Portfolio Optimization based on Alternative Risk Measures with R

MAFS6010R - Portfolio Optimization with R MSc in Financial Mathematics Fall 2018-19, HKUST, Hong Kong

Prof. Daniel P. Palomar

Hong Kong University of Science and Technology (HKUST)

This R session will introduce portfolio optimization based on alternative risk measures to variance and compare with Markowitz portfolio.

(Useful R links: Cookbook R (http://www.cookbook-r.com/), Quick-R (https://www.statmethods.net/), R documentation (https://www.rdocumentation.org/), CRAN (https://cran.r-project.org/), METACRAN (https://r-pkg.org/).)

Loading market data

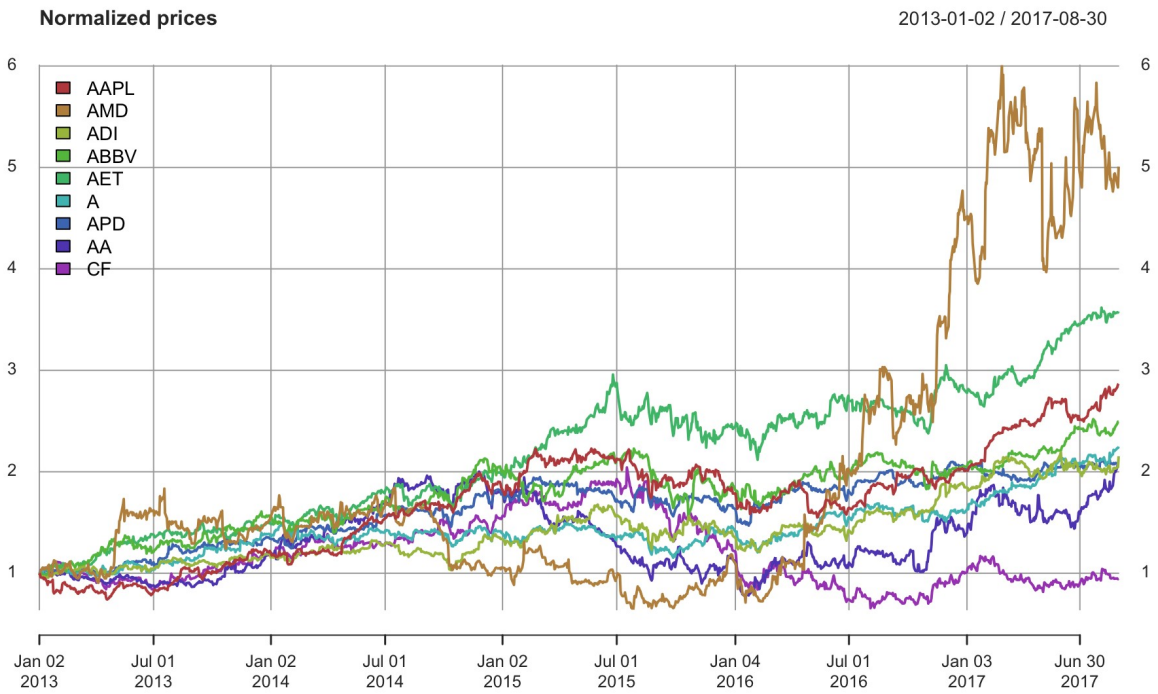
In this section, we will divide the stock market data into a training part (for the estimation of the expected return $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, and subsequent portfolio design) and a test part (for the out-of-sample performance evaluation).

We start by loading some stock market data:

```
library(xts)
library(quantmod)
library(PerformanceAnalytics)

# set begin-end date and stock namelist
begin_date <- "2013-01-01"
end_date <- "2017-08-31"
stock_namelist <- c("AAPL", "AMD", "ADI",  "ABBV", "AET", "A",  "APD", "AA","CF")

# download data from YahooFinance
prices <- xts()
for (stock_index in 1:length(stock_namelist))
  prices <- cbind(prices, Ad(getSymbols(stock_namelist[stock_index],
                                        from = begin_date, to = end_date, auto.assign = FALSE)))
colnames(prices) <- stock_namelist
indexClass(prices) <- "Date"
str(prices)
#> An 'xts' object on 2013-01-02/2017-08-30 containing:
#>   Data: num [1:1175, 1:9] 56.1 55.4 53.9 53.6 53.7 ...
#>  - attr(*, "dimnames")=List of 2
#>   ..$ : NULL
#>   ..$ : chr [1:9] "AAPL" "AMD" "ADI" "ABBV" ...
#>   Indexed by objects of class: [Date] TZ: UTC
#>   xts Attributes:
#>  NULL
head(prices)
#>                AAPL  AMD      ADI     ABBV      AET        A      APD
#> 2013-01-02 56.11887 2.53 37.90372 28.62027 43.32114 28.16434 67.74826
#> 2013-01-03 55.41053 2.49 37.29209 28.38393 42.40319 28.26521 67.51154
#> 2013-01-04 53.86707 2.59 36.62878 28.02536 42.59989 28.82338 68.41895
#> 2013-01-07 53.55021 2.67 36.74078 28.08241 43.23684 28.61492 68.35583
#> 2013-01-08 53.69434 2.67 36.36173 27.47122 41.75045 28.38627 68.48207
#> 2013-01-09 52.85514 2.63 36.26697 27.62606 42.31490 29.15291 69.40527
#>                 AA       CF
#> 2013-01-02 20.62187 29.72212
#> 2013-01-03 20.80537 29.58158
#> 2013-01-04 21.24121 30.24417
#> 2013-01-07 20.87419 30.13087
#> 2013-01-08 20.87419 29.68914
#> 2013-01-09 20.82831 30.72749
tail(prices)
#>                AAPL   AMD      ADI     ABBV      AET        A      APD
#> 2017-08-23 157.5971 12.48 77.08375 69.09010 154.8036 62.21310 140.5500
#> 2017-08-24 156.8977 12.50 77.27874 69.66007 154.3686 62.12389 140.6178
#> 2017-08-25 157.4789 12.43 76.98628 70.01749 154.3192 62.34195 141.3730
#> 2017-08-28 159.0649 12.23 77.44447 70.82896 154.7443 62.89698 141.1504
#> 2017-08-29 160.4835 12.15 77.55172 71.37959 154.6356 62.92671 140.7534
#> 2017-08-30 160.9169 12.67 81.61696 71.40857 155.1101 63.33307 140.8889
#>               AA       CF
#> 2017-08-23 41.06 28.08939
#> 2017-08-24 41.34 28.18649
#> 2017-08-25 41.21 28.13794
#> 2017-08-28 42.17 28.18649
#> 2017-08-29 43.00 27.99230
#> 2017-08-30 43.09 28.09910

# compute log-returns and linear returns
X_log <- diff(log(prices))[-1]
X_lin <- (prices/lag(prices) - 1)[-1]

# or alternatively...
X_log <- CalculateReturns(prices, "log")[-1]
X_lin <- CalculateReturns(prices)[-1]

N <- ncol(X_log)  # number of stocks
T <- nrow(X_log)  # number of days
```

