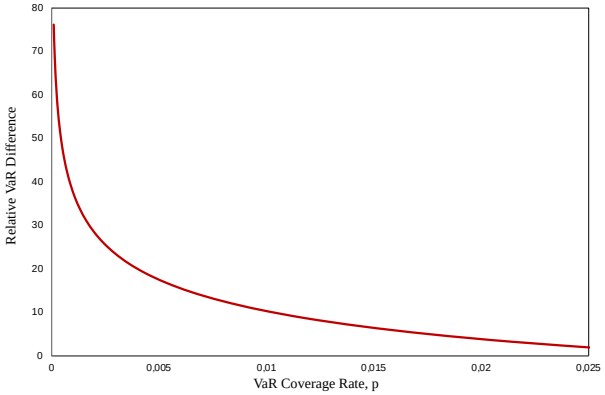
α	z	a	b	c
0,1%	-3,090	1,425	-0,843	1,210
0,5%	-2,576	0,939	-0,390	0,592
1,0%	-2,326	0,735	-0,234	0,376
5,0%	-1,645	0,284	0,020	0,019
10,0%	-1,282	0,107	0,072	-0,061

Therefore, we can perform these steps in R with the following lines of code:

```
> # ---------------------
> # cornish-fisher method
> # ---------------------
>
> # compute the 5% normal quantile
> z_std_norm_5 <- std_norm_quantile
> # define the coefficients for a confidence level of 5%
> a <- 0.284
> b <- 0.020
> c <- 0.019
> skew <- skewness(mat_returns)
> kurt <- kurtosis(mat_returns)
>
> # compute the cornish-fisher quantile
> z_cf_5 <- z_std_norm_5 + a * skew + b * as.matrix(kurt) + c * as.matrix(skew)^2
> rownames(z_cf_5) <- NULL
> print(z_std_norm_5)
[1] -1.644854
> print(z_cf_5)
        IXC      IDV       SHY      GSPC
1 -1.757992 -1.80589 -1.664249 -1.738491
> print(skew)
              IXC       IDV        SHY       GSPC
Skewness -1.07459 -1.452833 -0.4377671 -0.9360636
> print(kurt)
                     IXC      IDV      SHY     GSPC
Excess Kurtosis 8.50524 10.57321 5.064472 7.777831
>
> VaR_cf_95 <- - (expected_returns + st_dev * z_cf_5)
> sharpe_VaR_cf_95 <- excess_returns / VaR_cf_95
> print(VaR_cf_95)
          IXC         IDV         SHY      GSPC
1 0.06969899 0.05927464 0.003036308 0.0448828
> print(sharpe_VaR_cf_95)
           IXC          IDV        SHY       GSPC
1 -0.00174524 -0.01270738 0.00661151 0.02587584
> # is the CF VaR higher than the gaussian VaR ?
> print(VaR_cf_95 > VaR_gauss_95)
      IXC  IDV  SHY GSPC
[1,] TRUE TRUE TRUE TRUE
```

Here we see that we obtain corrected quantiles that are slightly larger in absolute value than the 5% standard normal quantile of 1.64, with almost no difference for SHY.
Indeed, the cons of this approach is that the extra information provided by the third and fourth central moment are relevant only if we focus on the extreme deviations from the mean, namely largely beyond the 5% quantile. Therefore, there is almost no difference in the VaRs between gaussian and Cornish-Fisher VaR which is illustrated with this graph.

Figure 2.8: Relative Difference between Non-Normal (Excess Kurtosis=3) and Normal VaR



Here, we see that to obtain a significative difference, we should rather consider the 1% VaR so as to capture the extreme left tail. To stress our reasoning, we can compute the Cornish fisher quantiles and compare it with the 1% quantile from the standard normal distribution and expect at least corrected quantiles that are many times higher:

```
> # compute the 1% normal quantile
> z_std_norm_1 <- qnorm(p = 0.01, mean = 0, sd = 1)
> # define the coefficients for a confidence level of 1%
> a <- 1.425
> b <- - 0.843
> c <- 1.210
> # compute the cornish-fisher quantile
> z_cf_1 <- z_std_norm_1 + a * skew + b * as.matrix(kurt) + c * as.matrix(skew)^2
> rownames(z_cf_1) <- NULL
> print(z_std_norm_1)
[1] -2.326348
> print(z_cf_1)
        IXC       IDV       SHY     GSPC
1 -9.630316 -10.75588 -6.987632 -9.15673
```

Here we see that Cornish-fisher quantiles are more than 3 times higher for SHY, and almost 3 times for the others which fits what we could attend.

