

Multivariate Models

The goal of this project is to study a concrete example of multivariate stock returns using R packages and functions.

- Perform graphical and formal tests of normality
- Fit multivariate models to the data
- test the goodness-of-fit of the model using the log-likelihood ratio test.

0) Setup

The first step is to load the necessary libraries.

```
#install.packages(c('xts', 'qrmttools', 'qrmdata', 'ghyp'))  
  
library(xts)  
library(qrmttools)  
library(qrmdata)  
library(ghyp)
```

1) Data preparation

The data consists of an `xts` object containing adjusted close prices of the constituents of the Dow Jones index. The data has been obtained from Yahoo Finance as of 2016-01-03, via the function `get_data()` from the package `qrmttools`. The constituents data ranges from the first date at least one of the constituents is available (with missing data if not available) to 2015-12-31.

```
## Dow Jones constituents  
data("DJ_const")  
str(DJ_const)  
  
## An 'xts' object on 1962-01-02/2015-12-31 containing:  
##   Data: num [1:13595, 1:30] NA ...  
##   - attr(*, "dimnames")=List of 2  
##     ..$ : NULL  
##     ..$ : chr [1:30] "AAPL" "AXP" "BA" "CAT" ...  
##   Indexed by objects of class: [Date] TZ: UTC  
##   xts Attributes:  
##   List of 2  
##     $ src    : chr "yahoo"  
##     $ updated: POSIXct[1:1], format: "2016-01-03 03:51:40"  
head(DJ_const[,1:10],5)
```

```
##          AAPL AXP      BA      CAT CSCCO CVX       DD       DIS       GE GS  
## 1962-01-02  NA  NA 0.212905 0.593184  NA  NA 1.227958 0.061014 0.145967 NA  
## 1962-01-03  NA  NA 0.217163 0.598964  NA  NA 1.229230 0.061832 0.144501 NA  
## 1962-01-04  NA  NA 0.215034 0.614372  NA  NA 1.220331 0.061832 0.142794 NA  
## 1962-01-05  NA  NA 0.210773 0.620152  NA  NA 1.187280 0.062039 0.139132 NA  
## 1962-01-08  NA  NA 0.211306 0.624001  NA  NA 1.169484 0.061832 0.138889 NA
```

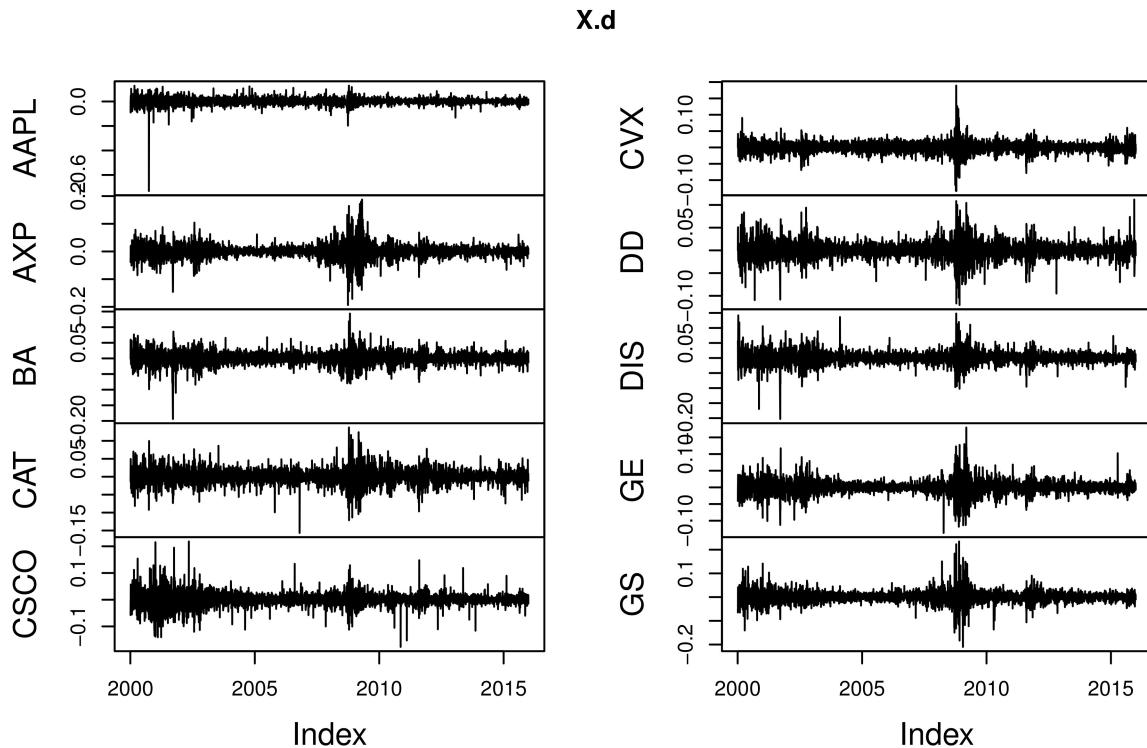
We will only focus on the 10 first constituents, starting from January 1st, 2000.

