Pamukkale Üniversitesi

Sosyal Bilimler Enstitüsü Dergisi

Pamukkale University Journal of Social Sciences Institute

ISSN1308-2922 EISSN2147-6985

Article Info/Makale Bilgisi
√Received/Geliş:20.03.2022 **√Accepted/Kabul:**27.05.2022
DOİ:10.30794/pausbed.1090560
Research Article/Araştırma Makalesi

İlbasmış M. (2022). "Asset Allocation with Dynamic Conditional Correlations (Dcc) Model: an Implementation in The R Program", *Pamukkale University Journal of Social Sciences Institute, 2022 Issue 51: Special issue 1*, Denizli, ss. Ö149-Ö175.

ASSET ALLOCATION WITH DYNAMIC CONDITIONAL CORRELATIONS (DCC) MODEL: AN IMPLEMENTATION IN THE R PROGRAM

Metin İLBASMIŞ*

Abstract

This study demonstrates how to use the R programming language to estimate time varying volatility in returns and correlations between several asset classes by employing a model called Dynamic Conditional Correlations (DCC) and to form portfolios using those estimates. A number of user-written R commands are presented in the study, designed for practitioners, academics, and students of finance interested in active portfolio optimization. The study uses these commands to access financial data, analyze statistical characteristics of the data, estimate dynamic correlations, and finally compute the optimal weights of several asset classes in portfolios optimized for a variety of purposes.

Keywords: *R program, Estimation, DCC, Correlations, Portfolio optimization.*

JEL Classification: C22, C51, G11, G13, G15

DİNAMiK KOŞULLU KORELASYONLAR (DCC) MODELİ İLE VARLIK TAHSİSİ: R PROGRAMINDA BİR UYGULAMA

Öz

Bu çalışma, dinamik koşullu korelasyon (DCC) modeli kullanarak varlık sınıfları arasındaki zamanla değişen korelasyonların tahmini ve bu tahminleri kullanarak portföy oluşumu sürecini R programının kullanımıyla sunmaktadır. Portföy optimizasyonu ile ilgilenen yatırımcılar, akademisyenler ve finans öğrencileri için tasar- lanan çalışmada, kullanıcı tarafından yazılan bir dizi R komutu sunulmaktadır. Bu komutlar, finansal verilere erişmek, verilerin istatistiksel özelliklerini analiz etmek, dinamik korelasyonları tahmin etmek ve son olarak çeşitli amaçlar için optimize edilmiş portföylerdeki çeşitli varlık sınıflarının optimal ağırlıklarını hesaplamak için kullanılmıştır.

Anahtar Kelimeler: *R programı, Tahmin, DCC, Korelasyonlar, Portföy optimizasyonu.*

JEL Kodları: *C22, C51, G11, G13, G15*

*Dr., Aksaray University, Faculty of Economics and Administrative Sciences, AKSARAY.
e-posta: metinilbasmis@aksaray.edu.tr, (https://orcid.org/0000-0001-9657-4604)

1.Introduction

In this study, the implementation of a dynamic portfolio allocation scheme is presented that relies on time- varying variances and covariances described by Engle (2002)'s dynamic conditional correlation model (DCC). The DCC model is a popular technique for discovering relationships between different markets or asset classes (Chong et al., 2009; Cappiello et al., 2006; Yang et al., 2012). The approach taken in the current study involves estimating the conditional standard deviation for each market based on an asymmetric GARCH framework. The Asymmetric DCC model is then used to estimate the time-varying conditional correlations between two asset classes. The study is conducted using a statistical and graphical programming language R-project (R Core Team, 2021), available at https://www.r-project.org/. The use of R program for the estimation of time varying correlations and portfolio formation utilizing these estimates are the two steps of this study. A number of user-written R commands are presented in the study, designed for practitioners, academics, and students of finance interested in asset allocation and portfolio optimization problems. The study uses these commands to access financial data, analyze statistical characteristics of the data, estimate dynamic correlations and covariances, and finally estimate the optimal weights of several asset classes in portfolios optimized for a variety of purposes.

The empirical application in this study assumes a specific case; an investor who wishes to diversify their portfolios by investing in stocks, gold, crude oil, Bitcoin, and a riskless asset. Diversification is defined as being on the efficient frontier of all possible portfolios. Whether the investor is risk-averse or risk-loving, the portfolio they choose is on the efficient frontier. Diversification is defined as being on the efficient frontier of all possible portfolios. Whether the investor is risk-averse or risk-loving, the portfolio they choose is on the efficient frontier. The study will describe the model, generate equations for estimations, and provide R commands for each estimation and calculation.

The asset classes selected for this paper are intended to offer perspective on whether including alternative assets in stock market portfolios can increase diversification (Gorton & Rouwenhorst, 2006; Hillier et al., 2006; Conover et al., 2010; Gao & Nardari, 2018; Liu et al., 2018), including incorporating Bitcoin into a portfolio (Briere et al., 2015; Klein et al., 2018). In a similar manner to this study, (Bouri et al., 2017; Platanakis & Urquhart, 2020) employ a DCC model to determine assets weights in dynamic portfolio allocations.

A growing body of finance literature uses R program in their analysis (e.g., Ardia et al., 2017; Chen & Nie, 2018; Ensor & Koev, 2014; Mullen, 2014). While several powerful statistical software are available to researchers estimating dynamic volatility or correlations, such as SPSS, Stata, SAS, MATLAB, EViews, MS Excel; R is one of the strongest tools for data manipulation, visualization, and statistical analysis (NYU-Libraries, 2022) and it is a free and open-source software package.[1] R was created by Ross Ihaka and Robert Gentleman, GNU public license, the University of Auckland, New Zealand, 1992, as an implementation of the S language, which is a statistical programming language, by John Chambers, Bell Laboratories, New Jersey, 1975−1976. Today, R is being developed by the R Development Core Team (R Core Team, 2021). As of March 2021, R has 18970 packages. Statisticians, data miners and investors have been increasingly using R for their empirical analysis in recent years.

The study will demonstrate how portfolio allocation can empirically be conducted using the R program, accessed through RStudio (RStudio Team, 2016).[2] The R programming language provides several implementations of Garch-type univariate models, such as the **Time Series Analysis** (Hyndman,

[1] Readers can download the latest version of R from the CRAN home page https://cloud.r-project.org/. Those who already have R installed on their systems are encouraged to update the R program by running the following commands in their R console in the listed order *"install.packages(^{t}devtools^t), library(devtools), install_github(^{t}andreacirilloac/updateR^t), library(updateR), updateR()"* and packages on R by running *"update.packages(checkBuilt = TRUE)"*.

[2] A free version of RStudio is available to install at https://www.rstudio.com/products/rstudio/download/.

2022), and **Empirical Finance** (Eddelbuettel, 2022) task views in CRAN Task Views (Zeileis, 2005) at https://cran.r- project.org/web/views/: **quantmod** (Ryan & Ulrich, 2020), **rugarch** (Ghalanos, 2022), **rmgarch** (Galanos, 2022), **PerformanceAnalytics** (Peterson & Carl, 2020), **moments** (Komsta & Novomestky, 2015), **tseries** (Trapletti & Hornik, 2021), **forecast** (Hyndman & Khandakar, 2008), and **portfolio.r** (Zivot, 2008).[3]

The remainder of this article is arranged as follows. Section 2 presents a brief review of the empirical model. Empirical applications including the data description is given in Section 3 along with the empirical results. The study is concluded in Section 4.

2.Empirical model

The empirical model adopted in this study will be briefly reviewed in this section. The first part of the discussion presents the mean-variance optimization problem, followed by the time-varying variance-covariance matrix that will be used in the optimization problem.

2.1.Mean-variance portfolio optimization

Investors optimize portfolio returns by comparing their mean and variance in Markowitz (1952)'s mean- variance portfolio optimization framework. In order to compute portfolio weights (x), the sample mean (μ) and covariance matrix (Σ) parameters are used to maximize the following quadratic utility function with respect to x:

$$U = x^T\mu - \frac{\lambda}{2}x^T\Sigma x \qquad (1)$$

where λ represents the level of risk aversion of the investor. To be consistent with reality, it is assumed that short-selling is possible ($x_i \geq 0, \forall i$) or ($x_i \leq 0, \forall i$) and portfolio weights are normally distributed ($\sum_i^N x_i = 1$), which means that the investor can invest her own and borrowed wealth in the portfolio, and she will invest it all in the asset classes listed in the optimization problem. The classic portfolio optimization equation is therefore as follows:

$$\underbrace{max}_{x} \left\{x^T\mu - \frac{\lambda}{2}x^T\Sigma x\right\} \qquad (2)$$

$$\sum_i^N x_i = 1$$

The mean-variance portfolio optimization in the Markowitz framework typically assumes that the covariance matrix is constant, in which case the investor uses a static optimization model, looking one period ahead. Investment optimization decisions can also be made using optimization models that look into multiple future time periods. These models are known as dynamic optimization models. Unlike a static model, a dynamic model assumes that the variance-covariance matrix is time-varying, so portfolio weights must be adjusted based on the changes in the optimization parameters. By estimating time-varying correlations using the DCC model, this study assumes a time-dependent matrix of variance-covariance (Σ). With this assumption, it is possible to calculate the optimal weights for an efficient portfolio that is updated in real time.