7.4) Sharpe-CVaR ratio

As we have seen in the previous section, one important flaw of the VaR is that it is only sensitive to the p^{th} quantile and not the observations beyond. Furthermore, the VaR is not considered as coherent risk measure as it does not respect to a very property, namely the sub additivity. As a matter of fact, the sum of the VaR of 2 assets could lead to a higher VaR when holding the 2 assets at the same time, so it does not follow the diversification principle.

To remedy these drawbacks, the fundamental review of the trading book in 2017 introduced the new standard risk measure in the internal model approach for computing the market risk for bank capital requirements, namely the Expected Shortfall or Conditional VaR.

Formally, the CvaR is defined as the average losses for those beyond the VaR as given by the formula below:

$$CVaR_{\alpha}(X) = E(X|X \geq VaR_{\alpha}(X))$$

where X denotes a loss

Similarly to the VaR, the CVaR can be computed with different methods that we are going to describe in the following sections.

7.4.1) Gaussian method

The gaussian CVaR follows the same principle than the gaussian VaR, but this time we must consider the truncated standard normal distribution conditional on it being below the VaR. The general formula is provided as follows:

$$CVaR = \mu + \sigma \frac{\varphi(\Phi^{-1}(\alpha))}{\alpha}$$

$$where,$$
$$\mu = Portfolio\ mean\ return$$
$$\sigma = Portfolio\ standard\ deviation$$
$$\varphi = Normal\ distribution\ PDF$$
$$\Phi^{-1} = Standard\ Normal\ Quantile$$
$$\alpha = Confidence\ Interval$$

Accordingly, we can accomplish that in R with the following lines of code:

```
> # *****************
> # Sharpe-CVaR Ratio
> # *****************
>
> # ---------------
> # gaussian method
> # ---------------
> # compute the pdf
> phi <- dnorm(z_std_norm_5, mean = 0, sd = 1)
> CVaR_gauss_95 <- expected_returns + st_dev * (phi / conf_level)
> sharpe_CVaR_gauss_95 <- excess_returns / CVaR_gauss_95
> print(CVaR_gauss_95)
        IXC         IDV         SHY        GSPC
0.081515855 0.066090750 0.003808234 0.055792625
> print(sharpe_CVaR_gauss_95)
         IXC          IDV          SHY         GSPC
-0.001492243 -0.011396839  0.005271363  0.020816013
> # is the CVaR always larger than the VaR ?
> print(CVaR_gauss_95 > VaR_gauss_95)
 IXC  IDV  SHY GSPC
TRUE TRUE TRUE TRUE
> # relative difference between CvaR and VaR
> print((CVaR_gauss_95 /VaR_gauss_95) - 1)
      IXC       IDV       SHY      GSPC
0.2498364 0.2226322 0.2691199 0.3157768
```

Here we note that the CVaR is always larger than the VaR, which is what we could have expected. Indeed, one property of the CVaR is that it is always larger in absolute value than the VaR computed on the same time horizon and the same confidence level. Furthermore, the relative difference between the CvaR and the VaR is approximately 25% for all our ETFs, so it is significant. Therefore, we obtain lower Sharpe ratios because the risk measure is higher. The main drawback of the gaussian method can also be corrected using a model-free approach as have done in the previous section.

7.4.2) Historical method

We can also compute the historical CVaR where the idea is very similar to the historical VaR. In contrast to the historical VaR, instead of considering the k-observation from the index obtained by multiplying the number of observations by the confidence level, we take all the worst returns until this index and take the mean of these extreme returns.