We can take a look at the prices of the selected stocks:

```
plot(prices/rep(prices[1, ], each = nrow(prices)), col = rainbow10equal, legend.loc = "topleft",
     main = "Normalized prices")
```



We now divide the data into a training set and test set:

```
# split data into training and set data
T_trn <- round(0.7*T)  # 70% of data
X_log_trn <- X_log[1:T_trn, ]
X_log_tst <- X_log[(T_trn+1):T, ]
X_lin_trn <- X_lin[1:T_trn, ]
X_lin_tst <- X_lin[(T_trn+1):T, ]
```

We can now use the training set to obtain the sample estimates from the returns $\mathbf{x}_t$ (i.e., sample means and sample covariance matrix) as

$$\hat{\boldsymbol{\mu}} = \frac{1}{T}\sum_{t=1}^{T}\mathbf{x}_t$$

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{T-1}\sum_{t=1}^{T}(\mathbf{x}_t - \hat{\boldsymbol{\mu}})(\mathbf{x}_t - \hat{\boldsymbol{\mu}})^T$$

```
mu <- colMeans(X_log_trn)
Sigma <- cov(X_log_trn)
```

Warm-up: Markowitz portfolio

The mean-variance Markowitz portfolio with no shorting is

$$\begin{aligned} \underset{\mathbf{w}}{\text{maximize}} \quad & \mathbf{w}^T\boldsymbol{\mu} - \lambda\mathbf{w}^T\boldsymbol{\Sigma}\mathbf{w} \\ \text{subject to} \quad & \mathbf{1}^T\mathbf{w} = 1 \\ & \mathbf{w} \geq \mathbf{0}. \end{aligned}$$

For completeness, we can also consider the Global Minimum Variance Portfolio (GMVP), which doesn't make use of $\boldsymbol{\mu}$:

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimize}} \quad & \mathbf{w}^T \boldsymbol{\Sigma} \mathbf{w} \\ \text{subject to} \quad & \mathbf{1}^T \mathbf{w} = 1 \\ & \mathbf{w} \geq \mathbf{0}. \end{aligned}$$

We can compute the optimal $\mathbf{w}$ with a solver (the closed-form solution does not exist with the constraint $\mathbf{w} \geq \mathbf{0}$):

```
library(CVXR)

portolioMarkowitz <- function(mu, Sigma, lmd = 0.5) {
  w <- Variable(nrow(Sigma))
  prob <- Problem(Maximize(t(mu) %*% w - lmd*quad_form(w, Sigma)),
                  constraints = list(w >= 0, sum(w) == 1))
  result <- solve(prob)
  return(as.vector(result$getValue(w)))
}

portolioGMVP <- function(Sigma) {
  w <- Variable(nrow(Sigma))
  prob <- Problem(Minimize(quad_form(w, Sigma)),
                  constraints = list(w >= 0, sum(w) == 1))
  result <- solve(prob)
  return(as.vector(result$getValue(w)))
}

w_Markowitz <- portolioMarkowitz(mu, Sigma)
w_GMVP <- portolioGMVP(Sigma)
```