2.2.The asymmetric DCC model

Engle (2002) merits credit for establishing the basic DCC model. Several versions of the model have been developed since then to capture different aspects of dynamic volatility and correlations (Cappiello et al., 2006; Hafner & Franses, 2003; Billio et al., 2006; McAleer et al., 2008; Glosten et al., 1993). Based on previous literature, this study uses the time-varying correlations between asset pairs derived from the

[3] It is worth mentioning, for those of you who use MATLAB frequently, that Professor Kevin Sheppard from Oxford University developed an excellent MATLAB toolbox called MFE Toolbox. There are several MATLAB functions available for analysis of financial time series, including GARCH and DCC estimations.

asymmetric DCC-GJR-GARCH method proposed by Cappiello et al. (2006), assuming that factors such as negative market shocks influence the variance-covariance process.

Optimal lags ARMA(p,q) & GARCH(1,1)- DCC(1,1) with asymmetric effects:

Using ***auto.arima*** function from the ***forecast*** library, a commonly-used R function, the process of identify- ing the most optimal ARIMA model parameters, the auto-ARIMA algorithm selecting a model, is performed and all return series are found to be an ARMA(0,0) process based on BIC information criteria.[4] Thus, the mean equation, presented in Eq. 3, states that the return time series follows a mean and a zero-mean white noise process. There is only one coefficient to estimate in the mean equation: μ_0, which is used to calculate the zero-mean white-noises, residuals. The univariate GARCH process in Equation 5 is then calculated using these residuals.

$$r_{i,t} = \mu_{i,0} + \epsilon_{i,t}, \qquad \epsilon_{i,t} \sim \mathrm{N}(0, \sigma_{i,t}) \tag{3}$$

$$\epsilon_{i,t} = r_{i,t} - \mu_{i,0} = \sqrt{\sigma_{i,t}}\nu_{i,t} \tag{4}$$

$$\sigma_{i,t} = \omega_i + \epsilon_{i,t-1}^2\alpha_i + \sigma_{i,t}\beta_i + \epsilon_{i,t-1}^2 I_{i,t-1}\theta_i \tag{5}$$

The volatility is defined in Equation 5. The time-dependent variance is $\sigma_{i,t}$, the residuals from the mean equation for asset *i* at time *t* is $E_{i,t}$, and excess returns are represented by $r_{i,t}$ (Hereafter, excess return is referred to as return.) *i* is the returns on the index, gold, crude-oil, or Bitcoin prices. Shocks' asymmetric effects are captured by parameter ϑ_i. The indicator function $I_{i,t-1}$ takes the value 1 when residuals are negative and 0 when they are not. The coefficients of the GARCH model are required to be non-negative values: a constant ω_i, a parameter of previous errors from the mean model α_i and the previous time-varying variance parameter β_i. A stationary process requires α_i and β_i to be less than one.

The asymmetric DCC(1,1) model given in Equation 7 is then employed to estimate the dynamic correlations by using the time-dependent variances calculated above.

The DCC model begins by defining a diagonal matrix of conditional volatilities: D_t:

$$D_t = diag\{\sqrt{\sigma_{1,t}}, \sqrt{\sigma_{2,t}}, \sqrt{\sigma_{3,t}}, \dots \sqrt{\sigma_{n,t}}\}$$

Next, the model defines a matrix of conditional correlations, R_t. Thus, being a combination of volatility and correlation, the matrix of conditional variance-covariance can be defined as follows:

$$H_t = D_t R_t D_t$$

Denoting the standardized residuals by $\upsilon_{i,t}$:

$$\nu_{i,t} = \frac{\epsilon_{i,t}}{\sqrt{\sigma_{i,t}}} = \epsilon_{i,t} D_{i,t}^{-1} \tag{6}$$

where the standardized residuals have a unit variance and are correlated. Following Cappiello et al. (2006)'s specifications:

$$\sigma_{ij,t} = \bar{\sigma}_{ij}(1 - (a + b + g)) + \nu_{i,t-1}\nu_{i,t-1}a + \sigma_{ij,t-1}b + \nu_{i,t-1}\nu_{i,t-1}I_{ij,t}g \tag{7}$$

where $\sigma_{ij,t}$ is time-varying covariance process, σ_{ij} is the asset *j* constant correlation with assets *i*.

Negative news negatively impacts conditional correlations in the form of asymmetric effect parameter *g*, past shocks are accounted for in parameter *a* and parameter *b* indicates how historical covariance affects the present covariance. The restriction of $a + b < 1$ is implied by the men-reverting process and non-negativity is ensured by coefficients *a* and *b*. As the sum of the two coefficients

[4] AIC information criteria results in different ARMA(p, q) processes for each of the return series in this study. While this study chooses BIC-based lags, users are referred to Section 3.4 for alternative information criteria that can be used to determine the optimal lags.

approaches to 1, the correlation process becomes more persistent.

Conditional correlations, $\rho ij,t$, can be calculated from the time-dependent variances generated in phase one and from the time-dependent covariance generated in phase two:

$$\rho_{ij,t} = \frac{\sigma_{ij,t}}{\sqrt{\sigma_{i,t}}\sqrt{\sigma_{j,t}}}$$

3.Empirical application and results

The R-procedure to allocate assets based on the matrix of variance-covariance estimated by the DCC model is presented in this section. Since the study intends for even a beginner practitioner of investment to be able to reproduce the results, it will start with acquiring financial data, preparing the data and running the model step-by-step along with all R commands.

3.1.Preparing the data

The study collects daily data on Dow Jones 30 Industrial Average index (DJI) prices, data on an ETF tracking the price of gold bullion in the OTC (over-the-counter) market (SPDR Gold Shares, GLD), crude oil shares prices (CL=F), and the Bitcoin (BTCUSD) data from Yahoo Finance using R's **quantmod** library, while 1-year Treasury bill rate (DTB1YR) data from the St. Louis FedâĂŹs FRED database. Readers who would like to produce the coefficient estimates and graphics in this study can run the R commands given in the presented order, unless otherwise asked.

Financial data can be loaded into R in several ways. R's ***quantmod*** and ***tidyquant*** packages are the most popular ways to do it. The ***quantmod*** package provides tools for modelling financial data and a trading framework in R. The current study uses this package to acquire financial data from its source. In order to utilize the functions of this package, first, the package needs to be installed and added to your R library by running the following code in your R console:[5]

```
install.packages("quantmod");
library(quantmod);
```

Once, the package is added to your R library, you can now use the ***getSymbols*** function of this package to install the desired data to your R environment. This study runs the following commands to download the dataset used in the analysis;[6]

While Yahoo Finance is the default source for the data, the ***quantmod*** package is capable of downloading data from multiple sources, including: Yahoo, MySQL, FRED, CSV, RData, and Oanda.[7]

To combine all time series in one object, run the following commands once all five time series have been installed in your R environment.[8]

[5] When a semicolon is placed at the end of an R-command line, the command has been completed. Having no semicolon at the end of a line means that the command continues in the following line.
[6] There are only three types of names that are acceptable in R: letters, numbers, and dot or underline characters. Once the Bitcoin data has been imported, you will notice that the names of the columns contain the symbol "−", which is not a valid column name in R. Therefore, this issue must be resolved. Using the following command, the name of the data is replaced for convenience: BTCUSD = 'BTC-USD'; names(BTCUSD) = make.names(names(BTCUSD), unique=TRUE); rm('BTC-USD').
[7] Google Finance was also used as a source of data for this function. As of March 2018, it stopped providing data.
[8] A command line starting with the symbol # is a deactivated one. To activate this line, simply delete the symbol # from the beginning of R command line. The use of two symbols together (##) indicates the text is a simple explanation of the code.

```
getSymbols(Symbols = "^DJI");
getSymbols(Symbols = "GLD");
getSymbols(Symbols = "CL=F");
getSymbols(Symbols = "BTC-USD");
getSymbols(Symbols = "DTB1YR",src="FRED");
```

```
mydata = merge(DJI$DJI.Adjusted, GLD$GLD.Adjusted, CLF$CL.F.Adjusted,
BTCUSD$BTC.USD.Adjusted, DTB1YR);
colnames(mydata) = c("DJI","GLD","CLF","BTCUSD","DTB1YR");
mydata = window(mydata, start ="2015-01-01" , end ="2020-01-01" );
mydata = mydata[complete.cases(mydata), ];
prices = cbind(mydata[,"DJI"],mydata[,"GLD"],mydata[,"CLF"],
mydata[,"BTCUSD"]);
## For weekly portfolio optimization:
# prices = apply.weekly(prices, FUN=last);
## For monthly portfolio optimization:
# prices = apply.monthly(prices, FUN=last);
```

The sample period of this study covers data from the beginning of 2015 to the end of 2019 using the **window**

function and all rows of data including at least one missing data point are removed from the data.[9]

Note that all the commands given in this study make daily estimates of the time-varying variance, covariance, and correlation processes. As a result, dynamic portfolios are rebalanced every day. For weekly or monthly estimations, the data must be transformed into a weekly or monthly frequency. To convert the data into a weekly or monthly frequency, use the last two lines of commands given above.

3.2.The object of interest

The main task of financial econometrics is to estimate the parameters of a well-defined probability model that describes financial time series. Asset prices and interest rates are among the most important financial time series. In financial literature, asset prices are mainly studied as returns. A wide range of return definitions exist in practice. Considering a time series of closing prices, this study uses logarithmic returns, $r_{i,t} = ln(P_{i,t}/P_{i,t-1}) \times 100$. Logarithmic returns are usually preferred by financial econometricians because they have superior properties compared to arithmetic returns. Prices also follow a log-normal distribution when we assume a normal distribution for returns. Log-normal distributions guarantee that prices will be positive. Secondly, closed-form solutions to pricing formulas are usually derived from logarithmic returns.