The code for realizing this operation is provided as follows:

```
> # -----------------
> # historical method
> # -----------------
>
> # consider the returns worst than the historical VaR at 95%
> k_largest_losses <- sorted_returns[1:idx_largest_losses,]
> # CvaR is the mean of these returns
> CVaR_hist_95 <- - colMeans(k_largest_losses)
> sharpe_CVaR_hist_95 <- excess_returns / CVaR_hist_95
> print(CVaR_hist_95)
        IXC         IDV         SHY        GSPC
0.099354116 0.084527039 0.004737688 0.066295511
> print(sharpe_CVaR_hist_95)
         IXC          IDV          SHY         GSPC
-0.001224323 -0.008911062  0.004237211  0.017518230
> # is the hist CvaR larger than the gaussian one ?
> print(CVaR_hist_95 > CVaR_gauss_95)
 IXC  IDV  SHY GSPC
TRUE TRUE TRUE TRUE
> # relative difference between the hist CvaR and the gaussian
> print((CVaR_hist_95 / CVaR_gauss_95) - 1)
      IXC       IDV       SHY      GSPC
0.2188318 0.2789541 0.2440643 0.1882487
```

Here, we see that the historical CVaRs are approximately 20% large than the gaussian ones, which is a significative difference. Here too, the Sharpe ratios are accordingly lower than all previous methods for SHY and GSPC, but higher for IXC and IDV.

This result indicates that the extreme observations from the normal distribution are less frequent than for our sample, which therefore emphasize that left tail of our return distribution is higher than the gaussian. However, the flaw of the historical CVaR is that extreme returns are pushed only by outliers, so it is not a robust approach. But in this case by considering a 5% confidence level, we obtain a more accurate result that if we had considered a confidence level of 1% for instance. Indeed, the problem with the CvaR is that if we consider a small confidence level, for instance 1%, we will not have a sufficient observations which therefore could lead a bias estimate. In contrast, by considering a larger confidence level, for instance 10%, we will obtain a high variance in our estimate because the 10% worst outcomes are too far from the extreme left tail.

7.4.3) Cornish-Fisher method

Lastly, we can tap into the information provided by the sample skewness and excess kurtosis to adjust the standard normal quantile. In this case, we cannot directly compare the quantiles, but instead we will compare the risk measures directly.

```
> # ---------------------
> # cornish-fisher method
> # ---------------------
>
> y_05 <- st_dev * (phi / cov_rate)
> # define the coefficients for a confidence interval of 95%
> m <- -0.2741
> p <- -0.1225
> q <- 0.0711
>
> Y_05 <- y_05 * (1 + m * skew + p * skew^2 + q * kurt)
> rownames(Y_05) <- NULL
> CVaR_cf_95 <- expected_returns + Y_05
> sharpe_CVaR_cf_95 <- excess_returns / CVaR_cf_95
> print(CVaR_cf_95)
        IXC       IDV         SHY       GSPC
1 0.1433817 0.1256763 0.005537908 0.09415701
> print(sharpe_CVaR_cf_95)
            IXC          IDV        SHY      GSPC
1 -0.0008483754 -0.005993377 0.00362494 0.0123345
> # is the cf CVaR larger than the normal CVaR ?
> print(CVaR_cf_95 > CVaR_gauss_95)
      IXC  IDV  SHY GSPC
[1,] TRUE TRUE TRUE TRUE
> # relative difference between CF CvaR and gaussian CVaR
> print((CVaR_cf_95 / CVaR_gauss_95) - 1)
        IXC       IDV       SHY      GSPC
1 0.7589424 0.9015724 0.4541931 0.6876246
```

Here we see a positive relative difference highlighting a Cornish-Fisher CVaR which is always larger than the gaussian one. In addition, the relative difference is the highest for IDV with a Cornish-Fisher CVaR which is almost two times larger. Nevertheless, as SHY has the lowest risk profile it shows a relative difference of only 50%.

8) Drawdown performance measures

Up to now, we have proposed extensions of the standard deviation as a risk measure based on the return distribution to compute risk adjusted performances. However, we have neglected one important aspect that causes pain to investor, namely the drawdown risk. As a matter of fact, successive negative returns provoke more pain to investors than one-period negative returns considered so far from the return distribution.

Therefore, it would not be exhaustive if we did not consider this psychological characteristic in our modelling approach.

