Homework 4

Vladislav Zakatov

15 December 2015

This document has been created as part of the fourth homework assignment at CMF.

Initially, the working directory, system locale are set and the required packages are loaded.

```
##### Initialisation #####
setwd("~/")
library(quantmod)
library(ggplot2)
library(ghyp)
library(FinTS)
library(copula)
library(fGarch)
library(tseries)
Sys.setlocale("LC_ALL","English")
```

The historical prices of S&P 500 and Dow Jones ETFs from 2014 up to present are loaded from Google Finance using *getSymbols* function from *quantmod* package.

```
# specify the tickers to load
tickers = c("SPY","DIA")

# environment where data will be stored
e <- new.env()

# suppress warning about future change in the function behaviour
options("getSymbols.warning4.0"=FALSE)

# load data from Yahoo Finance
getSymbols(tickers, src = "google", from="2014-01-01", env=e)

# merge data into a single xts taking only adjusted close prices
data = do.call(merge, eapply(e, Cl)[tickers])
names(data) = tickers
```

Daily returns are calculated and saved as *xts*.

```
##### Calculate returns #####
rets = apply(data, 2, function(x) diff(x)/x[-length(x)])
rets = xts(rets, order.by = index(data)[-1])
```

We now apply an LM-test with 12 lags for ARCH effects to both series.

```
##### LM-test #####
ArchTest(rets$SPY,lags=12)
```

```
##
```

```
##  ARCH LM-test; Null hypothesis: no ARCH effects
##
## data:  rets$SPY
## Chi-squared = 103.26, df = 12, p-value < 2.2e-16
```

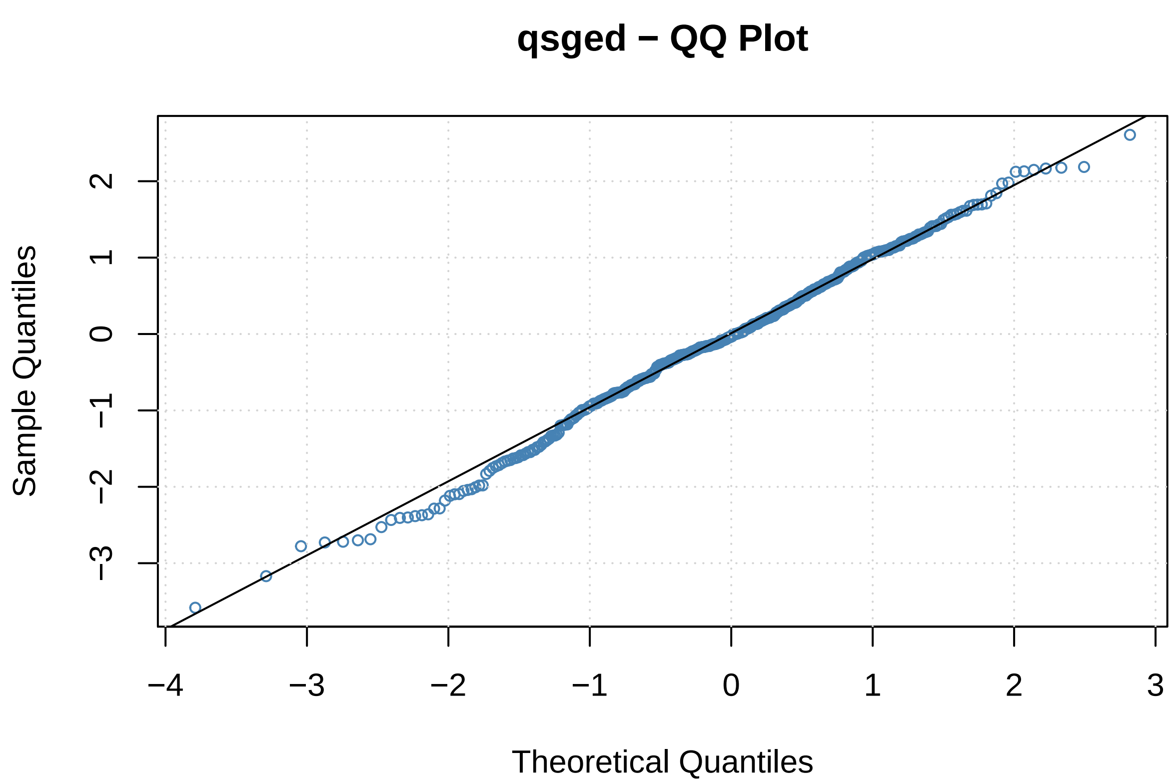
```r
ArchTest(rets$DIA,lags=12)
```

```
##
##  ARCH LM-test; Null hypothesis: no ARCH effects
##
## data:  rets$DIA
## Chi-squared = 114.07, df = 12, p-value < 2.2e-16
```