The choice of data representation (object class) is the starting point for any time series analysis in R, which can deal with a variety of data types. Picking out the right object class in R is crucial because it determines which functions will be available for analyzing the data. In the ***getSymbol*** function, prices are stored as ***xts*** and ***zoo*** objects, which are the appropriate data storage types for a time series analysis.

[9] Taking a close look at the data, it is apparent that while index, gold, and crude oil prices are not available on weekends, Bitcoin prices are available seven days a week, so prices are available even on weekends. This makes the data unbalanced. In order to carry out the empirical analysis of this study, the original data is corrected by removing missing data points, which removes the Bitcoin's weekend prices and creates a balance data.

Thus, this study makes use of an object class that is designed for time series data.

In this study, excess returns are used as the basis for the empirical analysis, defined as the return over the riskless rate. Prices are given in daily frequency, so the returns to be calculated using these prices are daily returns. The riskless rate is given as an annual rate, so it needs to be converted to a daily rate. Run the following commands to calculate log returns, defined as the difference between log of prices, daily riskless rate and excess returns.

```
diff.log.prices = diff(log(prices));
RF = (((mydata[,"DTB1YR"]/100)+1)^(1/251)-1);
returns = Return.excess(diff.log.prices,mean(RF, na.rm=TRUE));
```

3.3.Approaching the dataset

In order to analyze the characteristics of data for the return series, the study first produces a chart of the price and return series, Figure 1, using the following commands. *i* is equal to *"DJI"* for the index, *"GLD"* for gold, *"CLF"* for crude oil, and *"BTCUSD"* for Bitcoin in the commands below.[10] [11]

```
i = "DJI";
plot(as.zoo(prices[,i]));
par(new = TRUE);
plot(as.zoo(returns[,i]));
```

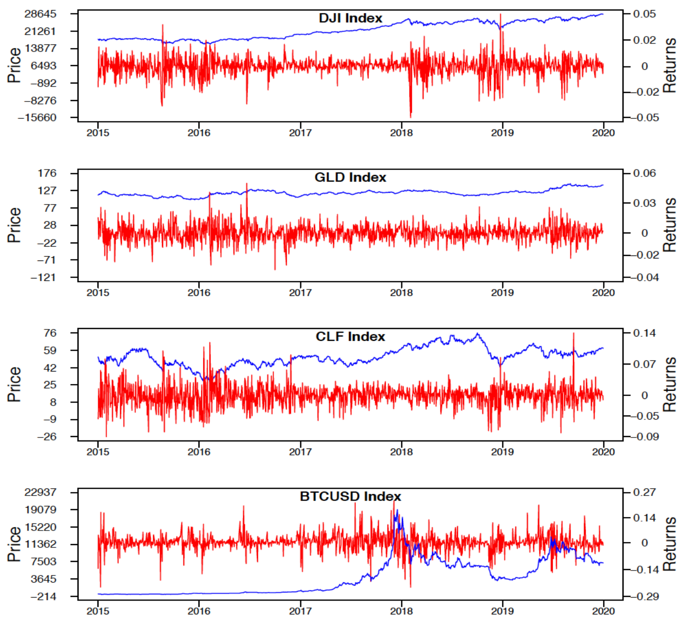

Figure 1: Prices and Returns Over-time

[10] "*i*" is equal to either "DJI", "GLD", "CLF", or "BTCUSD" through the study and this line of code needs to be updated accordingly.

[11] The configuration of plots is left to the user's taste. In your R console, type help(plot) to see plot options. All commands necessary to produce the figures of this study are available upon request from the author.

Figure 1 shows the closing prices (given in blue) and returns (given in red) of the four asset classes on a daily basis from January 1, 2015 to December 31, 2019. As shown in the figure, return series exhibit volatility clustering at both similar and dissimilar times.

3.4.Testing for normality, stationarity, and autocorrelation

Descriptive statistics are documented in Table 1 as well as Pearson correlations. The DJI index, gold, crude oil and Bitcoin prices have positive average returns as can be seen from the table. For the period understudy, Bitcoin is the asset with the highest standard deviation while gold is the lowest. According to Table 1, the DJI index and Bitcoin return series exhibit non-normality with negative skewness, and all return series exhibit excess kurtosis. Jarque-Bera statistics also confirm the non-normality of the return series. The correlation matrix is presented in Panel B of Table 1. Correlations between the DJI index and gold prices are negative and significant, while correlations between the DJI index and crude oil prices are positive and significant. All other correlations between asset pairs are insignificant.

Table 1: Descriptive Statistics & Unit Root Tests

The table presents descriptive statistics in Panel A, Pearson correlation matrix in Panel B. DJI is the Dow Jones 30 Industrial Average index return; GLD is SPDR Gold Shares price return, an ETF tracking the price of gold bullion in the OTC market; CLF is crude oil price return, and BTCUSD is the Bitcoin price return. Data is collected from Yahoo in daily frequency and spans from January 1, 2015 to December 31, 2019.

	DJI	GLD	CLF	BTCUSD
A: Descriptive Statistics				
Mean (%)	0.0515	0.0131	0.0068	0.2457
Median (%)	0.0515	0.0256	0.1272	0.2262
Maximum (%)	4.8593	4.7824	13.6894	22.5069
Minimum (%)	-4.7193	-3.5378	-9.0752	-23.8790
Std. Dev. (%)	0.8583	0.7953	2.4017	4.5787
Skewness	-0.5447	0.1580	0.1542	-0.1692
Kurtosis	6.8100	5.4236	5.5714	7.7955
Jarque-Bera	816.58	310.64	348.78	1201.78
p-value	[0.0000]	[0.0000]	[0.0000]	[0.0000]
N	1248	1248	1248	1248
B: Pearson Correlations				
GLD	-0.18***			
CLF	0.32***	0.02		
BTCUSD	0.03	0.04	-0.02	

Table 2 reports test statistics and their p-values for stationarity and autocorrelation tests. All unit root tests in Panel A of Table 2, Augmented Dickey-Fuller (ADF), Philipps-Perron (PP) and Kwiatkowski-Phillips-Schmidt-Shin (KPSS), indicate that all returns series are stationary during the sample period and can be used directly for analysis without requiring further transformation. In Panels B and C of the table, tests of the Ljung-Box (LB) hypothesize for the presence of autocorrelation in both returns and squared returns up to a specified lag indicate that, although the null hypothesis of no autocorrelation cannot be rejected for returns, it can be rejected for squared returns. LB test on residuals is employed to detect autocorrelation in time series, whereas LB test on squared residuals is used to detect volatility clustering. According to these results, there is an indication of volatility clustering in return series according to Panels B and C of Table 2.

Overall, the table indicates that no further cleaning is required for the returns and that volatility clustering exists in the series, indicating the use of the GARCH model for time-varying volatility processes.

Table 2: Unit Root Tests

The table presents unit root test statistics and p-values in Panel A, and autocorrelations tests in returns and squared returns in Panel B and C, respectively. ADF indicates the Augmented Dickey-Fuller, PP indicates the Philipps-Perron, KPSS indicates the Kwiatkowski-Phillips-Schmidt-Shin unit root tests. DJI is the Dow Jones 30 Industrial Average index return; GLD is SPDR Gold Shares price return, an ETF tracking the price of gold bullion in the over-the-counter (OTC) market; CLF is crude oil price return, and BTCUSD is the Bitcoin price return. Data is collected from Yahoo in daily frequency and spans from January 1, 2015 to December 31, 2019.

	DJI	GLD	CLF	BTCUSD
A: Unit Root Tests Panel				
ADF-statistic -10.8656		-11.2987	-11.2836	-10.0715
p-value	<0.01	<0.01	<0.01	<0.01
PP-statistic	-1200.85	-1313.73	-1370.57	-1268.68
p-value	<0.01	<0.01	<0.01	<0.01
KPSS-statistic	0.0521	0.0423	0.0441	0.1469
p-value	>0.1	>0.1	>0.1	>0.0491
B: Ljung-Box Test Returns (LB)				
LB5-statistic	8.3018	11.9246	17.6820	2.6779
p-value	0.1404	0.0358	0.0034	0.7495
LB10-statistic	14.5013	18.4403	21.2803	9.1001
p.value	0.1513	0.0480	0.0192	0.5226
LB20-statistic	26.8874	30.2081	31.9447	19.8268
p-value	0.1385	0.0666	0.0439	0.4688
C: Ljung-Box Test Squared Returns (LB^2)				
$LB^2$5-statistic	294.6192	23.3595	158.8630	92.6313
p-value	0.0000	0.0003	0.0000	0.0000
$LB^2$10-statistic	397.2684	37.7060	243.2282	145.8640
p-value	0.0000	0.0000	0.0000	0.0000
$LB^2$20-statistic	473.8704	67.1865	347.8581	186.3273
p-value	0.0000	0.0000	0.0000	0.0000

The **mean, median, max, min, sd, skewness, kurtosis, jarque.test, and cor** functions calculate descriptive statistics of assets. The usage of these commands in the current study is given below. The **jarque.test** function belongs to the **moments** package and the stationary tests, ADF, PP, and KPSS, are conducted using **adf.test, pp.test, and kpss.test** functions of **tseries** package. In order to be able to use these functions, they need to be installed first and loaded to the current R session. All functions of the descriptive statistics table are as follows:[12] [13]

[12] Change the mean command to the median function for the median calculation, to the maximum function for the maximum value calculation, to the minimum function for the minimum value calculation, to the standard deviation function for the standard deviation calculation, to the skewness function for the skewness calculation and to the kurtosis function for the kurtosis calculation.
[13] Repeat this command after changing the function to **pp.test** and **kpss.test**, respectively.

```
install.packages("tseries");
library(tseries);
install.packages("moments");
library(moments);
i = "DJI";
Mean[,i] = mean((returns[,names(returns)[i]]));
adf.test((returns[,i]), alternative = c("stationary"));
```

While Tables 1 and 2 conclude that the return series are ready for use in the GJR-GARCH-ADCC model, financial time series tend not to behave as well. Further investigations of autocorrelations in returns (or the residual time series from the mean model, Eq.3) may be necessary. Thus, in addition to the LB tests reported in Table 1, next autocorrelations and partial autocorrelations are examined.