8.1) Draw-down calculation

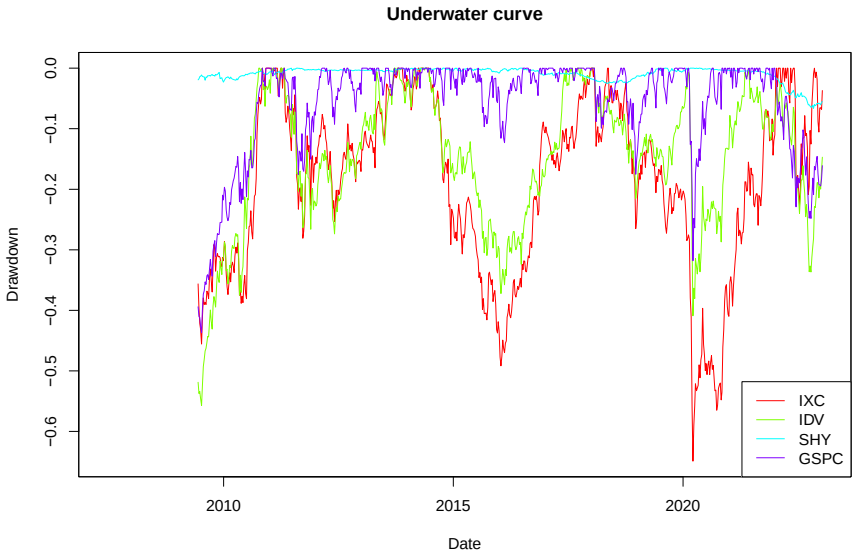
In this section, we are interested in computing the drawdowns based on a 2-year rolling window, that is discarding the latest observation and adding the most recent one. As there are almost 52 trading weeks in a year, the window size is accordingly 104. As we are using a rolling window, we must be careful about how our data is going to be treated. Indeed, rolling windows allow to compute time varying statistics by considering data only for that window. Therefore, the first data will be available only after the window size, namely after the 104th weekly return, so the time series will have the total number of observations minus the window size, namely 804 – 104 observations.

As such, we create an empty xts matrix to store our results, and in a for loop, we take the maximum of the rolling window adding the most recent weekly prices and discarding the last ones. Finally, we can showcase the first few NA observations as well as the first available ones, namely after the window size.

```
> # ********************
> # Draw-down Calculation
> # ********************
>
> # window size of 2 years
> window_size <- 2 * 52
>
> # number of observations
> nb_obs <- nrow(mat_cl_prices)
>
> # instantiate an empty matrix to store results
> moving_max <- xts(matrix(NA, nrow = nb_obs, ncol = 4,
+                          dimnames = list(index(mat_cl_prices),
+                                          colnames(mat_cl_prices))),
+                   index(mat_cl_prices))
>
> # we have to compute the first iteration apart
> moving_max[window_size, ] <- apply(mat_cl_prices[1:window_size], 2, max)
> # find the maximum for the moving window one week ahead
>
> for (i in 1:(nb_obs - window_size)) {
+   # take the maximum of the window discarding the first obs and adding the next one
+   moving_max[window_size + i, ] <- apply(mat_cl_prices[(i+1):(window_size + i), ], 2, max)
+ }
>
> # matrix of draw-downs for all ETFs
> mat_dd<- (mat_cl_prices / moving_max) - 1
> print(mat_dd[1:5, ])
           IXC IDV SHY GSPC
2007-06-22  NA  NA  NA   NA
2007-06-29  NA  NA  NA   NA
2007-07-06  NA  NA  NA   NA
2007-07-13  NA  NA  NA   NA
2007-07-20  NA  NA  NA   NA
> print(mat_dd[window_size:(window_size+5)])
                 IXC        IDV         SHY       GSPC
2009-06-12 -0.3564193 -0.5189056 -0.01931465 -0.3941542
2009-06-19 -0.3973240 -0.5373361 -0.01754806 -0.4101486
2009-06-26 -0.4212169 -0.5339160 -0.01401487 -0.4116404
2009-07-02 -0.4311564 -0.5453164 -0.01342606 -0.4260341
2009-07-10 -0.4556228 -0.5570967 -0.01118838 -0.4371046
2009-07-17 -0.4105129 -0.5234657 -0.01366157 -0.3978871
```