In the code below, we:

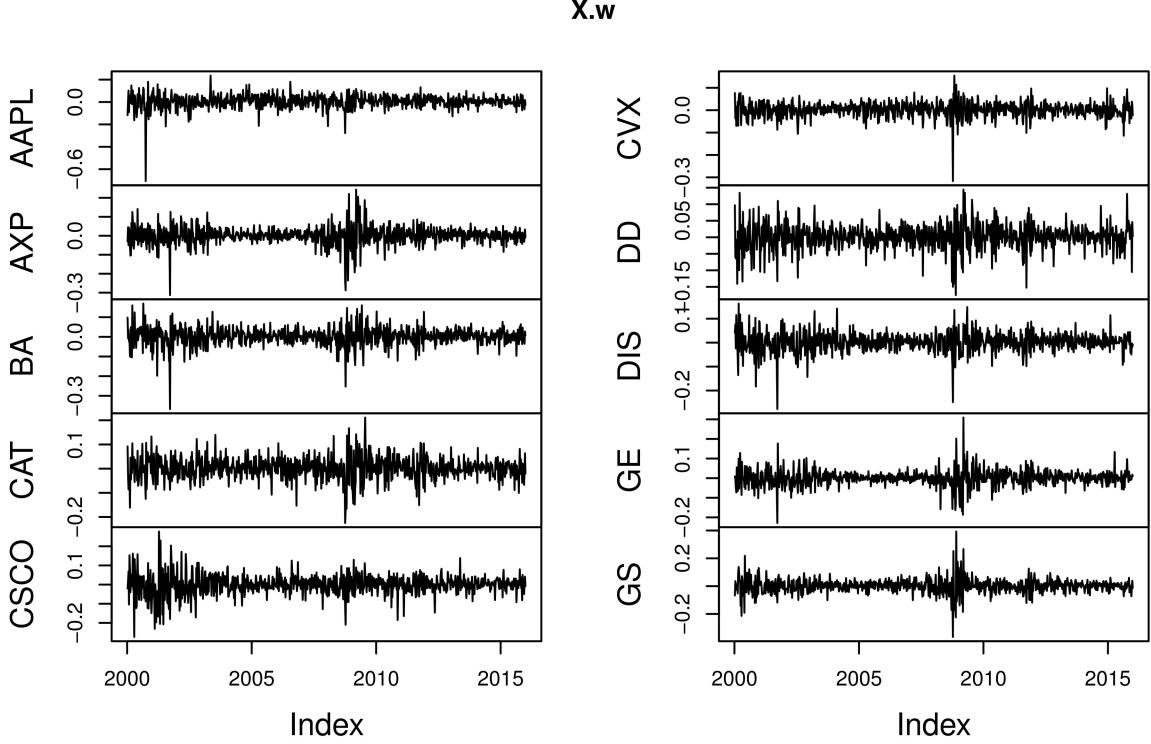
- Compute the daily and weekly log-returns using the function `returns`
- Plot the daily (and weekly) log-returns of each constituent.

```
d <- 10 # for the number of components we pick out
S <- DJ_const['2000-01-01/', 1:d] # first d components from January 1st 2000 onwards

## Daily log-returns
X.d <- returns(S)
plot.zoo(X.d)
```



```
## Weekly log-returns
X.w <- apply.weekly(X.d, FUN = colSums)
plot.zoo(X.w)
```



2) Is the return data jointly normal?

Our first step is to test the normality of the return data (first daily then weekly),

$$\mathcal{H}_0 : \mathbf{X}_1, \dots, \mathbf{X}_n \stackrel{iid}{\sim} N(\mu, \sigma)$$

2.1) Visual tests of normality

- We create scatterplots for each pair of constituents, to examine any possible resemblance to bivariate Gaussian distributions?
- Then, for each individual constituent, we draw QQ-plots against a standard Gaussian reference distribution.
- We define the squared Mahalanobis distance

$$D_i^2 = (\mathbf{X}_i - \bar{\mathbf{X}})' S^{-1} (\mathbf{X}_i - \bar{\mathbf{X}}); \quad i = 1, \dots, n$$

and the Mahalanobis angle between $(\mathbf{X}_i - \bar{\mathbf{X}})$ and $(\mathbf{X}_j - \bar{\mathbf{X}})$ is