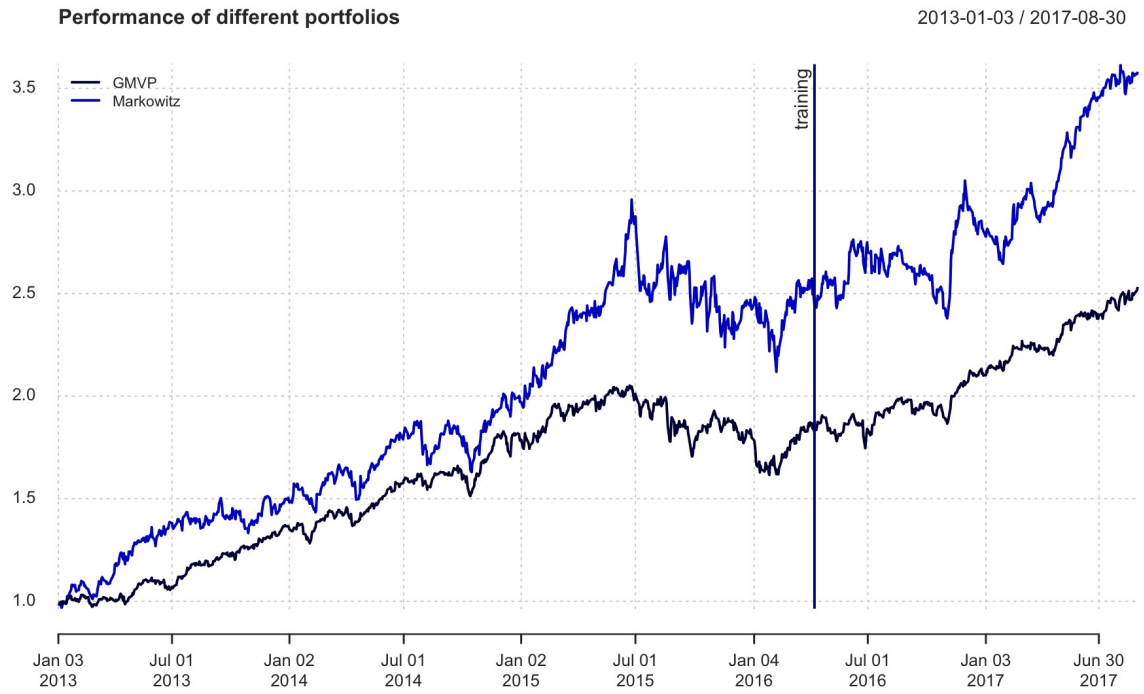
Then we can assess the performance (in sample vs out-of-sample):

```
# combine portfolios
w_all <- cbind("GMVP" = w_GMVP,
               "Markowitz" = w_Markowitz)
rownames(w_all) <- colnames(X_lin)

# compute returns of all portfolios
ret_all <- xts(X_lin %*% w_all, index(X_lin))
ret_all_trn <- ret_all[1:T_trn, ]
ret_all_tst <- ret_all[-c(1:T_trn), ]

# performance
table.AnnualizedReturns(ret_all_trn)
#>                             GMVP Markowitz
#> Annualized Return         0.2041    0.3154
#> Annualized Std Dev        0.1622    0.2503
#> Annualized Sharpe (Rf=0%) 1.2584    1.2599
table.AnnualizedReturns(ret_all_tst)
#>                             GMVP Markowitz
#> Annualized Return         0.2604    0.3138
#> Annualized Std Dev        0.1234    0.1860
#> Annualized Sharpe (Rf=0%) 2.1092    1.6875
```

We can see that the mean-variance Markowitz portfolio performs even worse than the GMVP in the out-of-sample (the in-sample Sharpe ratio is approximately the same though).

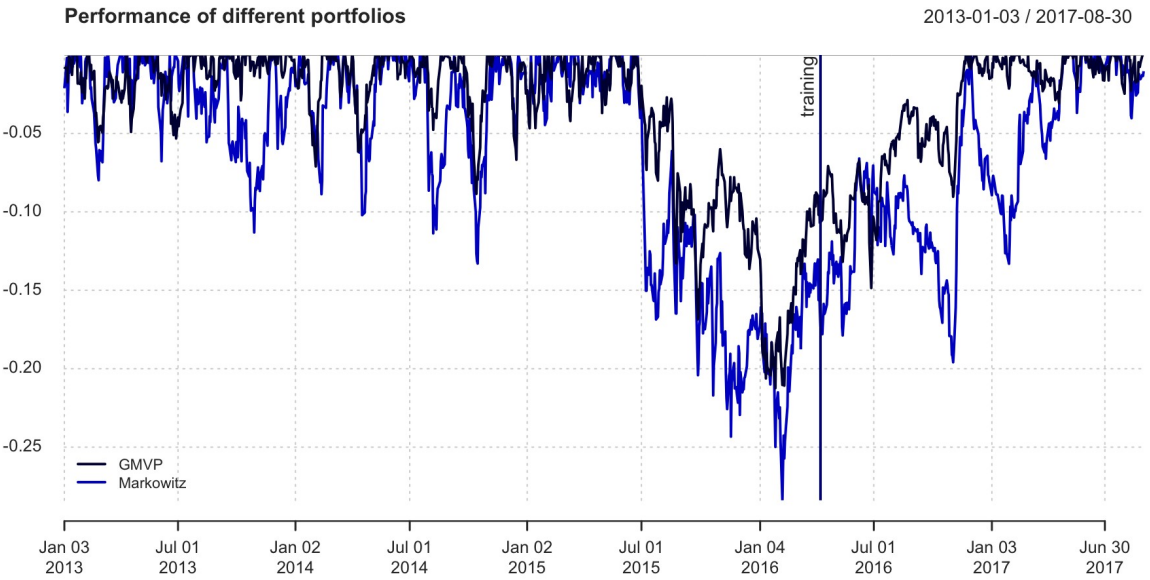
Let us plot the cumulative PnL over time:

```
{ chart.CumReturns(ret_all, main = "Performance of different portfolios",
                   wealth.index = TRUE, legend.loc = "topleft", colorset = rich8equal)
  addEventLines(xts("training", index(X_lin[T_trn])), srt=90, pos=2, lwd = 2, col = "darkblue") }
```



The cum PnL may seem contradictory at first because the mean-variance portfolio seems to be doing much better than the GMVP. This is however a visual effect. The drawdown is very instructive:

```
{ chart.Drawdown(ret_all, main = "Performance of different portfolios",
                 legend.loc = "bottomleft", colorset = rich8equal)
  addEventLines(xts("training", index(X_lin[T_trn])), srt=90, pos=2, lwd = 2, col = "darkblue") }
```



We can see that the drawdown of the mean-variance Markowitz portfolio is indeed much worse than that of the GMVP.