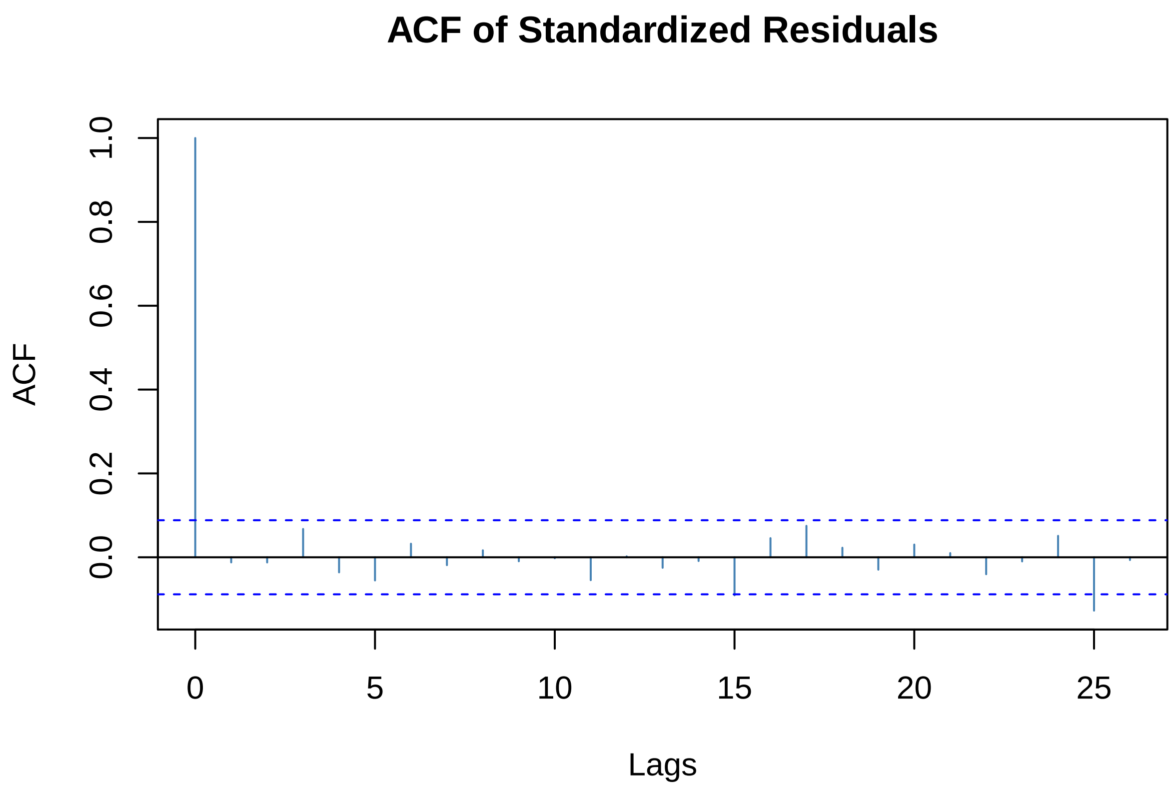
We reject the null hypothesis with a very low p-value for both series and thus conclude that there are ARCH effects in the returns series.

We now fit a GARCH model including leverage and shape (set to 1.2) parameters to both series. We use the Skew Generalized Error Distribution as a conditional distribution.