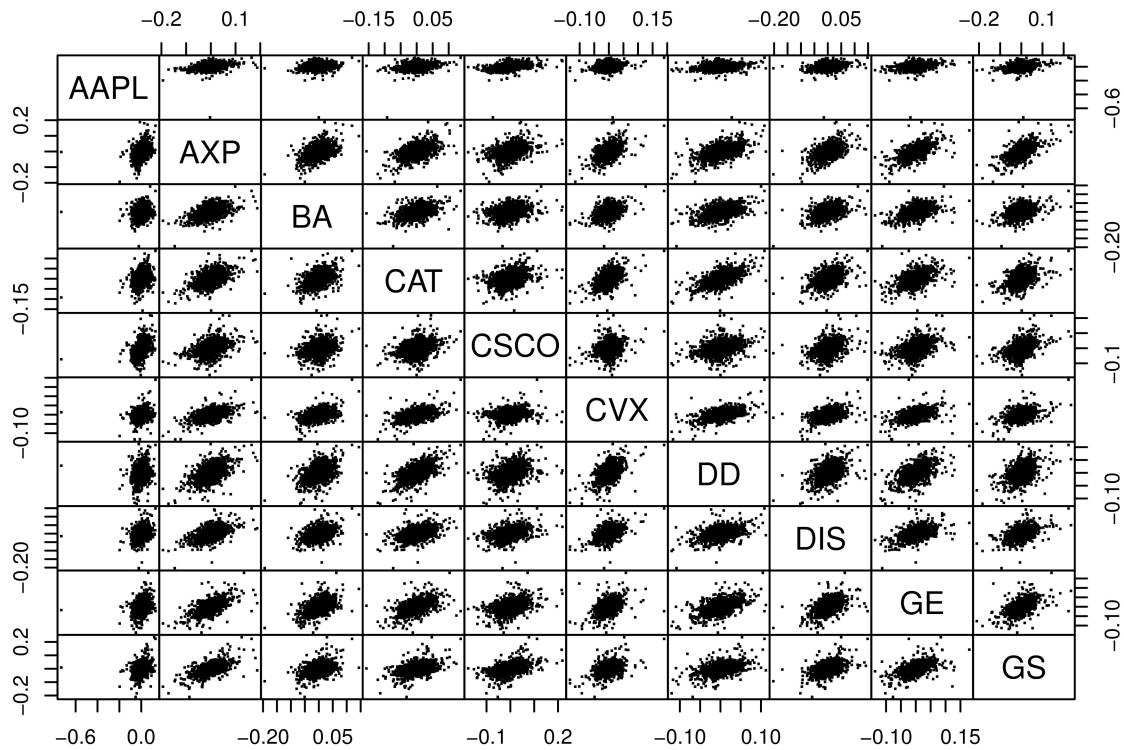
$$D_{ij}^2 = (\mathbf{X}_i - \bar{\mathbf{X}})' S^{-1} (\mathbf{X}_j - \bar{\mathbf{X}}); \quad i, j = 1, \dots, n$$

The marginal distribution of D_i^2 under the null hypothesis are not exactly chi-squared, although they converge to it for large values of n . We expect D_1^2, \dots, D_n^2 to behave roughly like an iid sample from the χ_d^2 distribution, so we test them against this reference distribution to check whether the original data $\mathbf{X}_1, \dots, \mathbf{X}_n$ are multivariate Gaussian.