The study allows optimal lags (*p* and *q*) in AR(p) and MA(q) parts of the mean equation to be selected by the Bayesian information criterion (BIC). The R commands presented below find the model of ARMA(p,q) that fits the data best based on BIC, by conducting the differencing test of Kwiatkowski – Phillips – Schmidt – Shin (KPSS).[14] In addition, Augmented Dickey – Fuller (ADF) and Phillips – Perron (PP) tests can be used if necessary.

To find the optimal autocorrelation (*p*) and moving average (*q*) lags in the ARMA(*p*,*q*) model, the **auto.arima** function of the **forecast** package is used. After determining the optimal lags, they are used to compute fitted returns as well as residual time series from the mean model. The residual time series are calculated as the difference between returns and the fitted returns. The residuals from the optimal ARMA(*p*,*q*) model are estimated by running the following commands:

[14] Users can find the model that best fits the data by other information criteria, such as Akaike information criteria by changing the part ic = ”*bic*” to ic = ”*aic*”.

```
install.packages("forecast");
library(forecast);
fit.all = c();
ar.order = c();
ma.order = c();
for (i in 1:NCOL(returns)){
fit = auto.arima(returns[,names(returns)[i]], trace=TRUE, test="kpss",
ic="bic");
ar.order[i] = arimaorder(fit)[1];
ma.order[i] = arimaorder(fit)[3] } ;
arma.order = cbind(ar.order, ma.order);
colnames(arma.order) = c("ar.order","ma.order");
fitted.returns = list();
resids = matrix(NA, nrow = NROW(returns),ncol = NCOL(returns));
for (i in 1:NCOL(returns)){;
fitted.returns[[i]] = arima(returns[,names(returns)[1]],
order=c(arma.order[i,1],0,arma.order[i,2]),method="ML");
resids[,i] = fitted.returns[[i]]$residuals };
resids2[,i] = resids[,i]^2;
```

3.4.1.ACF, PAC, and Heteroskedasticity

The study uses the autocorrelogram (ACF) and partial-autocorrelogram (Partial ACF) charts to visualize the analysis of autocorrelations in residuals and squared residuals from the optimal ARMA(p,q) model, which are shown in Figure 2 and Figure 3. The results in Table 1 are confirmed by both figures.

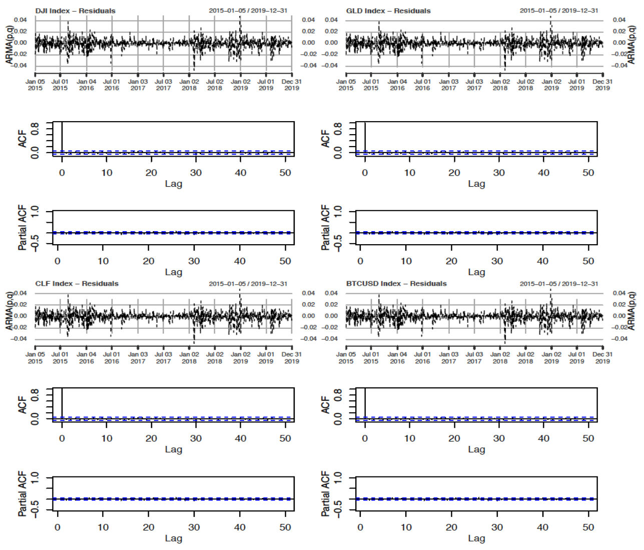

Figure 2: Autocorrelations and Partial Autocorrelations of Residuals

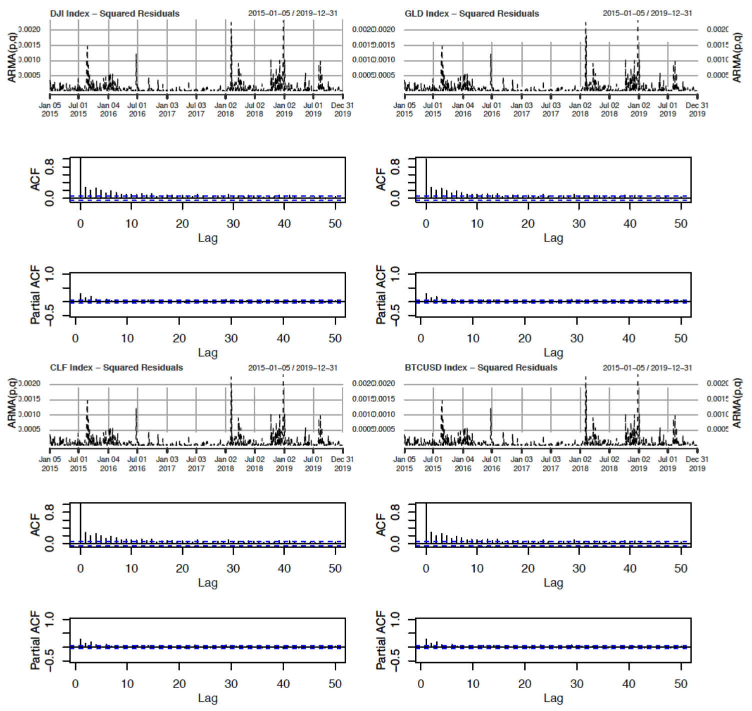

Figure 3: Autocorrelations and Partial Autocorrelations of Squared Residuals

As shown in Figure 2, the residual time series of the mean model are not autocorrelated as there is no peak outside of the statistically insignificant area. However, Figure 3 shows autocorrelation up to 15 lags for all variables. In addition, the figure displays a pattern of squared residuals, which indicates heteroscedasticity. Periods of low volatility are followed by periods of high volatility. Two periods of high volatility can be observed around a period of low volatility, for example. The first period of high volatility covers the end of 2015 and the beginning of 2016. A relatively very low volatility period can be distinguished between the second half of 2016 and the beginning of 2018. During the remainder of the sample period, multiple sub-periods of high volatility with short breaks of low volatility are observed.

Figures 2 and 3 are the results of executing the following commands separately for each of the variables.

```
i = "DJI";
plot(resids[,i]);
acf(resids[,i], main="", lag.max=50,na.action=na.pass);
pacf(resids[,i], main="", lag.max=50,ylim=c(-0.5,1), na.action=na.pass);
plot(resids2[,i]);
acf(resids2[,i], main="", lag.max=50,na.action=na.pass);
pacf(resids2[,i], main="", lag.max=50,ylim=c(-0.5,1), na.action=na.pass);
```

3.5.GJR-GARCH(1,1)-ADCC model

The volatility of each time series is modeled using a GJR-GARCH(1,1) model with a multivariate Student-t distribution, while the mean equation was fitted with an ARMA(0,0) specification. The parameters estimated in this volatility estimation process are then used to calculate standardized residuals, which are then used to estimate the time-varying variance-covariance and correlation processes. Panel A of Table 3 displays the coefficient estimates of GARCH models for the DJI index, gold, crude oil, and Bitcoin price returns. Panel B of the table shows the parameter estimates from the asymmetric DCC model.

The α coefficient in Equation 5 represents the effect of past shocks on the current covariance process, and the coefficient estimates shown in Table 3 indicate that past shocks have a significant (insignificant) impact on the covariance processes of gold and BTC (DJI index and crude oil prices). Equation 5's ϑ coefficient measures the asymmetric impact of negative shocks on the current covariance process. The coefficient estimates indicate that negative shocks have a significant (insignificant) impact on the DJI index and gold prices (Crude oil and BTC).

The parameter β in Equation 5 is used to represent persistence in the volatility process (GARCH parameter), and it is significantly high for all-time series, with gold price having the highest and the DJI index having the lowest persistence in volatility. Asymmetry coefficients are significant and positive in the DJI index and gold returns, but not in crude oil or Bitcoin. Significantly positive asymmetry coefficients indicate that negative market shocks have an increasing effect on volatility.

Equation 7's b parameter represents variance-covariance's lag one effect on the present matrix of variance- covariance, gauging how persistent correlations are. This persistence is significantly high for all correlation processes. As the economy experiences recessions and the market receives negative news, the parameter g demonstrates how the correlations behave. Estimates of g parameter show positive and significant asymmetry effects in the daily correlations between all asset pairs.