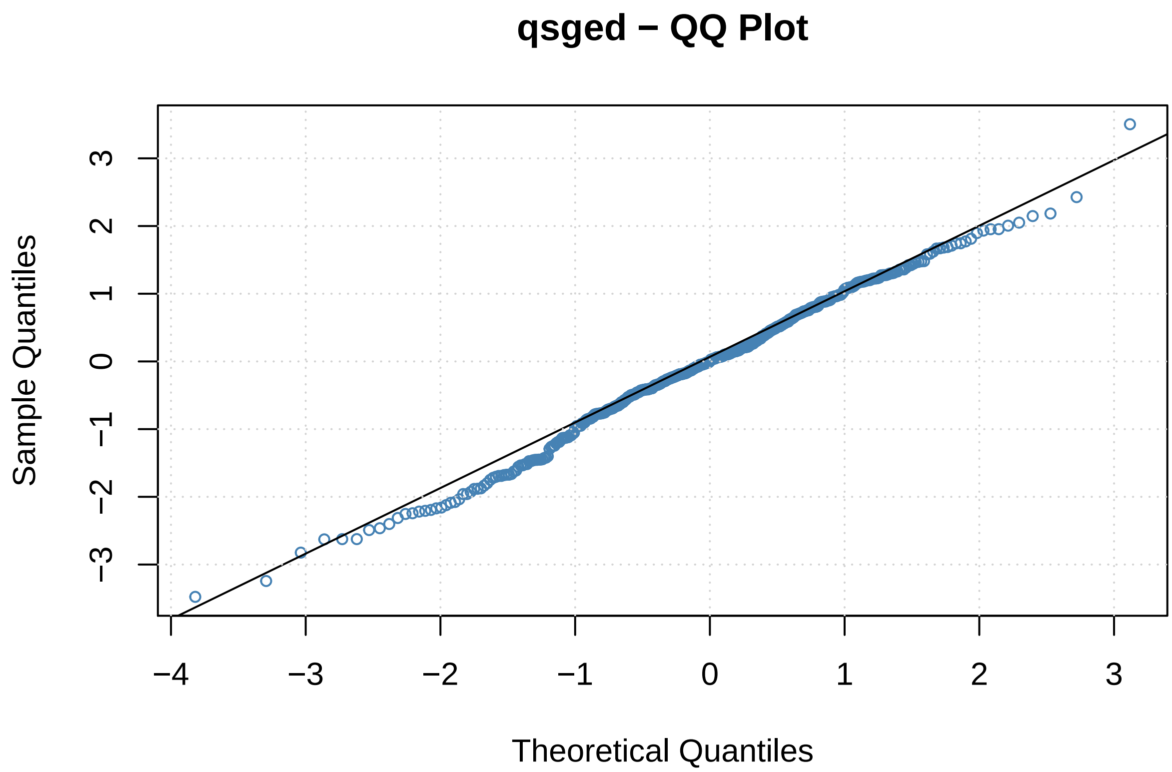
```r
##### GARCH modelling #####
#SPY
SPY.gfit <- garchFit(formula=~aparch(1,1),data=rets$SPY,delta=2,
                     include.delta=FALSE,leverage=TRUE,cond.dist="sged",
                     shape=1.2,include.shape=TRUE,trace=FALSE)

plot(SPY.gfit,which=13)
```