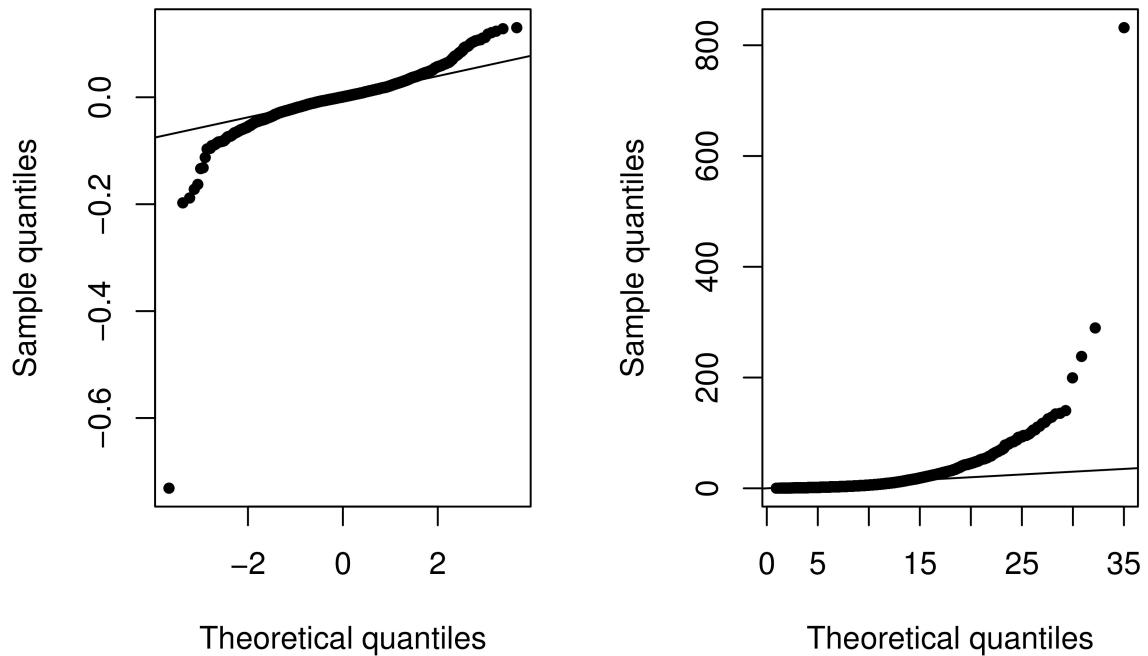
Finally, we compute the squared Mahalanobis distances D_1^2, \dots, D_n^2 and draw the corresponding QQ-plot.

Daily log-returns

```
pairs(as.matrix(X.d), gap = 0, pch = ".",lower=NULL) # visual assessment
```

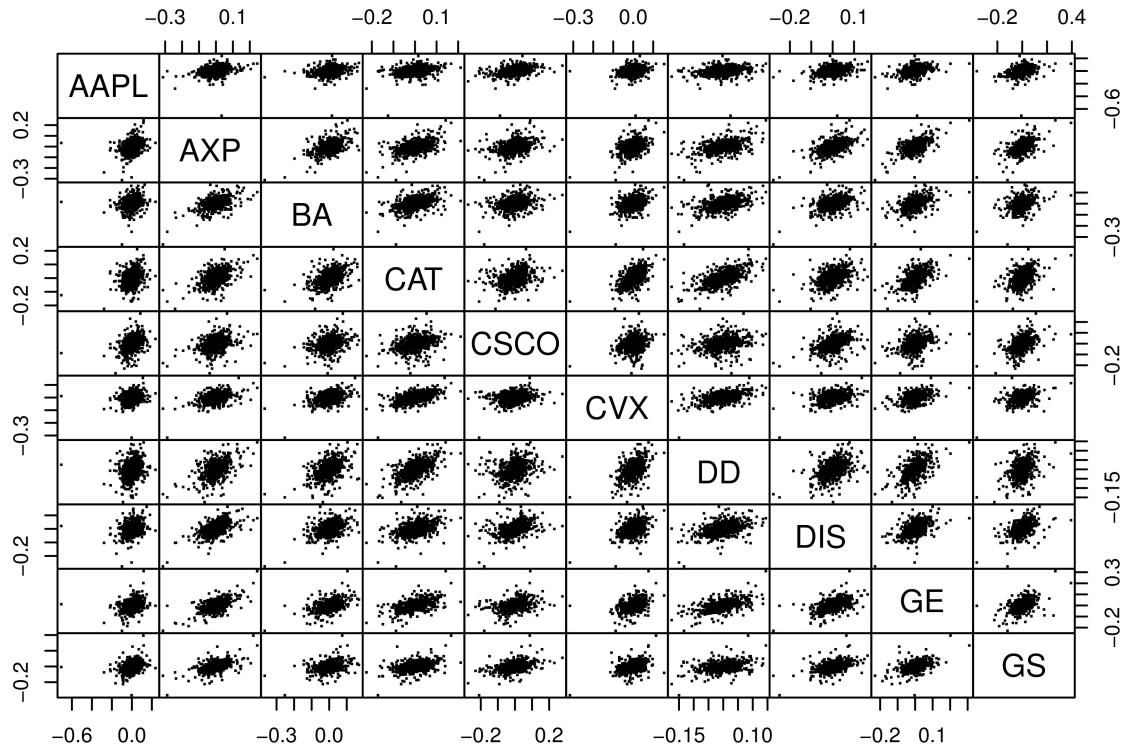


```
par(mfrow=c(1,2))
qq_plot(X.d[,1], FUN = qnorm, method = "empirical",pch=20) # first margin only => already not normal
D2.d <- mahalanobis(X.d, center = colMeans(X.d), cov = cov(X.d)) # squared Mahalanobis distances
qq_plot(D2.d, FUN = function(p) qchisq(p, df = d),pch=20) # => departure clearly visible
```