Alternative measures of risk

The Markowitz porftolio, while it started the field of modern portfolio theory in 1952, has not been embraced by practitioners because of several reasons. One of the main drawbacks is that the variance $\mathbf{w}^T \mathbf{\Sigma} \mathbf{w}$ is not a good measure of risk in practice since it penalizes both the unwanted high losses and the desired low losses (or gains).

Indeed, the mean-variance portfolio framework penalizes up-side and down-side risk equally, whereas most investors don't mind up-side risk.

To overcome the limitations of the variance as risk measure, a number of alternative risk measures have been proposed, for example:

- **Downside Risk (DR)**: Let R be a random variable representing the return. The lower partial moments (LPM) is defined as

$$\mathsf{LPM} = \mathsf{E}\left[\left((\tau - R)^+\right)^\alpha\right],$$

where $(\cdot)^+ = \max(0, \cdot)$ and we will use $\tau = \mathsf{E}[R]$ as disaster level. The parameter α allows different levels of risk-aversion:

  - $\alpha = 1$ suits a neutral investor:

$$\mathsf{E}\left[(E[R] - R)^+\right]$$

  - $\alpha = 2$ is more risk-averse and yields the semivariance:

$$\mathsf{SV} = \mathsf{E}\left[\left((E[R] - R)^+\right)^2\right]$$

  - $\alpha = 3$ is even more risk-averse:

$$\mathsf{E}\left[\left((E[R] - R)^+\right)^3\right]$$

- **Value-at-Risk (VaR)**: Let ξ be a random variable representing the loss. The VaR is defined as

$$\mathsf{VaR}_\alpha = \inf\{\xi_0 : \mathsf{Pr}(\xi \le \xi_0) \ge \alpha\}$$

with α the confidence level, say, $\alpha = 0.95$.

- **Conditional Value-at-Risk (CVaR)**: The CVaR takes into account the shape of the losses exceeding the VaR through the average:

$$\mathsf{CVaR}_\alpha = \mathsf{E}[\xi \mid \xi \ge \mathsf{VaR}_\alpha].$$

- **Drawdown (DD)**: The drawdown (DD) at time t is defined as the decline from a historical peak of the cumulative profit $X(t)$:

$$D(t) = \frac{\mathsf{HWM}_t - X(t)}{\mathsf{HWM}_t}$$

where $\mathsf{HWM}_t = \max_{1 \le \tau \le t} X(\tau)$ is the high water mark of $X(t)$.

  - Then one can define the maximum DD (Max-DD) over a period $t = 1, \ldots, T$ as

$$M(T) = \max_{1 \le t \le T} D(t)$$

  - Also the average DD (Ave-DD) over a period $t = 1, \ldots, T$ as

$$A(T) = \frac{1}{T} \sum_{1 \le t \le T} D(t)$$

  - Similarly to the CVaR, we can define the Conditional Drawdown at Risk (CDaR) as the mean of the worst $100(1-\alpha)\%$ drawdowns.

Downside Risk portfolio

We will use sample approximation of the downside risk:

$$\begin{aligned} E\left[\left((E[R] - R)^+\right)^\alpha\right] &\approx \frac{1}{T} \sum_{t=1}^{T} \left((E[R] - R_t)^+\right)^\alpha \\ &\approx \frac{1}{T} \sum_{t=1}^{T} \left(\left(\frac{1}{T} \sum_{\tau=1}^{T} R_\tau - R_t\right)^+\right)^\alpha. \end{aligned}$$

The mean-downside risk portfolio formulation is the convex (well, depends on α) problem

$$\begin{array}{ll} \underset{\mathbf{w}}{\text{maximize}} & \mathbf{w}^T \boldsymbol{\mu} - \lambda \frac{1}{T} \sum_{t=1}^{T} \left(\left(\mathbf{w}^T \boldsymbol{\mu} - \mathbf{w}^T \mathbf{r}_t\right)^+\right)^\alpha \\ \text{subject to} & \mathbf{1}^T \mathbf{w} = 1 \\ & \mathbf{w} \ge \mathbf{0}. \end{array}$$

The mean-semivariance ($\alpha = 2$) portfolio formulation is the convex QP problem

$$\begin{array}{ll} \underset{\mathbf{w}}{\text{maximize}} & \mathbf{w}^T\boldsymbol{\mu} - \lambda \frac{1}{T}\sum_{t=1}^T \left(\left(\mathbf{w}^T\boldsymbol{\mu} - \mathbf{w}^T\mathbf{r}_t\right)^+ \right)^2 \\ \text{subject to} & \mathbf{1}^T\mathbf{w} = 1 \\ & \mathbf{w} \geq \mathbf{0}. \end{array}$$

Interestingly, for $\alpha = 1$ the problem is an LP:

$$\begin{array}{ll} \underset{\mathbf{w}}{\text{maximize}} & \mathbf{w}^T\boldsymbol{\mu} - \lambda \frac{1}{T}\sum_{t=1}^T \left(\mathbf{w}^T\boldsymbol{\mu} - \mathbf{w}^T\mathbf{r}_t\right)^+ \\ \text{subject to} & \mathbf{1}^T\mathbf{w} = 1 \\ & \mathbf{w} \geq \mathbf{0}. \end{array}$$

For $\alpha = 3$ the problem is not an LP or QP, but still convex.