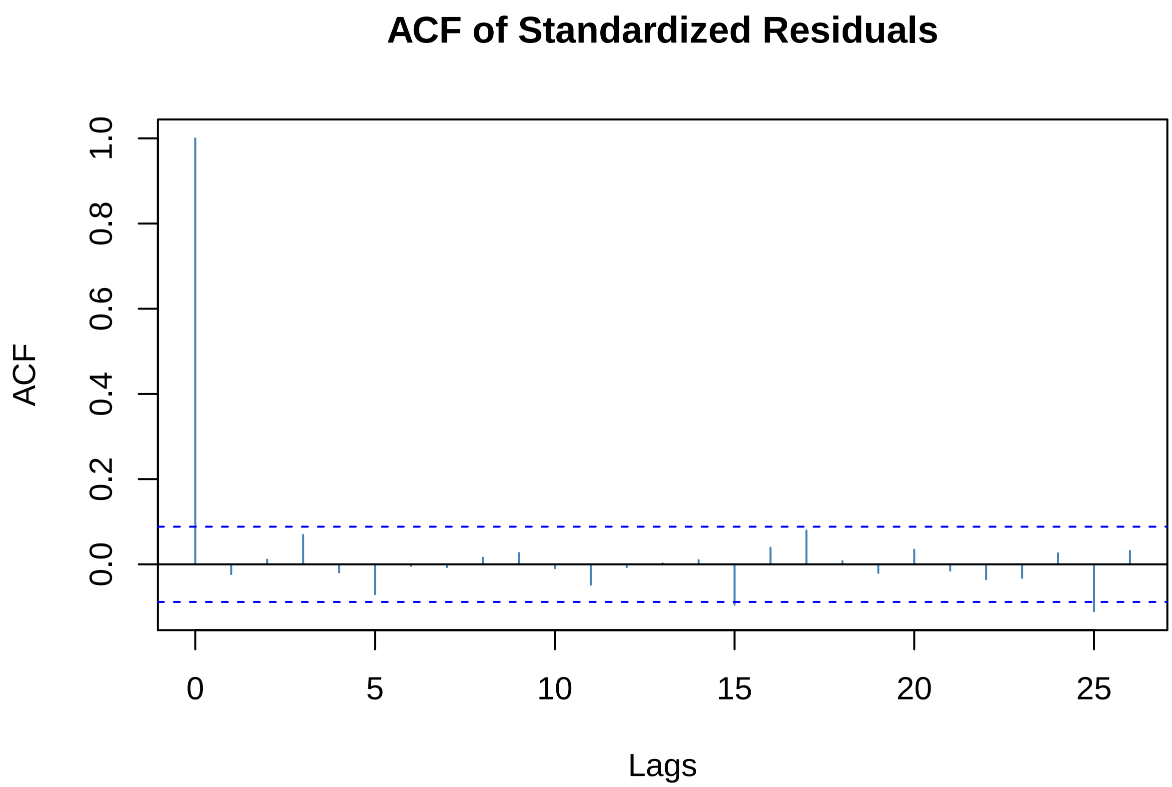
```r
plot(SPY.gfit,which=10)
```




```r
#DIA
DIA.gfit <- garchFit(formula=~aparch(1,1),data=rets$DIA,delta=2,
                     include.delta=FALSE,leverage=TRUE,cond.dist="sged",
                     shape=1.2,include.shape=TRUE,trace=FALSE)

plot(DIA.gfit,which=13)
```




```
plot(DIA.gfit,which=10)
```



We now conduct ADF-, PP- and KPSS-tests for stationarity.

```
##### Stationarity tests #####
adf.test(rets$SPY)
adf.test(rets$DIA)

pp.test(rets$SPY)
pp.test(rets$DIA)

kpss.test(rets$SPY, null="Level")
kpss.test(rets$DIA, null="Level")
```

We can conclude about the stationarity of the series.

We now standardise the residuals and compute their partial distributions using the CDF of the Skew Generalized Error Distribution.

```
##### Standardized residuals #####
z <- matrix(nrow=length(rets$SPY),ncol=2)
z[,1] <- SPY.gfit@residuals / SPY.gfit@sigma.t
z[,2] <- DIA.gfit@residuals / DIA.gfit@sigma.t

##### Residuals partial distributions #####
mean <- c(0,0); sd <- c(1,1); nu <- c(1.2,1.2)

xi <- c(SPY.gfit@fit$par["skew"], DIA.gfit@fit$par["skew"])
```

```
cdf <- matrix(nrow=length(rets$SPY),ncol=2)
for (i in 1:2) cdf[,i] = psged(z[,i],mean=mean[i],sd=sd[i],nu=nu[i],xi=xi[i])
```