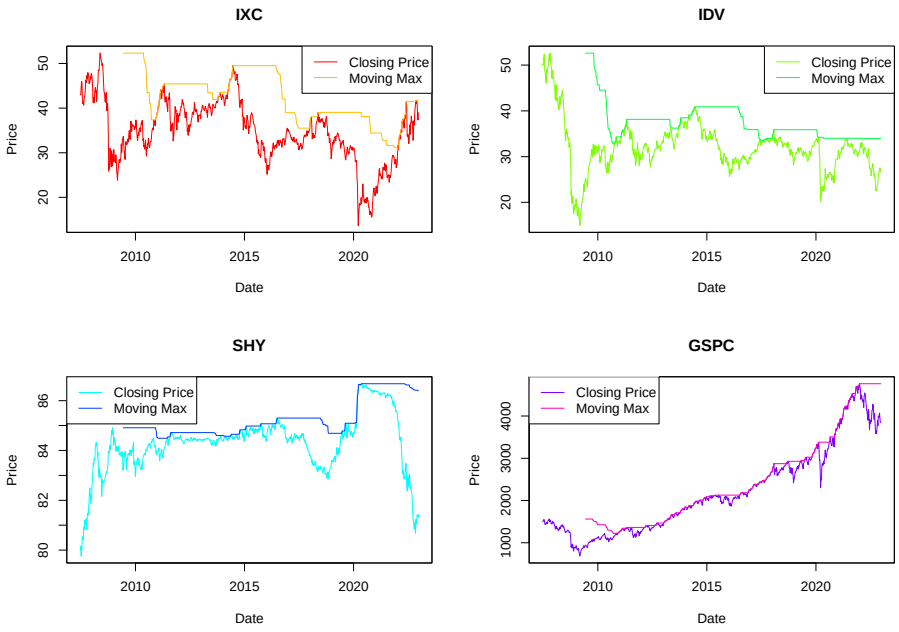
Hence, the first 2 years of data have NA values, but we do it so as to highlight the delay of the rolling window drawdown where the first observation appears only at the 104th week.

From our matrix of drawdowns, we can plot the underwater curve, namely the drawdowns' time-series:



As we are using the first 2 years of our data from 2007, the first drawdown appears in 2009. Moreover, we can argue that IXC has the riskiest drawdown profile, and it achieves the worst drawdown during the covid 19 crisis reaching a more than 60% drawdown. By contrast, SHY has a very low risk profile in terms of drawdowns, yet since the last year it achieves its highest ones. Furthermore, the second riskiest one seems to be IDV followed by GSPC as it has larger drawdowns, especially during the subprime crisis and in 2016. Based on these observations, we can expect the following relative order between our investment in terms of risk profiles from the riskiest to the lowest, namely IXC, IDV, GSPC, and SHY.

By the way, we can support our argument about the moving window by plotting the cumulative moving maximums on top of the closing prices and expect to see the moving maximums beginning a few steps after the closing prices.



Indeed, we see much clearly the delay of the moving max series compared to the closing price series. Besides, we see that the cumulative moving maximums stay constant until the previous maximums are attained, but only in a window of 2 years. Indeed, after that, the moving maximum starts to decline as it discards the highest previous maximum prices. For instance, we can see that since 2020 SHY has reached a peak at 87, but never came back to this level whereas the moving maximum has been starting to decline.

Now, we can move on to the computation of alternative performance risk measures based on the drawdowns we have just calculated.

8.3) Ulcer index and ulcer performance index

Up until now, we have computed risk measures based on the return distribution, but now we are going to consider the drawdown distribution instead. In fact, the drawdown provides insightful information about the cumulative negative returns that an investor would occur which could lead her to cancel its position if she stomachs consecutive severe losses. Consequently, we can consider the ulcer index that like the standard deviation penalizes quadratically all deviations, but instead, focuses on all the drawdowns.

The formula can be obtained as follows and closely resemble the standard deviation's formula:

$$UI = \sqrt{\frac{1}{T}\sum_{t=1}^{T} DD_t^{\,2}} \ ,$$

where $DD_t = \frac{P_t}{\max\limits_{i\in[0,t]} P_i} - 1$ is the drawdown at date t.

Based on this new risk measure, we can hence compute its return adjusted version, namely the ulcer performance index (UPI) that we would try to maximize like any risk adjusted performance measures.

In R, the computation is straightforward, we can compute it by breaking down every step, but before that we must ensure that we remove all missing values so as to not compromise our results:

```
> # -----------
> # ulcer index
> # -----------
>
> # compute the ulcer index
> sum_squared_drawdown <- colSums(mat_drawdown)^2
> ulcer_index <- sqrt(sum_squared_drawdown/nb_obs)
> print(ulcer_index)
      IXC       IDV       SHY      GSPC
4.6863614 3.6803945 0.2315132 1.4142846
> # compute the ulcer performance index
> ulcer_performance_index <- excess_returns / ulcer_index
> print(ulcer_performance_index)
          IXC           IDV           SHY          GSPC
-3.350883e-05 -2.097394e-04  8.736697e-05  8.158886e-04
```

We can note that these results comply with the hypothesis we have made previously based on the drawdown plot, namely the riskiest ETF is IXC followed by IDV, GSPC, and the less risky is SHY.
Therefore, now the best investment strategy is our benchmark the S&P 500 index which achieves the highest UPI, followed by SHY and IDV, whereas IXC is the worst.