As a curiosity, note that the mean-variance formulation can also be written similarly to the mean-semivariance:

$$\begin{array}{ll} \underset{\mathbf{w}}{\text{maximize}} & \mathbf{w}^T\boldsymbol{\mu} - \lambda \frac{1}{T}\sum_{t=1}^T \left(\mathbf{w}^T\boldsymbol{\mu} - \mathbf{w}^T\mathbf{r}_t\right)^2 \\ \text{subject to} & \mathbf{1}^T\mathbf{w} = 1 \\ & \mathbf{w} \geq \mathbf{0}. \end{array}$$

We can now compute the different portfolios $\mathbf{w}$ for different values of α with the **CVXR** package (see https://cvxr.rbind.io/cvxr_functions/ (https://cvxr.rbind.io/cvxr_functions/) for list of functions):

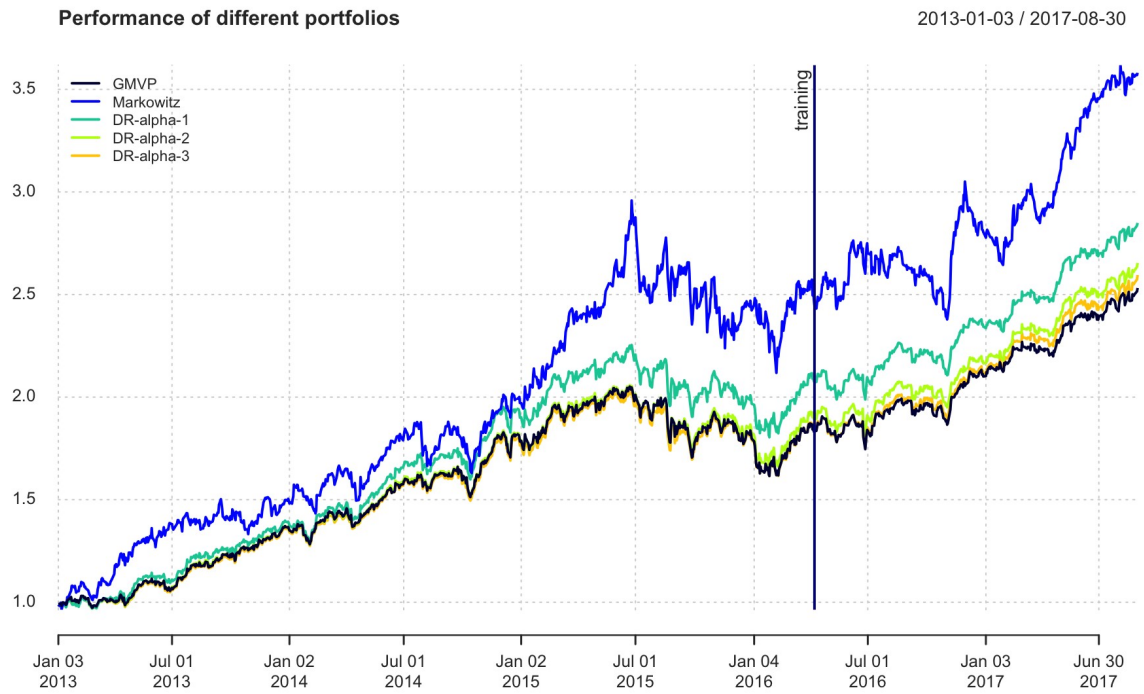
```
library(CVXR)

portfolioDR <- function(X, lmd = 0.5, alpha = 2) {
  T <- nrow(X)
  N <- ncol(X)
  X <- as.matrix(X)
  mu <- colMeans(X)
  w <- Variable(N)

  prob <- Problem(Maximize(t(w) %*% mu - (lmd/T) * sum(pos(t(mu) %*% w - X %*% w))^alpha),
                  constraints = list(w >= 0, sum(w) == 1))
  result <- solve(prob)
  return(as.vector(result$getValue(w)))
}

w_DR_alpha1 <- portfolioDR(X_log_trn, alpha = 1)
w_DR_alpha2 <- portfolioDR(X_log_trn, alpha = 2)
w_DR_alpha3 <- portfolioDR(X_log_trn, alpha = 3)
```

Let compare the performance (in sample vs out-of-sample):

```
# combine portfolios
w_all <- cbind(w_all,
               "DR-alpha-1" = w_DR_alpha1,
               "DR-alpha-2" = w_DR_alpha2,
               "DR-alpha-3" = w_DR_alpha3)

# compute returns of all portfolios
ret_all <- xts(X_lin %*% w_all, index(X_lin))
ret_all_trn <- ret_all[1:T_trn, ]
ret_all_tst <- ret_all[-c(1:T_trn), ]

# performance
t(table.AnnualizedReturns(ret_all_trn))
#>            Annualized Return Annualized Std Dev Annualized Sharpe (Rf=0%)
#> GMVP                  0.2041            0.1622                    1.2584
#> Markowitz             0.3154            0.2503                    1.2599
#> DR-alpha-1            0.2503            0.1674                    1.4947
#> DR-alpha-2            0.2159            0.1632                    1.3231
#> DR-alpha-3            0.2057            0.1628                    1.2630
t(table.AnnualizedReturns(ret_all_tst))
#>            Annualized Return Annualized Std Dev Annualized Sharpe (Rf=0%)
#> GMVP                  0.2604            0.1234                    2.1092
#> Markowitz             0.3138            0.1860                    1.6875
#> DR-alpha-1            0.2560            0.1169                    2.1906
#> DR-alpha-2            0.2742            0.1189                    2.3056
#> DR-alpha-3            0.2789            0.1217                    2.2922
```