Table 3: GARCH and DCC Coefficients Estimates

The table reports the estimates of GARCH model in Panel A, and DCC model in Panel B. DJI is the Dow Jones 30 Industrial Average index return; GLD is SPDR Gold Shares price return, an ETF tracking the price of gold bullion in the over-the-counter (OTC) market; CLF is crude oil price return, and BTCUSD is the Bitcoin price return. Data is collected from *Yahoo* in daily frequency and spans from January 1, 2015 to December 31, 2019.

A: Coefficient Estimates of GARCH Model				
	DJI	Gold	Crude oil	BTC
α	0.0106	0.0274	0.0035	0.1483
	[0.2616]	[0.0000]	[0.8322]	[0.0005]
β	0.7957	0.9783	0.94	0.8085
	[0.0000]	[0.0000]	[0.0000]	[0.0000]
ϑ	0.2716	0.0000	0.0901	0.0000
	[0.0000]	[0.0134]	[0.1163]	[0.8283]
B: Coefficient Estimates of DCC Model				
	a	*b*	*g*	
DJI correlations				
with Gold	0.0448	0.9328	0.0000	
	[0.001]	[0.0000]	[0.0000]	
with Crude oil	0.0064	0.9679	0.0024	
	[0.6094]	[0.0000]	[0.0000]	
with BTC	0.0059	0.9642	0.0006	

	[0.482]	[0.0000]	[0.0000]	
Gold correlations				
with Crude oil	0.0354	0.8417	0.0033	
	[0.5554]	[0.0823]	[0.0823]	
with BTC	0.0107	0.9773	0.0000	
	[0.0752]	[0.0000]	[0.0000]	
Crude oil correlations				
with BTC	0.0000	0.9318	0.0000	
	[1.0000]	[0.0000]	[0.0000]	
Numbers in brackets are *p*-values.				
10%, 5%, and 1% significance levels are denoted by *, **, ***, respectively.				
GARCH: $\sigma_{i,t} = \omega_i + \epsilon^2_{i,t-1}\alpha_i + \sigma_{i,t}\beta_i + \epsilon^2_{i,t-1}I_{i,t-1}\theta_i$				
DCC: $\sigma_{ij,t} = \bar{\sigma}_{ij}\big(1-(a+b+g)\big) + v_{i,t-1}v_{i,t-1}a + \sigma_{ij,t-1}b + v_{i,t-1}v_{i,t-1}I_{ij,t}g$				

Coefficients in Table 3 are estimated with the **rmgarch** package. A number of commands must be used, each defining a parameter for the next step. The **ugarchfit** and **dccfit** commands finally estimate the coefficients of GARCH and DCC processes.[15] The following commands contain details of all steps taken by this paper to estimate the parameters of the DCC(1,1) model with asymmetric effects:

```
## dccdata = returns; install.packages("rmgarch; library(rmgarch);
garch.fit.matrix = list(); dcc.fit.matrix = list();
for (i in 1:NCOL(dccdata)){;
for (j in 1:NCOL(dccdata)){;
modeldata = na.omit(cbind(dccdata[,i], dccdata[,j]));
## remove all rows with non-finite values:
modeldata = modeldata[!rowSums(!is.finite(modeldata)),];
## remove the row if any column contains NA
modeldata = modeldata[complete.cases(modeldata), ];
names(modeldata) = c(names(dccdata)[i],names(dccdata)[j]);
name = paste0(c(names(modeldata))[1],"-",c(names(modeldata))[2]);
```

```
## The specification of ADCC - GJR - GARCH:
xspec = ugarchspec(mean.model = list(armaOrder = c(arma.order[i,1],
arma.order[i,2]), include.mean = TRUE), variance.model = list(garchOrder
= c(1,1), model = ’gjrGARCH’), distribution.model = ’norm’);
## 2 in the line below represents the number of variables uspec
= multispec(replicate(2, xspec));
## Asymmetric DCC(1,1) specification:
dccspec = dccspec(uspec = uspec, dccOrder = c(1, 1), model=’aDCC’, dis-
tribution = ’mvnorm’);
```

[15] There is no need to use the **ugarchfit** function in order to estimate GARCH coefficients in the table. Both GARCH and DCC processes can be estimated using **dccfit** function. For the user’s information, how to use the **ugarchfit** is shown here.

```
## Fit Garch
garch.fit = ugarchfit(xspec, data = modeldata, fit.control=list(scale=TRUE));
garch.fit.matrix[[name]] = garch.fit;
## Fit DCC
dcc.fit = dccfit(dccspec, data = modeldata, fit.control=list(scale=TRUE));
dcc.fit.matrix[[name]] = dcc.fit;
}}
```

```
## Print coefficint estimates from the model
dcc.fit.matrix[[names(dcc.fit.matrix)[2]]] ## DJI-GLD
dcc.fit.matrix[[names(dcc.fit.matrix)[3]]] ## DJI-CLF
dcc.fit.matrix[[names(dcc.fit.matrix)[4]]] ## DJI-BTCUSD
dcc.fit.matrix[[names(dcc.fit.matrix)[7]]] ## GLD-CLF
dcc.fit.matrix[[names(dcc.fit.matrix)[8]]] ## GLD-BTCUSD
dcc.fit.matrix[[names(dcc.fit.matrix)[12]]] ## CLF-BTCUSD
```

Those who are attentive will notice that the first command above indicates that the DCC estimation process is started with returns, not residuals. The reason is not that the returns are ARMA(0,0) processes. The set of commands given above feed the returns into the mean equation with the optimal lags in the ARMA(*p*,*q*). In this way, the returns series need not be white noise. The standardized residuals of the mean equation are first calculated and then used in the DCC process. The last six lines of the commands above print the coefficient estimates for ARMA(0,0)-GJR-GARCH(1,1)-ADCC(1,1) processes.

Figure 4 displays the time-varying volatility processes for each time series of this study. The figure confirms the interpretations of results about squared residuals in Panel E of Table 1 and Figure 3. Volatility process is clustered for the returns of all assets. Figures 5 and 6 illustrate the evolution of covariance and correlation processes over time, respectively. Despite the small changes in the covariance processes over time, the correlation processes undergo large shifts. Some asset pairs show correlation that flows from high to low, low to high, from positive to negative, or from negative to positive. It indicates that the diversification potential of these assets will differ in an optimized portfolio. Moreover, diversification potential changes as time passes.