Weekly log-returns

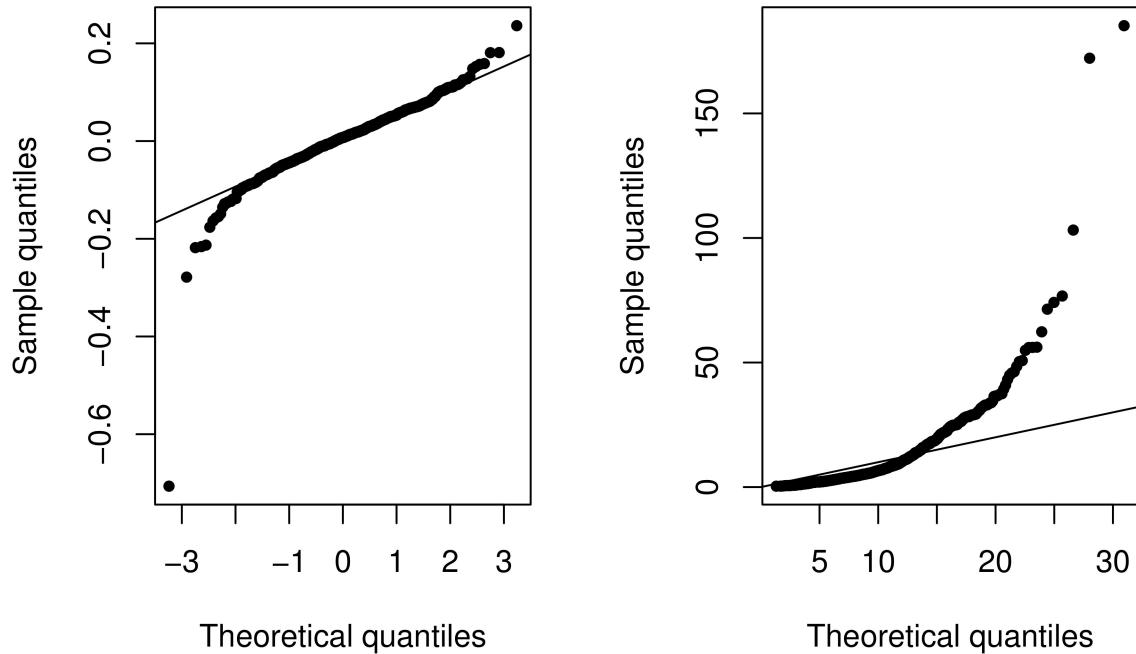
```
pairs(as.matrix(X.w), gap = 0, pch = ".") # visual assessment
```



```

par(mfrow=c(1,2))
qq_plot(X.w[,1], FUN = qnorm, method = "empirical",pch=20) # better but still not too good
D2.w <- mahalanobis(X.w, center = colMeans(X.w), cov = cov(X.w)) # squared Mahalanobis distances
qq_plot(D2.w, FUN = function(p) qchisq(p, df = d),pch=20) # => departure clearly visible

```



2.2) Formal tests of normality We will now conduct formal tests based on the return data themselves and the corresponding squared Mahalanobis distances and angles by:

- Checking the marginal normality of each constituent using the shapiro test.
 - Checking the joint normality of the data:
1. Testing whether D_1^2, \dots, D_n^2 are χ_d^2 using the Anderson-Darling test.
 2. Using the Mardia test of skewness and kurtosis (defined as functions of D_i^2 and D_{ij}^2)

Based on the plots and the tests p -values, we can see that the multivariate normal is not a particularly suitable model for the data, which of example shows fatter tails.