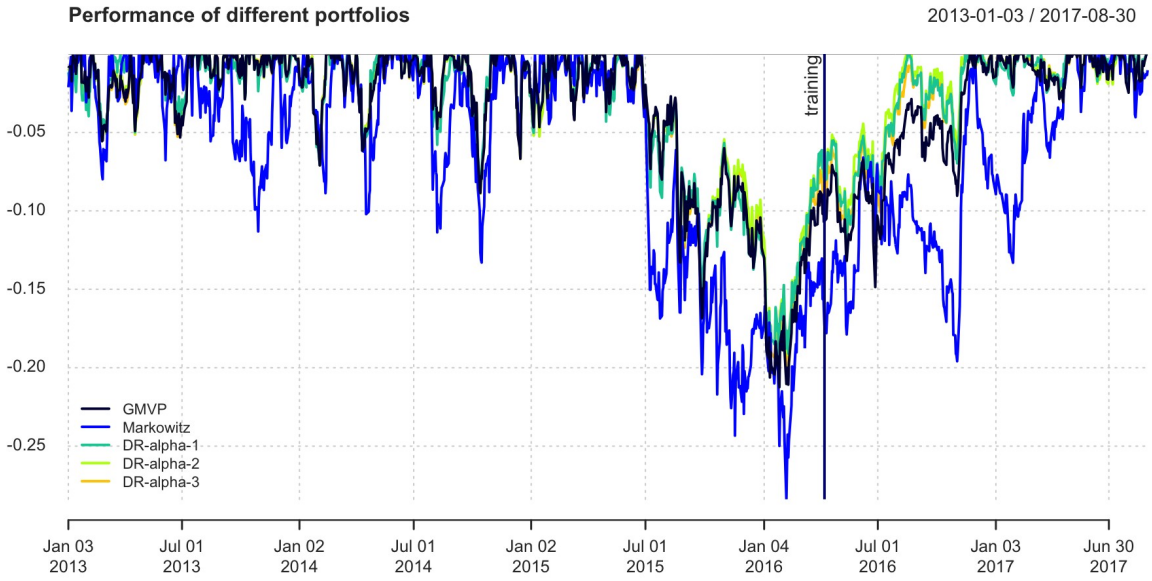
Let us plot the cumulative PnL over time:

```
{ chart.CumReturns(ret_all, main = "Performance of different portfolios",
                   wealth.index = TRUE, legend.loc = "topleft", colorset = rich6equal)
  addEventLines(xts("training", index(X_lin[T_trn])), srt=90, pos=2, lwd = 2, col = "darkblue") }
```



The cum PnL may seem contradictory at first because the mean-variance portfolio seems to be doing much better than the others. However, this is just a visual effect. The drawdown is very instructive:

```
{ chart.Drawdown(ret_all, main = "Performance of different portfolios",
                 legend.loc = "bottomleft", colorset = rich6equal)
  addEventLines(xts("training", index(X_lin[T_trn])), srt=90, pos=2, lwd = 2, col = "darkblue") }
```



Indeed, the mean-variance Markowitz portfolio has the worst drawdown. As for the downside risk, we will keep the one with $\alpha = 1$.

```
w_all <- w_all[, !colnames(w_all) %in% c("DR-alpha-2", "DR-alpha-3")]
```

CVaR portfolio

The CVaR can be conveniently obtained as the minimum of an auxiliary convex function[1]:

$$\text{CVaR}_\alpha\left(\mathbf{w}^T\mathbf{r}\right) = \min_\zeta F_\alpha\left(\mathbf{w}, \zeta\right)$$

where

$$F_\alpha(\mathbf{w}, \zeta) = \zeta + \frac{1}{1-\alpha}\mathsf{E}\left[-\mathbf{w}^T\mathbf{r} - \zeta\right]^+.$$

We can use a sample average approximation of $F_\alpha(\mathbf{w}, \zeta)$:

$$F_\alpha(\mathbf{w}, \zeta) \approx \zeta + \frac{1}{1-\alpha}\frac{1}{T}\sum_{t=1}^{T}\left[-\mathbf{w}^T\mathbf{r}_t - \zeta\right]^+.$$

We define the dummy variables z_t:

$$z_t \geq \left[-\mathbf{w}^T\mathbf{r}_t - \zeta\right]^+ \Longrightarrow z_t \geq -\mathbf{w}^T\mathbf{r}_t - \zeta,\ z_t \geq 0.$$

The mean-CVaR portfolio formulation can be finally written as the (convex) LP:

$$\begin{aligned} \underset{\mathbf{w},\mathbf{z},\zeta}{\text{maximize}} \quad & \mathbf{w}^T\boldsymbol{\mu} - \lambda\left(\zeta + \tfrac{1}{1-\alpha}\tfrac{1}{T}\textstyle\sum_{t=1}^{T} z_t\right) \\ \text{subject to} \quad & 0 \leq z_t \geq -\mathbf{w}^T\mathbf{r}_t - \zeta, \quad t = 1, \ldots, T \\ & \mathbf{1}^T\mathbf{w} = 1, \quad \mathbf{w} \geq \mathbf{0}. \end{aligned}$$

We are now ready to compute the mean-CVaR portfolio with the **CVXR** package:

```
portolioCVaR <- function(X, lmd = 0.5, alpha = 0.95) {
  T <- nrow(X)
  N <- ncol(X)
  X <- as.matrix(X)
  mu <- colMeans(X)
  # variables
  w <- Variable(N)
  z <- Variable(T)
  zeta <- Variable(1)
  # problem
  prob <- Problem(Maximize(t(w) %*% mu - lmd*zeta - (lmd/(T*(1-alpha))) * sum(z)),
                  constraints = list(z >= 0, z >= -X %*% w - zeta,
                                     w >= 0, sum(w) == 1))
  result <- solve(prob)
  return(as.vector(result$getValue(w)))
}

w_CVaR095 <- portolioCVaR(X_log_trn, alpha = 0.95)
w_CVaR099 <- portolioCVaR(X_log_trn, alpha = 0.99)
```