8.4) Dar, CDaR, and the average drawdown

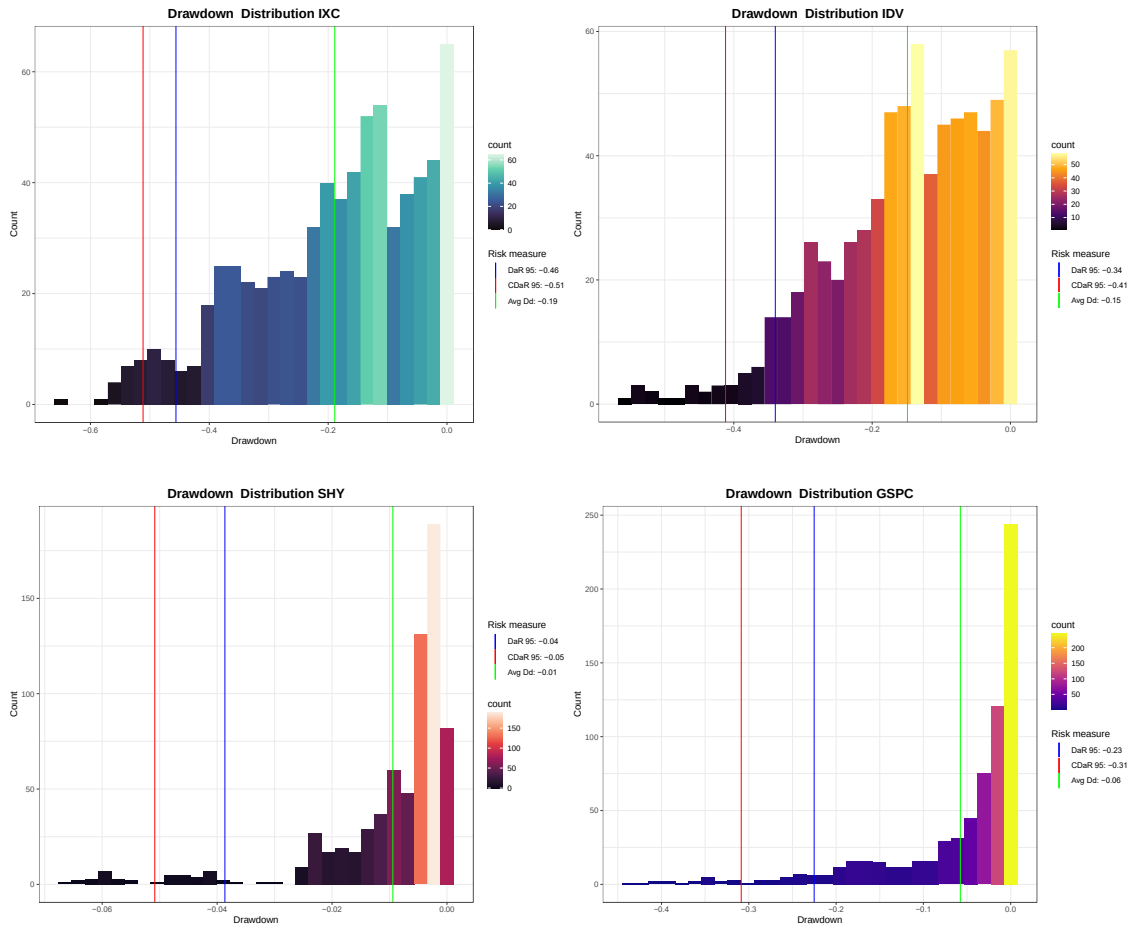
Like the volatility, the ulcer index suffers from the fact that it considers all deviations, here all the drawdowns, and does not penalize largely enough extreme drawdowns. Indeed, tail risk measures are more appropriate to account for the high deviations in the tails.
Fortunately, we can tap into the power provided by classic tail risk measures, namely the VaR and CVaR but using the drawdown distribution instead.