Daily log-returns

```
apply(X.d, 2, function(x) shapiro.test(x)$p.value) # tests of marginal normality

##          AAPL          AXP          BA          CAT          CSCO          CVX 
## 1.123225e-53 7.557251e-47 1.923304e-34 5.628375e-34 2.114241e-45 2.082424e-41 
##          DD          DIS          GE          GS 
## 6.720522e-39 2.922888e-42 1.628754e-45 6.515510e-47 

maha2_test(X.d) # Anderson--Darling test of squared Mahalanobis distances being chi_d^2

## 
## Anderson-Darling GoF Test
## 
## data: D2 and pchisq
## AD = Inf, p-value = 1.491e-07
## alternative hypothesis: NA
```

```

mardia_test(X.d) # Mardia's kurtosis test of joint normality based on squared Mahalanobis distances

##
##  Mardia's kurtosis test (computed with method = 'direct')
##
## data: X.d
## statistic = 756.63, p-value < 2.2e-16
## alternative hypothesis: two-sided

mardia_test(X.d, type = "skewness") # Mardia's skewness test of joint normality based on Mahalanobis and

##
##  Mardia's skewness test (computed with method = 'direct')
##
## data: X.d
## statistic = 26782, p-value < 2.2e-16
## alternative hypothesis: one-sided

Weekly log-returns

apply(X.w, 2, function(x) shapiro.test(x)$p.value) # tests of marginal normality

##          AAPL         AXP          BA          CAT          CSCO          CVX
## 2.523832e-24 2.175050e-21 2.404761e-19 4.184531e-10 4.774746e-17 7.576028e-18
##          DD          DIS          GE          GS
## 4.337361e-13 6.625413e-17 6.408195e-21 6.903104e-22

maha2_test(X.w) # Anderson-Darling test of squared Mahalanobis distances being chi_d^2

##
##  Anderson-Darling GoF Test
##
## data: D2 and pchisq
## AD = Inf, p-value = 7.186e-07
## alternative hypothesis: NA

mardia_test(X.w) # Mardia's kurtosis test of joint normality based on squared Mahalanobis distances

##
##  Mardia's kurtosis test (computed with method = 'direct')
##
## data: X.w
## statistic = 150.52, p-value < 2.2e-16
## alternative hypothesis: two-sided

mardia_test(X.w, type = "skewness") # Mardia's skewness test of joint normality based on Mahalanobis and

##
##  Mardia's skewness test (computed with method = 'direct')
##
## data: X.w
## statistic = 3238.7, p-value < 2.2e-16
## alternative hypothesis: one-sided

```

3) Fit various multivariate models

Now, we fit generalized hyperbolic models to the (daily then weekly) return data.

We will consider different subclasses - t , hyperbolic nad NIG distributions- with a symmetry constraint (meaning the skewness parameter γ is set to 0) then in the generalized form (where the skewness parameter γ varies).

```
max.iter <- 1e4 # maximal number of iterations for the fitting procedures
```

3.1) Symmetric models Daily log-returns

```
fit.t.sym.d <- fit.tmv (X.d, symmetric = TRUE, nit = max.iter, silent = TRUE) # t
fit.NIG.sym.d <- fit.NIGmv (X.d, symmetric = TRUE, nit = max.iter, silent = TRUE) # normal inverse Gaus
fit.GH.sym.d <- fit.ghypmv(X.d, symmetric = TRUE, nit = max.iter, silent = TRUE) # generalized hyperbo
```

Weekly log-returns

```
fit.t.sym.w <- fit.tmv (X.w, symmetric = TRUE, nit = max.iter, silent = TRUE) # t
fit.NIG.sym.w <- fit.NIGmv (X.w, symmetric = TRUE, nit = max.iter, silent = TRUE) # normal inverse Gaus
fit.GH.sym.w <- fit.ghypmv(X.w, symmetric = TRUE, nit = max.iter, silent = TRUE) # generalized hyperbol
```

3.2) Skewed models Daily log-returns

```
fit.t.skw.d <- fit.tmv (X.d, symmetric = FALSE, nit = max.iter, silent = TRUE) # t
fit.NIG.skw.d <- fit.NIGmv (X.d, symmetric = FALSE, nit = max.iter, silent = TRUE) # normal inverse Gau
fit.GH.skw.d <- fit.ghypmv(X.d, symmetric = FALSE, nit = max.iter, silent = TRUE) # generalized hyperb
```

Weekly log-returns

```
fit.t.skw.w <- fit.tmv (X.w, symmetric = FALSE, nit = max.iter, silent = TRUE) # t
fit.NIG.skw.w <- fit.NIGmv (X.w, symmetric = FALSE, nit = max.iter, silent = TRUE) # normal inverse Gau
fit.GH.skw.w <- fit.ghypmv(X.w, symmetric = FALSE, nit = max.iter, silent = TRUE) # generalized hyperb
```

4) Likelihood-ratio tests for comparing the goodness-of-fit of two models

In this section, we will compare the goodness-of-fit of the models via the log-likelihood ratio test,

\mathcal{H}_0 : Model 1 is no better than Model 2.

We compute the log-likelihood ratio statistics to compare

- Symmetric versus skewed t models
- Symmetric versus skewed hyperbolic models
- Symmetric t versus skewed NIG models

Compute the corresponding p -values and conclude.