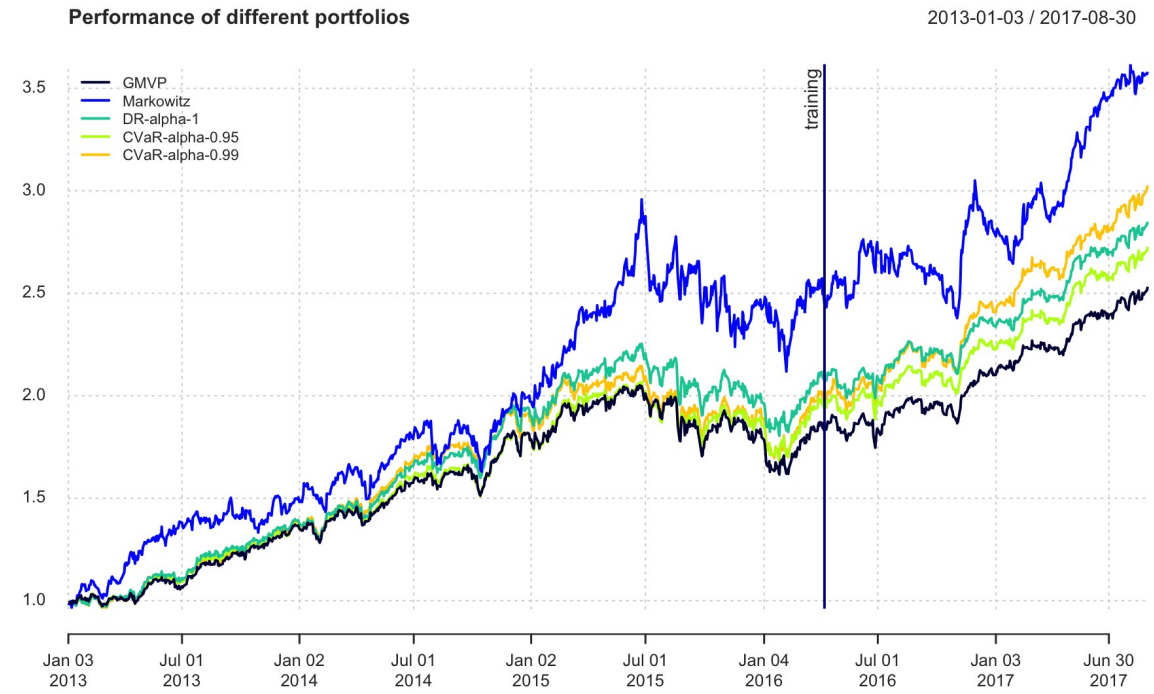
Let compare the performance (in sample vs out-of-sample):

```
# combine portfolios
w_all <- cbind(w_all,
               "CVaR-alpha-0.95" = w_CVaR095,
               "CVaR-alpha-0.99" = w_CVaR099)

# compute returns of all portfolios
ret_all <- xts(X_lin %*% w_all, index(X_lin))
ret_all_trn <- ret_all[1:T_trn, ]
ret_all_tst <- ret_all[-c(1:T_trn), ]

# performance
t(table.AnnualizedReturns(ret_all_trn)[3, ])
#>                 Annualized Sharpe (Rf=0%)
#> GMVP                               1.2584
#> Markowitz                          1.2599
#> DR-alpha-1                         1.4947
#> CVaR-alpha-0.95                    1.3543
#> CVaR-alpha-0.99                    1.3693
t(table.AnnualizedReturns(ret_all_tst)[3, ])
#>                 Annualized Sharpe (Rf=0%)
#> GMVP                               2.1092
#> Markowitz                          1.6875
#> DR-alpha-1                         2.1906
#> CVaR-alpha-0.95                    2.2988
#> CVaR-alpha-0.99                    2.7544
```

The CVaR with a confidence level of 99% has by far the highest Sharpe ratio. However, one cannot conclude too much from these results as more exhaustive backtests should be conducted.

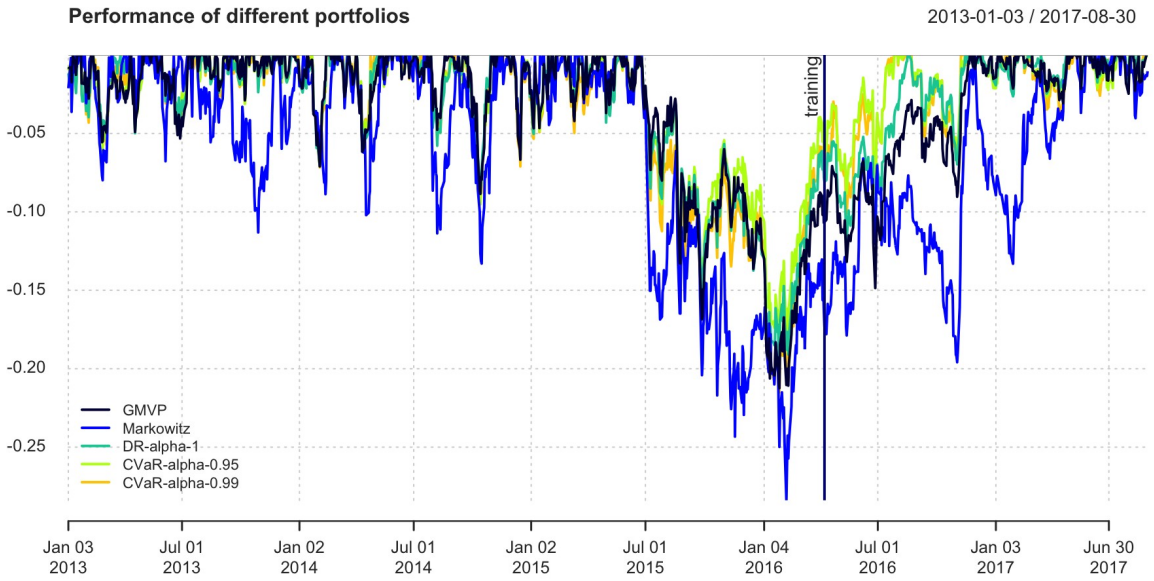
We can also observe its good performance from the cumulative PnL over time:

```
{ chart.CumReturns(ret_all, main = "Performance of different portfolios",
                   wealth.index = TRUE, legend.loc = "topleft", colorset = rich6equal)
  addEventLines(xts("training", index(X_lin[T_trn])), srt=90, pos=2, lwd = 2, col = "darkblue") }
```