We initialise four different copulas and fit them to our data.

```
##### Initialise copulas #####
norm.cop <- normalCopula(dim=2,param=0.5,dispstr="un")
stud.cop <- tCopula(dim=2,param=0.5,df=5,df.fixed=TRUE,dispstr="un")
gumb.cop <- gumbelCopula(dim=2,param=2)
clay.cop <- claytonCopula(dim=2,param=2)

##### Fit copulas #####
norm.fit <- fitCopula(cdf,copula=norm.cop)
stud.fit <- fitCopula(cdf,copula=stud.cop)
gumb.fit <- fitCopula(cdf,copula=gumb.cop)
clay.fit <- fitCopula(cdf,copula=clay.cop)
```

We now conduct a Monte-Carlo simulation of the returns based on the best-fit Student copula and calculate the portfolio returns assuming equal weights of the assets in the portfolio.

```
##### Monte-Carlo #####
N = 100000
cdf.sim <- rcopula(n=N,copula=stud.fit@copula)
z.sim <- matrix(nrow=N,ncol=2)
for (i in 1:2) z.sim[,i] <- qsged(cdf.sim[,i],
                                  mean=mean[i],sd=sd[i],nu=nu[i],xi=xi[i])
frc1 <- predict(SPY.gfit,n.ahead=1)
frc2 <- predict(DIA.gfit,n.ahead=1)
mu <- c(frc1[,1],frc2[,1])
sigma <- c(frc1[,3],frc2[,3])

##### Modelled portfolio returns #####
w = c(0.5, 0,5)
prt.sim <- w[1]*(mu[1]+sigma[1]*z.sim[,1]) +
  w[2]*(mu[2]+sigma[2]*z.sim[,2])
```

We now calculate Value-at-Risk and Expected Shortfall for the portfolio..

```
alpha <- 0.10
prt.sim <- sort(prt.sim)
VaR <- prt.sim[alpha*N]
ES <- mean(prt.sim[1:(alpha*N-1)])

print(VaR)
```

```
## [1] -0.007524986
```

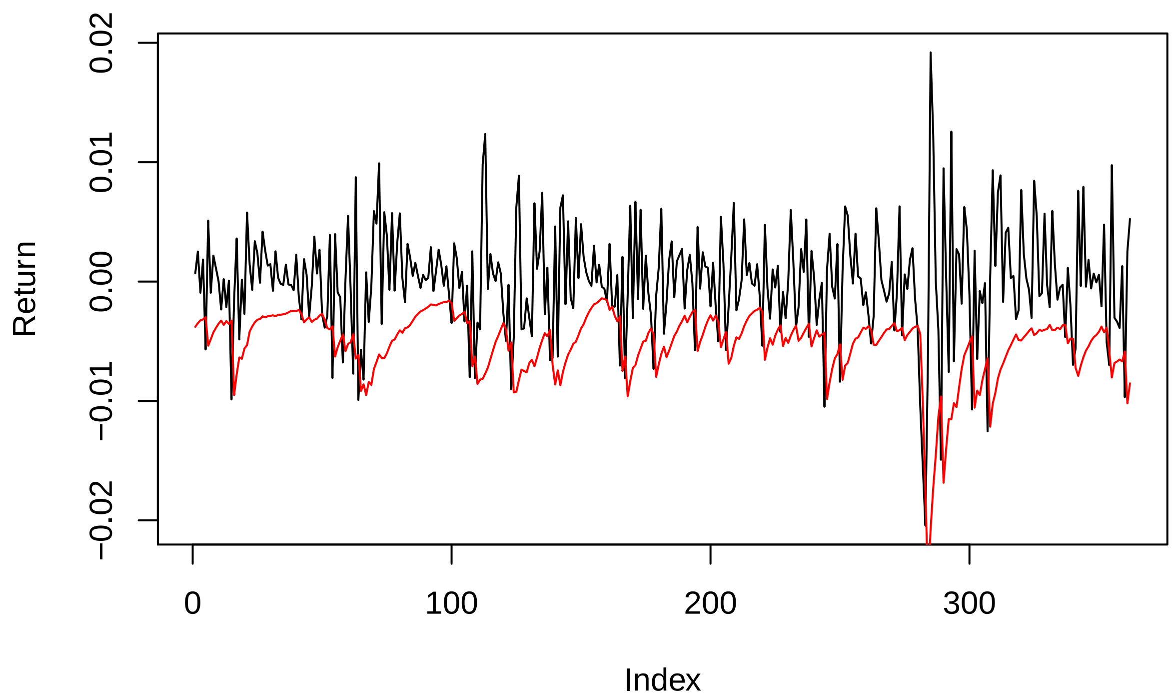
```
print(ES)
```

```
## [1] -0.0120965
```