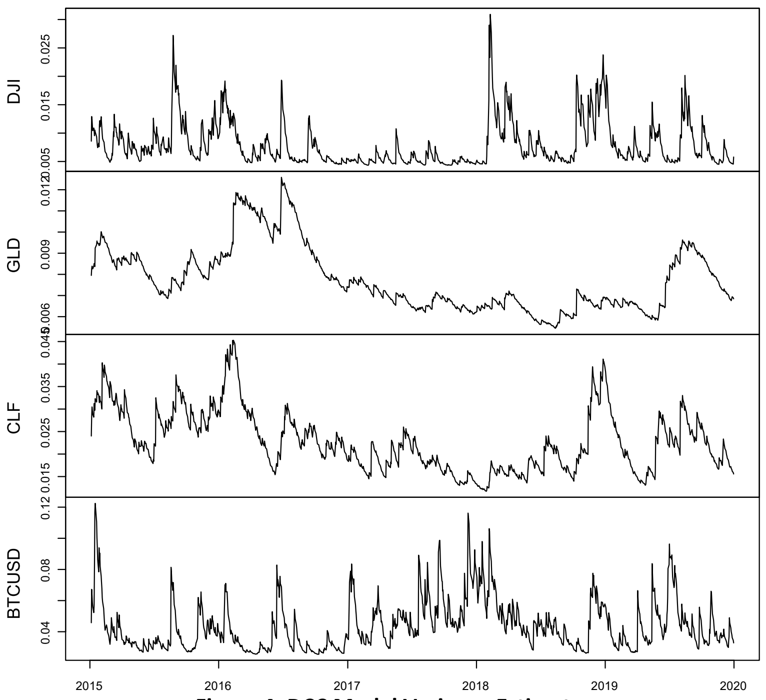

Figure 4: DCC Model Variance Estimates

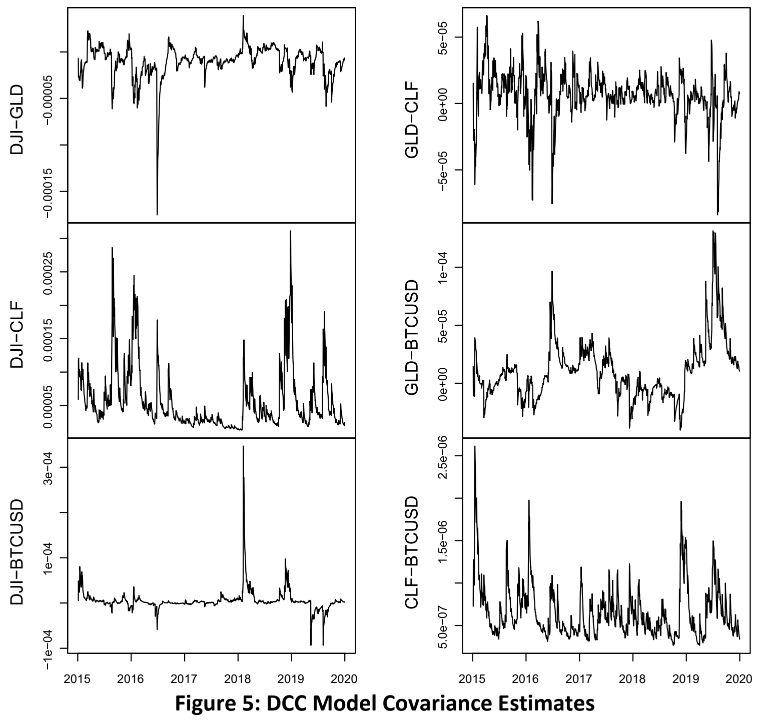

Figure 5: DCC Model Covariance Estimates

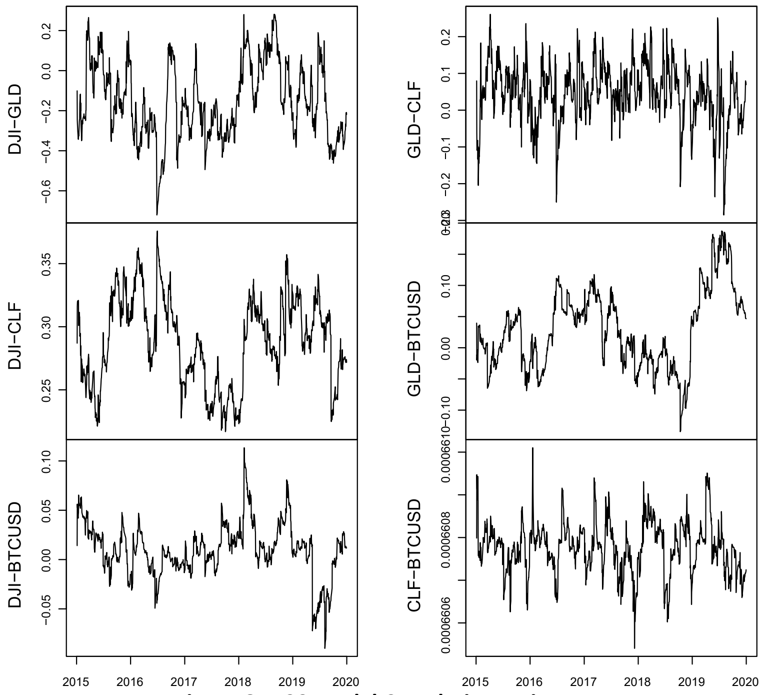

Figure 6: DCC Model Correlation Estimates

To produce Figures 4, 5, and 6, the following commands can be used. *i* is equal to 2 for DJI-GLD (index & gold) pair, 3 for DJI-CLF (index & crude oil) pair, 4 for DJI-BTDUSD (index & Bitcoin) pair, 7 for GLD-CLF (gold & crude oil) pair, 8 for GLD-BTCUSD (gold & Bitcoin) pair, and 12 for CLF-BTCUSD (crude oil & Bitcoin) pair.

```
i = 2;
## The variance processes:
plot(dcc.fit.matrix[[names(dcc.fit.matrix)[i]]],   which=2);
## The covariance processes:
plot(dcc.fit.matrix[[names(dcc.fit.matrix)[i]]],   which=3);
## The correlation processes:
plot(dcc.fit.matrix[[names(dcc.fit.matrix)[i]]],   which=4);
```

3.6.Portfolio optimization and rebalancing

For the portfolio optimization example in this study, the sample period is divided into two subperiods. The first two years of data are used to fit the model and the remaining data is used for portfolio optimization and rebalancing. The first time period spans January 5, 2015 to December 30, 2016. Using the data from this subperiod, all parameters necessary for forming the portfolio that optimizes the investor’s utility function are estimated. The first portfolio was constructed on January 4, 2017 using historical data up to that date. After 1 transaction day, portfolio weights are updated using data from January 5, 2015 to January 5, 2017. The updating process is repeated every 1 transaction days, using all the available data until the rebalancing day.[16]

[16] Explanations are provided regarding how to rebalance the portfolio at various time intervals as deemed necessary.

Using the DJI index, gold, crude oil, and Bitcoin, the study optimizes four portfolios with short sales. They are an equal weight portfolio, an efficient portfolio, a global minimum variance portfolio, and a tangency portfolio. The equal weight portfolio gives equal weights of 25 percent to each of the assets in the portfolio. The efficient portfolio simply tracks the returns on the DJI index. In the global minimum portfolio, the investor's wealth is allocated among the four financial assets so that the portfolio's risk is minimized. Finally, the tangency portfolio maximizes the portfolio's Sharpe ratio (Sharpe, 1966). Optimization is carried out using the variance-covariance matrices calculated from the DCC model in the previous section.[17]

There are several R packages available for portfolio optimization. In this article, Professor Eric Zivot's portfolio optimization code (*portfolio.R*) is used.[18] The code has all functions needed for the purpose of portfolio optimization in this study. The usage of the code is different from the usual procedure of installing and loading a package to R. In order to be able to use the functions, one must first save the code as an R file in the current working directory of R. Working directory simply refers to the location of any file you read into, or save from, R. To see your current working directory in R, run **getwd()** in your R console.[19] After the R file *portfolio.R* is saved in your working directory, use the **source** function to load the code to your R library:

```
source('/Volumes/(username)/Desktop/dccoptimization/portfolio.R')
```

The next step is to define some parameters of portfolio optimization and create some blank *xts* and *zoo* objects to be used later.[20] [21]

[17] See a review of modern portfolio theory in Elton & Gruber (1997).

[18] The code is available to public at https://faculty.washington.edu/ezivot/econ424/portfolio.r.

[19] Users are advised to set a new working directory for a new project. To set a new working directory in R, use **setwd()** function. For example, to set a file named "dccoptimization" on your computer's desktop as your current working directory, run **setwd("/Users/(username)/Desktop/dccoptimization**") for Windows users and **setwd("/Volumes/(username)/Desktop/dccoptimization")** for MacBook users.

[20] If the user does not wish to permit short-selling, change this line of command to shortselling = FALSE

[21] "rebalance = 1" is where the frequency of rebalancing the portfolio is determined. The user can change this line to rebalance = 5 for weekly updating, rebalance = 21 for monthly updating, and so on.

```
shortselling = TRUE;
rebalance = 1;
eret = returns*0;
eRF = returns[,1]*0;
covmat = matrix(NA, ncol=NCOL(eret),nrow=NCOL(eret));
ew = rep(1,NCOL(returns))/NCOL(returns);
equalWeight_portfolio_ER = returns[,1]*0;
equalWeight_portfolio_SD = returns[,1]*0;
equalWeight_portfolio_W = returns*0;
efficient_portfolio_ER = returns[,1]*0;
efficient_portfolio_SD = returns[,1]*0;
efficient_portfolio_W = returns*0;
globalMin_portfolio_ER = returns[,1]*0;
globalMin_portfolio_SD = returns[,1]*0;
globalMin_portfolio_W = returns*0;
tangency_portfolio_ER = returns[,1]*0;
tangency_portfolio_SD = returns[,1]*0;
tangency_portfolio_W = returns*0;
```

The next step is to calculate the expected returns of each asset class up to the portfolio creation and rebalancing date.

```
for (t in seq(from=501, to=NROW(returns), by=rebalance)){;
for (i in 1:NCOL(returns)){;
eret[t,i] = mean(returns[1:t,i],na.rm=TRUE) };
eRF[t,1] = mean(RF[1:t,1],na.rm=TRUE) };
```

This is another loop command. Using the data from the 1^{s}t observation (January 5, 2015) to the 501st observations (January 4, 2017), it will calculate the expected return for that day. In the next loop, all the data between the 1st and 502nd observations will be used to compute the expected returns, and so on. At this point, the following commands can be used to optimize portfolios with different purposes:

```
for (t in seq(from=501, to=NROW(returns), by=rebalance)){;
covmat[1,1]  = var.est.z[t,"DJI"];
covmat[2,2]  = var.est.z[t,"GLD"];
covmat[3,3]  = var.est.z[t,"CLF"];
covmat[4,4]  = var.est.z[t,"BTCUSD"];
covmat[2,1]  = covmat[1,2]  = cov.est.z[t,"DJI-GLD"];
covmat[3,1]  = covmat[1,3]  = cov.est.z[t,"DJI-CLF"];
covmat[4,1]  = covmat[1,4]  = cov.est.z[t,"DJI-BTCUSD"];
covmat[3,2]  = covmat[2,3]  = cov.est.z[t,"GLD-CLF"];
covmat[4,2]  = covmat[2,4]  = cov.est.z[t,"GLD-BTCUSD"];
covmat[4,3]  = covmat[3,4]  = cov.est.z[t,"CLF-BTCUSD"];
```

```
## The equal weight portfolio expected returns and std.dev.:
equalWeight_portfolio = getPortfolio(er=eret[t,],cov.mat = covmat,
weights = ew);
equalWeight_portfolio_ER[t,] = equalWeight_portfolio$er;
equalWeight_portfolio_SD[t,] = equalWeight_portfolio$sd;
equalWeight_portfolio_W[t,] = equalWeight_portfolio$weights;
```

```
## The target portfolio expected returns and std.dev.:
target.return = eret[t,"DJI"];
efficient_portfolio = efficient.portfolio(er=eret[t,],cov.mat = covmat,
target.return, shorts=paste0(shortselling)); efficient_portfolio_ER[t,] =
efficient_portfolio$er;
efficient_portfolio_SD[t,] = efficient_portfolio$sd;
efficient_portfolio_W[t,] = efficient_portfolio$weights;
```

```
## The global minimum variance portfolio expected returns and std.dev.:
globalMin_portfolio = globalMin.portfolio(er=eret[t,],cov.mat = covmat,
shorts=paste0(shortselling));
globalMin_portfolio_ER[t,] = globalMin_portfolio$er;
globalMin_portfolio_SD[t,] = globalMin_portfolio$sd;
globalMin_portfolio_W[t,] = globalMin_portfolio$weights;
```

```
## The tangency portfolio expected returns and std.dev.:
r.free = as.numeric(eRF[t,]);
tangency_portfolio = tangency.portfolio(er=eret[t,],cov.mat = covmat,
r.free, shorts=paste0(shortselling));
tangency_portfolio_ER[t,] = tangency_portfolio$er;
tangency_portfolio_SD[t,] = tangency_portfolio$sd;
tangency_portfolio_W[t,] = tangency_portfolio$weights; }
```