Therefore, we can utilize the drawdown at risk (DaR) and the conditional drawdown at risk (CDaR) to compute risk-adjusted performance measures.
Finally, we can compute the average drawdown that puts equal penalty for each drawdown. Consequently, the computations are similar to those of the VaR and CVaR, and the average drawdown is simply the average of those drawdowns:

```
> # -----------------
> # draw-down at risk
> # -----------------
>
> # sort drawdowns in ascending order
> sorted_drawdown <- apply(mat_drawdown, 2, sort)
> # extract the index of the largest drawdowns
> idx_k_largest_drawdown <- cov_rate * nb_obs
> # extract that particular drawdown, namely the DaR
> DaR_95 <- sorted_drawdown[idx_k_largest_drawdown, ]
> print(DaR_95)
        IXC         IDV         SHY        GSPC
-0.45619096 -0.33770329 -0.03636147 -0.22510962
>
>
> # ----------------------------
> # conditional draw-down at risk
> # ----------------------------
>
> # compute the mean of the largest drawdowns beyond the DaR
> CDaR_95 <- colMeans(sorted_drawdown[1:idx_k_largest_drawdown, ])
> print(CDaR_95)
        IXC         IDV         SHY        GSPC
-0.51157191 -0.40745757 -0.05023748 -0.30438532
>
>
> # -----------------
> # average draw-down
> # -----------------
>
> # average drawdown
> avg_drawdown <- colMeans(mat_drawdown)
> print(avg_drawdown)
         IXC          IDV          SHY         GSPC
-0.188918106 -0.148365245 -0.009332835 -0.057013096
```

Here we see that for each risk measures, the order is the same than the ulcer index, namely the ETF from the riskiest to the least risky are IXC, IDC, GSPC, and SHY respectively.
Additionally, we notice that the CDaR is always larger than the DaR which is the same results than for the CVaR and VaR respectively.
Also, the average drawdown is many times lower than the tail risk measures.

One useful plot that we have not presented in the previous visualization section is the display of the drawdown's histograms, but we abstained from presenting them in order to plot on the same plot the corresponding DaR, CDaR and average drawdown so as to represent them graphically:



From these drawdown distributions, we see that they are highly left-skewed (especially for SHY and GSPC) which is logical since the drawdown is by definition less or equal to 0. In addition, IXC and IDV have the highest risk profile based on the drawdowns with extremes drawdowns more frequent than for SHY and GSPC. Indeed, GSPC and SHY have most of their observations in the first buckets on the left, whereas IXC and IDV have only a few. Furthermore, IDV's higher count is achieved at -10% while the others are realized at 0%. The average drawdown seems to be far away from the DaR and CDaR but presents the advantage of considering all the drawdowns but is not focused on extreme risk which is its main drawback. In addition, the CDaR is very large compared to the DaR for SHY and GSPC, highlighting the non-robustness of this method as it uses only the average of drawdowns larger than the DaR which are very few. Therefore, combining extreme and average drawdown risk as a penalized risk metric could allow to leverage the pros of both methods and is going to be presented in the next section.

8.4) Serenity ratio

In the previous section, we have seen both the pros and cons of average as well as extreme risk measures but one more step for the extreme risk measure is required. Indeed, the CDaR would matter only relative to the amount of risk an investor has agreed to take on. To this regard, as the standard risk measure used is the standard deviation, we can compare the CDaR to the volatility and define the pitfall indicator. The name of this risk measure has not been choosing randomly, as it refers to an investor no falling in a pitfall, thus ensuring the extreme risk provide by the CDaR is not to large relative to the standard deviation.

Therefore, the pitfall indicator is defined as the CDaR divided by the standard deviation, and accordingly an investor would seek to minimize this ratio. The formula is provided below:

$$PI(\alpha) = \frac{CDaR(\alpha)}{Vol}$$

Consequently, we can use a penalized risk metric that combines the “average loss”, namely the Ulcer index along with the “extreme risk penalty”, namely the pitfall indicator:

$$\text{Penalized Risk} = \text{Ulcer} \times \text{Pitfall}$$

Finally, we can harness this penalized risk metric so as to compute the Serenity Ratio, and the higher, the more serene the investor will be as she will expect a less extreme risk than the one she has agreed to take on.