Finally, we now want to calculate the VaR curve through a 130 day window. We fit GARCH and copula models every iteration.

```
##### Calculate VaR curve #####
VaR_curve <- numeric()
h <- 0.5 * 260 # window length

for (i in (h+1):length(rets[,1]))
{
  h.rets <- rets[(i-h):(i-1),]

  SPY.gfit <- garchFit(formula=~aparch(1,1),data=h.rets$SPY,delta=2,
                       include.delta=FALSE,leverage=TRUE,cond.dist="sged",
                       shape=1.2,include.shape=TRUE,trace=FALSE)

  DIA.gfit <- garchFit(formula=~aparch(1,1),data=h.rets$DIA,delta=2,
                       include.delta=FALSE,leverage=TRUE,cond.dist="sged",
                       shape=1.2,include.shape=TRUE,trace=FALSE)

  z <- matrix(nrow=length(h.rets$SPY),ncol=2)
  z[,1] <- SPY.gfit@residuals / SPY.gfit@sigma.t
  z[,2] <- DIA.gfit@residuals / DIA.gfit@sigma.t

  mean <- c(0,0); sd <- c(1,1); nu <- c(1.2,1.2)
  xi <- c(SPY.gfit@fit$par["skew"], DIA.gfit@fit$par["skew"])
  cdf <- matrix(nrow=length(h.rets$SPY),ncol=2)
  for (j in 1:2) cdf[,j] = psged(z[,j],mean=mean[j],sd=sd[j],nu=nu[j],xi=xi[j])

  stud.fit <- fitCopula(cdf,copula=stud.cop)

  N = 100000
  cdf.sim <- rcopula(n=N,copula=stud.fit@copula)
  z.sim <- matrix(nrow=N,ncol=2)
  for (j in 1:2) z.sim[,j] <- qsged(cdf.sim[,j],
                                    mean=mean[j],sd=sd[j],nu=nu[j],xi=xi[j])
  frc1 <- predict(SPY.gfit,n.ahead=1)
  frc2 <- predict(DIA.gfit,n.ahead=1)
  h.mu <- c(frc1[,1],frc2[,1])
  h.sigma <- c(frc1[,3],frc2[,3])

  h.prt.sim <- w[1]*(h.mu[1]+h.sigma[1]*z.sim[,1]) +
    w[2]*(h.mu[2]+h.sigma[2]*z.sim[,2])

  h.prt.sim <- sort(h.prt.sim)

  VaR_curve[i-h] <- h.prt.sim[alpha*N]
}
```

```
fact <- rets[(h+1):length(rets[,1]),1]*w[1] + rets[(h+1):length(rets[,2]),2]*w[2]
plot(as.numeric(fact),type="l", ylab = "Return")
lines(VaR_curve,col="red")
```



In order to check the adequacy of the VaR curve we conduct the Kupiec test.

```
##### Kupiec test #####
K <- sum(fact<VaR_curve); alpha0 <- K/length(rets[,1])
S <- -2*log((1-alpha)^(length(rets[,1])-K)*alpha^K)+
  2*log((1-alpha0)^(length(rets[,1])-K)*alpha0^K)
p.value <- 1-pchisq(S,df=1)
```

Kupiec test shows that the actual number of drawdowns (alpha0 = 0.089) is not statistically different from the modelled one (alpha = 0.1) with a high p-value (0.43).