Daily log-returns

```
likelihoods.d <- c(fit.t.sym.d@llh, fit.NIG.sym.d@llh, fit.GH.sym.d@llh,
                     fit.t.skw.d@llh, fit.NIG.skw.d@llh, fit.GH.skw.d@llh)
which.max(likelihoods.d) # => skewed generalized hyperbolic
```

```
## [1] 6
LRstat <- 2 * (fit.t.skw.d@llh - fit.t.sym.d@llh) # test statistic
1 - pchisq(LRstat, 10) # => H0 not rejected at 5% level

## [1] 0.7547205
LRstat <- 2 * (fit.GH.skw.d@llh - fit.GH.sym.d@llh) # test statistic
1 - pchisq(LRstat, 10) # => H0 not rejected at 5% level
```

```

## [1] 0.7284726
LRstat <- 2 * (fit.GH.sym.d@llh - fit.t.sym.d@llh) # test statistic
1 - pchisq(LRstat, 1) # => H0 rejected at 5% level => symmetric generalized hyperbolic preferred

## [1] 8.056387e-05

Weekly log-returns

likelihoods.w <- c(fit.t.sym.w@llh, fit.NIG.sym.w@llh, fit.GH.sym.w@llh,
                     fit.t.skw.w@llh, fit.NIG.skw.w@llh, fit.GH.skw.w@llh)
which.max(likelihoods.w) # => also skewed generalized hyperbolic

## [1] 6

LRstat <- 2 * (fit.t.skw.w@llh - fit.t.sym.w@llh) # test statistic
1 - pchisq(LRstat, 10) # => H0 not rejected at 5% level

## [1] 0.2992959

LRstat <- 2 * (fit.GH.skw.w@llh - fit.GH.sym.w@llh) # test statistic
1 - pchisq(LRstat, 10) # => H0 not rejected at 5% level

## [1] 0.2712829

LRstat <- 2 * (fit.GH.sym.w@llh - fit.t.sym.w@llh) # test statistic
1 - pchisq(LRstat, 1) # => H0 rejected at 5% level => symmetric generalized hyperbolic preferred

## [1] 0.02429803

```

5) Visual assessment

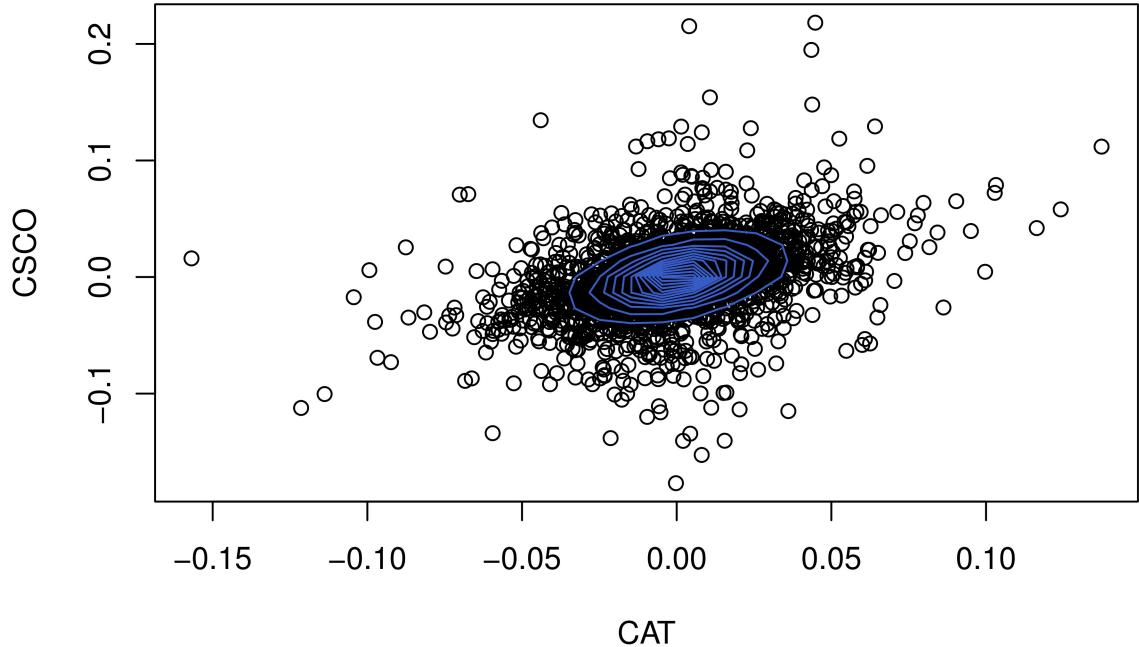
We now visually compare two constituents against the corresponding pairwise implied contour lines

```

X <- as.matrix(X.d) # data we consider

ind <- c(4, 5) # "CAT", "CSCO"
X. <- X[,ind]
plot(X.)
obj <- fit.GH.sym.d # get d-dimensional object
obj@mu <- obj@mu[ind]
obj@sigma <- obj@sigma[ind, ind]
obj@gamma <- obj@gamma[ind[2]] # (0 in symmetric case)
x. <- seq(min(X.[,1]), max(X.[,1]), length.out = 30) # x-locations of density evaluation points
y. <- seq(min(X.[,2]), max(X.[,2]), length.out = 30) # y-locations of density evaluation points
z. <- outer(x., y., function(xx, yy)
  dghyp(cbind(xx, yy), object = obj)) # implied bivariate density
contour(x., y., z = z., drawlabels = FALSE, axes = FALSE, col = "royalblue3", add = TRUE)