$$\text{Serenity Ratio} = \frac{\text{Return}}{\text{Penalized Risk}}$$

The R code for these computations is provided as follows:

```
> # -------------
> # pitfall index
> # -------------
>
> # annual standard deviation
> ann_std <- apply(mat_returns, 2, sd) * sqrt(52)
> # the pitfall index as an extreme risk measure
> pitfall_95 <- (- CDaR_95) / ann_std
> print(pitfall_95)
     IXC      IDV      SHY     GSPC
1.793616 1.743887 3.745090 1.593649
>
>
> # --------------
> # penalized risk
> # --------------
>
> # penalized risk measure
> penalized_risk <- ulcer_idx * pitfall_95
> print(penalized_risk)
       IXC        IDV        SHY       GSPC
0.42516656 0.32316754 0.05653784 0.15946382
>
>
> # --------------
> # serenity ratio
> # --------------
>
> # the serenity ratio as risk-adjusted performance measure
> ann_returns <- colMeans(mat_returns) * 52
> serenity_ratio <- ann_returns / penalized_risk
> print(serenity_ratio)
        IXC         IDV         SHY        GSPC
-0.01884724 -0.12229491  0.01902245  0.37635923
```

IXC has a pitfall indicator of 1.79, indicating that its CDaR is 1.79 times larger than its volatility which enables it to reach the 2nd position for this risk metric. However, things get worse when we focus on the penalized risk measure as it has the largest one indicating that is the riskiest investment compared to the others. Despite, its expected return is relatively high, so its serenity ratio gets better compared to IDV which allows it to reach the third position in terms of serenity ratio.

IDV only has a 1.74 pitfall indicator so in terms of this risk metric it attains the second position behind GSPC. Nonetheless, in terms of penalized risk, IDV reaches the third position behind GSPC and SHY. Finally, it displays the lowest serenity ratio, so it is the worst investment to consider compared to the other ones.

On the one hand, we note that SHY achieves the highest pitfall indicator, namely its CDaR is 3.7 higher than its standard deviation, indicating a higher risk of being trap in a drawdown. Indeed, we have seen that SHY has a very low volatility but display a significant CDaR of 5%. Nevertheless, as it achieves relatively low drawdowns, its penalized risk measure is therefore the lowest one, indicating that it is the less risky investment to consider. Finally, as it displays a positive expected return, its serenity ratio despite very low is positive which allows it to attain the 2nd position.

Our benchmark is the one achieving the lowest pitfall indicator with a CVaR which is only 1.59 larger than its annualized volatility. However, as it achieves larger drawdowns than SHY, it only achieves the 2nd position for the penalized risk measure as it has the second lowest one. Despite the last statement, it achieves the highest serenity ratio so we can assert that the best performing investment is GSPC and that none of our ETFs outperform our benchmark.

Nevertheless, several remarks can be added concerning this conclusion. Firstly, we have several investment opportunities, but we have considered them separately, so we have neglected the diversification principle. Additionally, as we have seen in the first section, the ETFs have very low and even negative correlation, so we could have obtained a lower risk.

9) Conclusion

A summary table is provided in order to resume all risk-adjusted performance measures so as to rank the best ones and counting for each ones the best. However, we must remember that the serenity ratio has been computed using annualized means and standard deviations so we cannot compare it directly with the other weekly measures:

	IXC	***IDV***	***SHY***	***GSPC***
Sharpe ratio	*-0.002*	*-0.021*	*0.013*	*0.045*
Smart Sharpe ratio	*-0.002*	*-0.021*	*0.013*	*0.047*
Sharpe-VaR (Gauss)	*-0.001*	*-0.012*	*0.008*	*0.028*
Sharpe-VaR (Hist)	*-0.001*	*-0.013*	*0.008*	*0.028*
Sharpe-VaR (CF)	*-0.001*	*-0.012*	*0.007*	*0.026*
Sharpe-CVaR (Gauss)	*-0.001*	*-0.011*	*0.006*	*0.021*

Sharpe-CVaR (Hist)	*-0.001*	*-0.008*	*0.005*	*0.018*
Sharpe-CVaR (CF)	*-0.0008*	*-0.005*	*0.004*	*0.012*
UPI	*-0.0005*	*-0.004*	*0.001*	*0.011*
Serenity ratio (annualized)	*-0.010*	*0.107*	*0.022*	*0.382*

Here, we see that the S&P 500 benchmark outperforms largely all our ETF, even though we could have achieved better results by considering a portfolio of these ETF as some of them are low and negatively correlated.