The commands above, first create a 4 by 4 variance covariance matrix using time-varying variance and covariance processes from the ADCC model estimates, and then use them in portfolio optimization along with the expected returns. The study employs the **getPortfolio** function for the equal weight portfolio, the **efficient.portfolio** function for the target return portfolio, the **globalMin.portfolio** function for the global minimum variance portfolio, and finally the **tangency.portfolio** function for the tangency portfolio. Once the portfolios are estimated, the next three lines of command after every function pulls the expected returns, standard deviations, and the weights allocated to the assets.

In Figure 7 are plotted the fluctuating returns of the portfolios and their standard deviations over time. It is not surprising that the expected returns of the tangency portfolio are the largest as well as the standard deviations. They fluctuate widely across all portfolios.

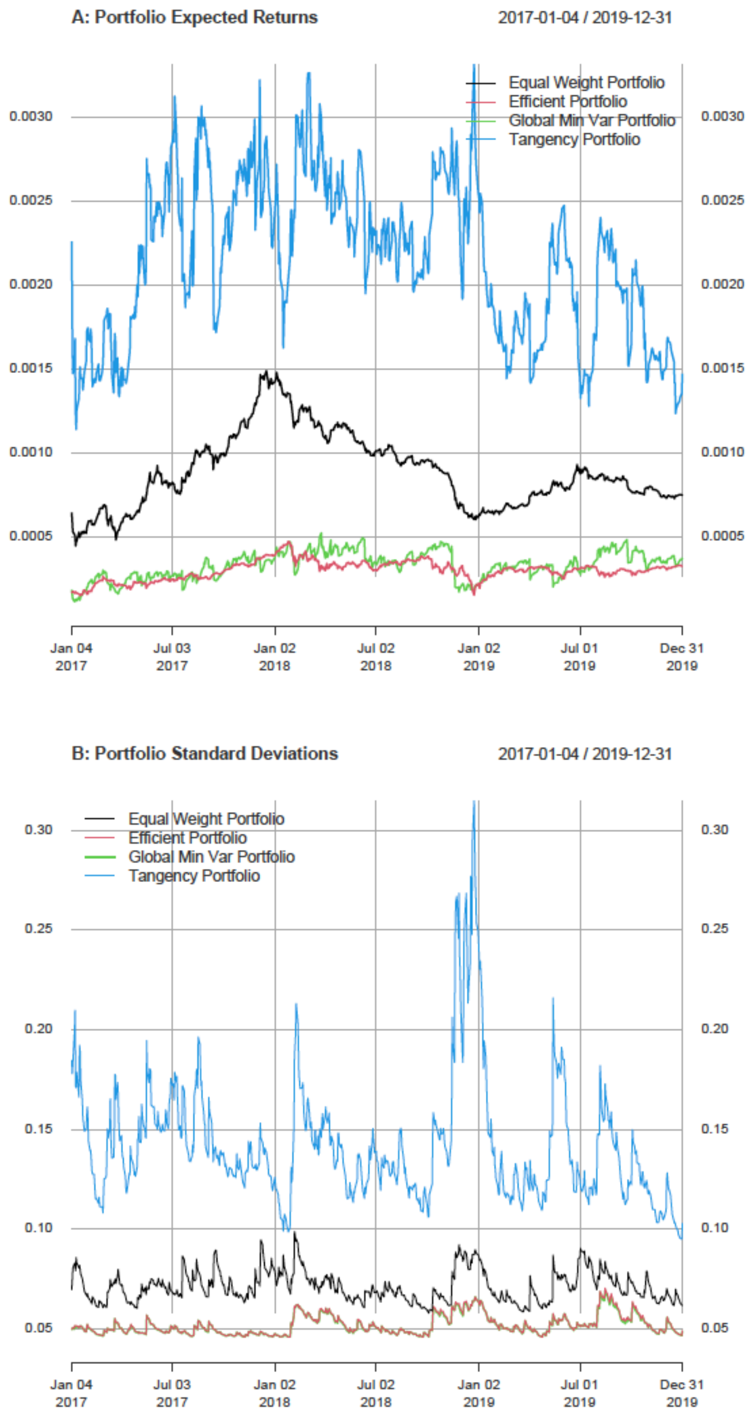

Figure 7: Expected Returns and Standard Deviations of Portfolios

Showing the changes in the allocations of weights to the four asset classes in the portfolio, Figure 8 provides a closer look at the tangency portfolio. Even though the weights of all asset classes fluctuate significantly, Bitcoin almost always holds the largest share. Since short-selling is permitted in the portfolio optimization, the investor takes a short position on gold and crude oil at the end of 2018 and the beginning of 2019 and allocates the amount to Bitcoin.

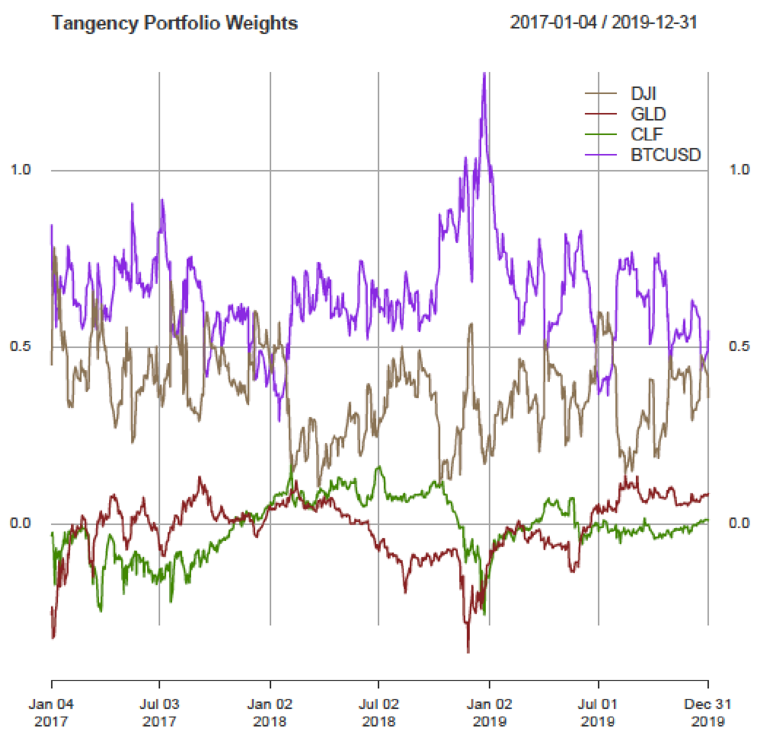

Figure 8: Tangency Portfolio Weights Over-time

Figures 7 and 8 can be created by using the following commands:

```
portfolio_ER = cbind(equalWeight_portfolio_ER, efficient_portfolio_ER,
globalMin_portfolio_ER, tangency_portfolio_ER);
colnames(portfolio_ER) = c("Equal Weight Portfolio","Efficient Portfo-
lio","Global Min Var Portfolio","Tangency Portfolio");
portfolio_SD = cbind(equalWeight_portfolio_SD, efficient_portfolio_SD,
globalMin_portfolio_SD, tangency_portfolio_SD);
colnames(portfolio_SD) = c("Equal Weight Portfolio","Efficient Portfo-
lio","Global Min Var Portfolio","Tangency Portfolio");
```

```
## Replace 0 with NA
portfolio_ER[portfolio_ER == 0] <- NA;
portfolio_SD[portfolio_SD == 0] <- NA
## remove the row if any column contains NA:
portfolio_ER <- portfolio_ER[complete.cases(portfolio_ER), ];
portfolio_SD <- portfolio_SD[complete.cases(portfolio_SD), ];
```

```
plot(portfolio_ER, main = "Portfolio Expected Returns");
addLegend("topright",legend.names = c(colnames(portfolio_ER)),lty =  c(1,
1,1,1))
plot(portfolio_SD, main = "Portfolio Standard Deviations");
addLegend("topright",legend.names = c(colnames(portfolio_SD)),lty =  c(1,
1,2,1));
plot(tangency_portfolio_W, main = "Tangency Portfolio Weights", col  =
c("burlywood4", "brown4","chartreuse4","blueviolet"));
addLegend("topright", legend.names = c(colnames(tangency_portfolio_W)),
col = c("burlywood4", "brown4","chartreuse4","blueviolet"), lty = c(1,
1,1,1));
```

The above commands combine the expected returns and standard deviations from each portfolio optimization strategy, set column names for the newly combined data, and remove the days on which there is no portfolio rebalancing. Finally, the plots illustrating the evolution of expected returns and standard deviations are constructed.