```



```

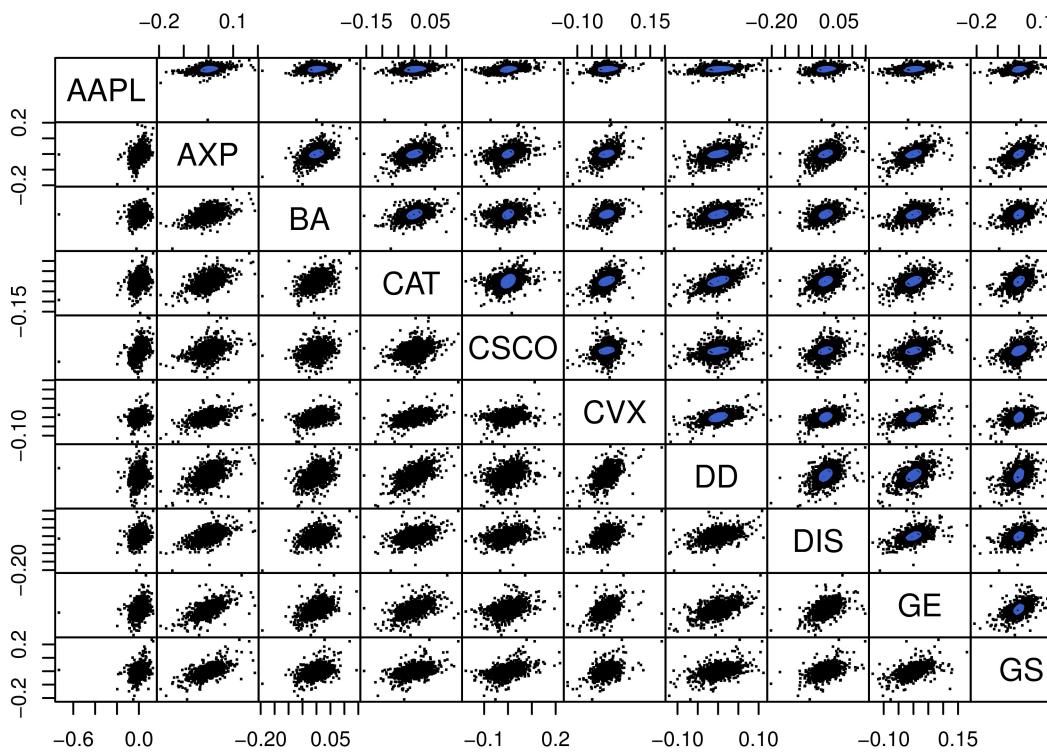
row <- 1 # row number in upper-right panel
col <- 1 # column number in upper-right panel
pairs(X, gap = 0, pch = ".",
      upper.panel = function(x, y, ...) {
        ## Figure out in which row and column we are in the plot (needed later)
        row <- row + 1
        if(row >= col) {
          col <- col + 1
          row <- 1
        }
        ## Plot the points with indices (i, j)
        points(x, y, ...)
        ## Extract the (i, j)th 'margin' of the object of class 'ghyp'
        obj <- fit.GH.sym.d # get d-dimensional object
        ind <- c(row, col) # pairwise indices to consider
        obj@mu <- obj@mu[ind]
        obj@sigma <- obj@sigma[ind, ind]
        obj@gamma <- obj@gamma[col] # (0 in symmetric case)
        ## Compute contours based on
        x. <- seq(min(x), max(x), length.out = 30) # x-locations of density evaluation points
        y. <- seq(min(y), max(y), length.out = 30) # y-locations of density evaluation points
        z. <- outer(x., y., function(xx, yy)
                    dghyp(cbind(xx, yy), object = obj)) # implied bivariate density
        ## Plot contours

```

```

    contour(x., y., z = z., nlevels = 5,
            drawlabels = FALSE, axes = FALSE, col = "royalblue3", add = TRUE)
  })

```



Similarly for all d pairs

References This project is basesd on R code from QRMtutorials.org