The study also displays all possible risk and return combinations for the DJI index, gold, crude oil, and Bitcoin. Figure 9 displays the efficient frontier and the capital market line using the four asset classes and a risk-free rate. Bitcoin offers the greatest risk and reward when compared to the DJI, gold, and crude oil. Utilizing Markowitz's mean-variance model, portfolios with improved risk-return characteristics can be constructed along the capital market line shown in green in the figure.

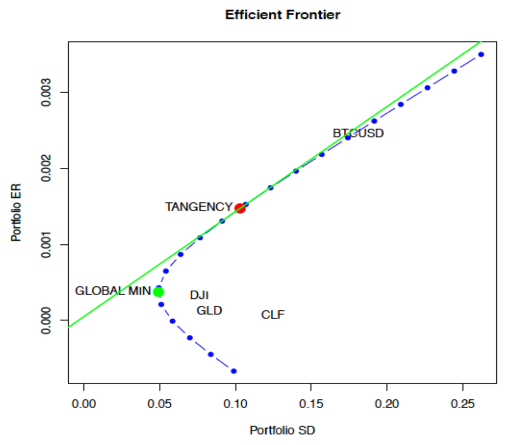

Figure 9: Efficient Frontier and Capital Market Line

Figure 9 can be created by using the following commands:

```
ef = efficient.frontier(eret[NROW(eret),], covmat, alpha.min = -0.5, al-
pha.max = 1.5, nport = 20);
plot(ef, plot.assets=T, col="blue", pch=16);
points(gmin.port$sd, gmin.port$er,col="green", pch=16, cex=2);
points(tan.port$sd, tan.port$er, col= "red", pch=16, cex=2);
text(gmin.port$sd, gmin.port$er, labels= "GLOBAL MIN", pos= 2);
text(tan.port$sd, tan.port$er, labels="TANGENCY", pos=2);
sr.tan = (tan.port$er - r.free)/tan.port$sd;
abline(a=r.free, b=sr.tan, col="green",lwd=2);
```

4.Concluding remarks

The present study demonstrates how to use a GJR(1)-GARCH(1,1)-Asymmetric DCC(1,1) model to estimate the dynamic variance-covariance matrix to optimize the portfolio according to Markowitz's mean-variance approach. In the portfolio optimization framework, each optimization step is explained by an equation, and the implementation of the whole process in the R programming language is demonstrated. This study shows how the R programming language can be used to download the financial data of various asset classes to which an investor will allocate wealth and what functions can be used to determine the optimal weight of each asset in portfolios to meet various objectives.

Based on empirical results, we can conclude that the variance-covariance processes of the assets being considered are time-varying, and that a dynamic portfolio optimization by rebalancing the portfolio at a regular interval will capture the changes in the riskiness of these assets. This study presents the necessary R com- mands to calculate the updated weights of assets to reflect any changes in their risk levels.

References

Ardia, D., Bolliger, G., Boudt, K., & Gagnon-Fleury, J.-P. (2017). The impact of covariance misspecification in risk-based portfolios. *Annals of Operations Research*, 254, 1–16.

Billio, M., Caporin, M., & Gobbo, M. (2006). Flexible dynamic conditional correlation multivariate garch models for asset allocation. *Applied Financial Economics Letters*, 2, 123–130.

Bouri, E., Molnár, P., Azzi, G., Roubaud, D., & Hagfors, L. I. (2017). On the hedge and safe haven properties of bitcoin: Is it really more than a diversifier? *Finance Research Letters*, 20, 192–198.

Briere, M., Oosterlinck, K., & Szafarz, A. (2015). Virtual currency, tangible return: Portfolio diversification with bitcoin. *Journal of Asset Management*, 16, 365–373.

Cappiello, L., Engle, R. F., & Sheppard, K. (2006). Asymmetric dynamics in the correlations of global equity and bond returns. *Journal of Financial econometrics*, 4, 537–572.

Chen, Y., & Nie, Y. (2018). *Value at risk when covariance is misspecified*. Proceedings of the International Conference on Industrial Engineering and Operations Management, Washington DC, USA.

Conover, C. M., Jensen, G. R., Johnson, R. R., & Mercer, J. M. (2010). Is now the time to add commodities to your portfolio? *The Journal of Investing*, 19, 10–19.

Eddelbuettel, D. (2022). *Cran task view: Empirical finance*.

Elton, E. J., & Gruber, M. J. (1997). Modern portfolio theory, 1950 to date. *Journal of banking & finance*, 21,1743–1759.

Engle, R. (2002). Dynamic conditional correlation: A simple class of multivariate generalized autoregressive conditional heteroskedasticity models. *Journal of Business & Economic Statistics*, 20, 339–350.

Ensor, K. B., & Koev, G. M. (2014). *Computational finance: correlation, volatility, and markets*. Wiley Interdisciplinary Reviews: Computational Statistics, 6, 326–340.

Galanos, A. (2022). *rmgarch: Multivariate GARCH models*. R package version 1.3-9.

Gao, X., & Nardari, F. (2018). Do commodities add economic value in asset allocation? new evidence from time-varying moments. *Journal of Financial and Quantitative Analysis*, 53, 365–393.

Ghalanos, A. (2022). *rugarch: Univariate GARCH models*. R package version 1.4-7.

Glosten, L. R., Jagannathan, R., & Runkle, D. E. (1993). On the relation between the expected value and the volatility of the nominal excess return on stocks. *The journal of finance*, 48, 1779–1801.

Gorton, G., & Rouwenhorst, K. G. (2006). Facts and fantasies about commodity futures. *Financial Analysts Journal*,62, 47–68.

Hafner, C., & Franses, P. H. (2003). *A generalized dynamic conditional correlation model for many asset returns*. Technical Report.

Hillier, D., Draper, P., & Faff, R. (2006). Do precious metals shine? an investment perspective. *Financial Analysts Journal*, 62, 98–106.

Rob J Hyndman, Rebecca Killick (2022). CRAN Task View: Time Series Analysis. Version 2022-07-19. URL https://CRAN.R-project.org/view=TimeSeries.

Hyndman, R. J., & Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for R. Journal of Statistical Software, 26, 1–22. doi:10.18637/jss.v027.i03.

Klein, T., Thu, H. P., & Walther, T. (2018). Bitcoin is not the new gold–a comparison of volatility, correlation, and portfolio performance. *International Review of Financial Analysis*, 59, 105–116.

Komsta, L., & Novomestky, F. (2015). moments: Moments, cumulants, skewness, kurtosis and related tests. URL: https://CRAN.R-project.org/package=moments r package version 0.14.

Liu, Q., Tse, Y., & Zhang, L. (2018). Including commodity futures in asset allocation in china. *Quantitative Finance*, 18, 1487–1499.

Markowitz, H. (1952). The utility of wealth. *Journal of political Economy*, 60, 151–158.

McAleer, M., Chan, F., Hoti, S., & Lieberman, O. (2008). Generalized autoregressive conditional correlation. *Econometric Theory*, 24, 1554–1583.

Mullen, K. M. (2014). Continuous global optimization in R. *Journal of Statistical Software*, *60*, 1-45.

NYU-Libraries, N. Y. U. N. (2022). Quantitative analysis guide: Which statistical software to use? (2022) (Accessed: 13 March 2022). URL: https://guides.nyu.edu/quant/statsoft.

Peterson, B. G., & Carl, P. (2020). PerformanceAnalytics: Econometric Tools for Performance and Risk Analysis. URL: https://CRAN.R-project.org/package=PerformanceAnalytics r package version 2.0.4.

Platanakis, E., & Urquhart, A. (2020). Should investors include bitcoin in their portfolios? a portfolio theory approach. *The British accounting review*, 52, 100837.

R Core Team (2021). R: A Language and Environment for Statistical Computing.R Foundation for Statistical Computing Vienna, Austria. URL: https://www.R-project.org/.

RStudio Team (2016). RStudio: Integrated Development for R. R Foundation for Statistical Computing Boston, MA. URL: https://www.rstudio.com/.

Ryan, J. A., & Ulrich, J. M. (2020). quantmod: Quantitative Financial Modelling Framework. URL: https://CRAN.R-project.org/package=quantmod r package version 0.4.18. Sharpe, W. F. (1966). Mutual fund performance. *The Journal of business*, 39, 119–138.

Trapletti, A., & Hornik, K. (2021). tseries: Time Series Analysis and Computational Finance. URL: https://CRAN.R-project.org/package=tseries r package version 0.10-49.

Zeileis, A. (2005). Cran task views. R News, 5, 39–40.

Zivot, E. (2008). Computing efficient portfolios in R. University of Washington - Technical Report.

Beyan ve Açıklamalar (Disclosure Statements)

1. Bu çalışmanın yazarları, araştırma ve yayın etiği ilkelerine uyduklarını kabul etmektedirler (The authors of this article confirm that their work complies with the principles of research and publication ethics).

2. Yazarlar tarafından herhangi bir çıkar çatışması beyan edilmemiştir (No potential conflict of interest was reported by the authors).

3. Bu çalışma, intihal tarama programı kullanılarak intihal taramasından geçirilmiştir (This article was screened for potential plagiarism using a plagiarism screening program).