And from the drawdown plot:

```
{ chart.Drawdown(ret_all, main = "Performance of different portfolios",
                 legend.loc = "bottomleft", colorset = rich6equal)
  addEventLines(xts("training", index(X_lin[T_trn])), srt=90, pos=2, lwd = 2, col = "darkblue") }
```



We will then keep the CVaR portfolio with $\alpha = 0.99$.

```
w_all <- w_all[, !colnames(w_all) %in% c("CVaR-alpha-0.95")]
```

Drawdown portfolio

The drawdown (DD) at time t can be written as

$$D(t) = \max_{1 \leq \tau \leq t} r_p^{\text{cum}}(\tau) - r_p^{\text{cum}}(t)$$

where $r_p^{\text{cum}}(t) = \mathbf{w}^T \mathbf{r}^{\text{cum}}(t)$ and $\mathbf{r}^{\text{cum}}(t) = \sum_{\tau=1}^{t} \mathbf{r}(\tau)$.

Then we can define the following possible risk measures:

- maximum DD (Max-DD) over a period $t = 1, \ldots, T$:

$$M(T) = \max_{1 \le t \le T} D(t)$$

- average DD (Ave-DD) over a period $t = 1, \ldots, T$:

$$A(T) = \frac{1}{T} \sum_{1 \le t \le T} D(t)$$

- similarly to the CVaR, we can define the Conditional Drawdown at Risk (CDaR) as the mean of the worst $100(1-\alpha)\%$ drawdowns:

$$\Delta_\alpha(T) = \frac{1}{(1-\alpha)T} \sum_{t \in \Omega_\alpha} D(t),$$

where $\Omega_\alpha = \{1 \le t \le T \mid D(t) \ge \xi_\alpha\}$ with ξ_α being the threshold such that exactly $100(1-\alpha)\%$ of drawdowns exceeds that limit. The CDaR can be conveniently expressed as

$$\Delta_\alpha(\mathbf{w}) = \min_\zeta \{\zeta + \frac{1}{1-\alpha} \frac{1}{T} \sum_{t=1}^{T} [D_t(\mathbf{w}) - \zeta]^+\}.$$

Let's now formulate the different drawdown portfolios and implement them with the **CVXR** package (see https://cvxr.rbind.io/cvxr_functions/ (https://cvxr.rbind.io/cvxr_functions/) for list of functions).

Max-DD portfolio

We can formulate the maximization of the expected return subject to a Max-DD constraint as

$$\begin{array}{ll} \underset{\mathbf{w}}{\text{maximize}} & \mathbf{w}^T \boldsymbol{\mu} \\ \text{subject to} & \max_{1 \le t \le T} \{\max_{1 \le \tau \le t} \mathbf{w}^T \mathbf{r}_\tau^{\text{cum}} - \mathbf{w}^T \mathbf{r}_t^{\text{cum}}\} \le c \\ & \mathbf{1}^T \mathbf{w} = 1, \quad \mathbf{w} \ge \mathbf{0}, \end{array}$$

which can be more conveniently rewritten as the following LP:

$$\begin{array}{ll} \underset{\mathbf{w}, \{u_t\}}{\text{maximize}} & \mathbf{w}^T \boldsymbol{\mu} \\ \text{subject to} & \mathbf{w}^T \mathbf{r}_t^{\text{cum}} \le u_t \le \mathbf{w}^T \mathbf{r}_t^{\text{cum}} + c, \quad \forall 1 \le t \le T \\ & u_{t-1} \le u_t \\ & \mathbf{1}^T \mathbf{w} = 1, \quad \mathbf{w} \ge \mathbf{0}. \end{array}$$

```
portfolioMaxDD <- function(X, c = 0.2) {
  T <- nrow(X)
  N <- ncol(X)
  X <- as.matrix(X)
  X_cum <- apply(X, MARGIN = 2, FUN = cumsum)
  mu <- colMeans(X)
  # variables
  w <- Variable(N)
  u <- Variable(T)
  # problem
  prob <- Problem(Maximize(t(w) %*% mu),
                  constraints = list(w >= 0, sum(w) == 1,
                                     u <= X_cum %*% w + c,
                                     u >= X_cum %*% w,
                                     u[-1] >= u[-T]))
  result <- solve(prob)
  return(as.vector(result$getValue(w)))
}

w_MaxDD_c018 <- portfolioMaxDD(X_log_trn, c = 0.18)
w_MaxDD_c021 <- portfolioMaxDD(X_log_trn, c = 0.21)
w_MaxDD_c024 <- portfolioMaxDD(X_log_trn, c = 0.24)
```

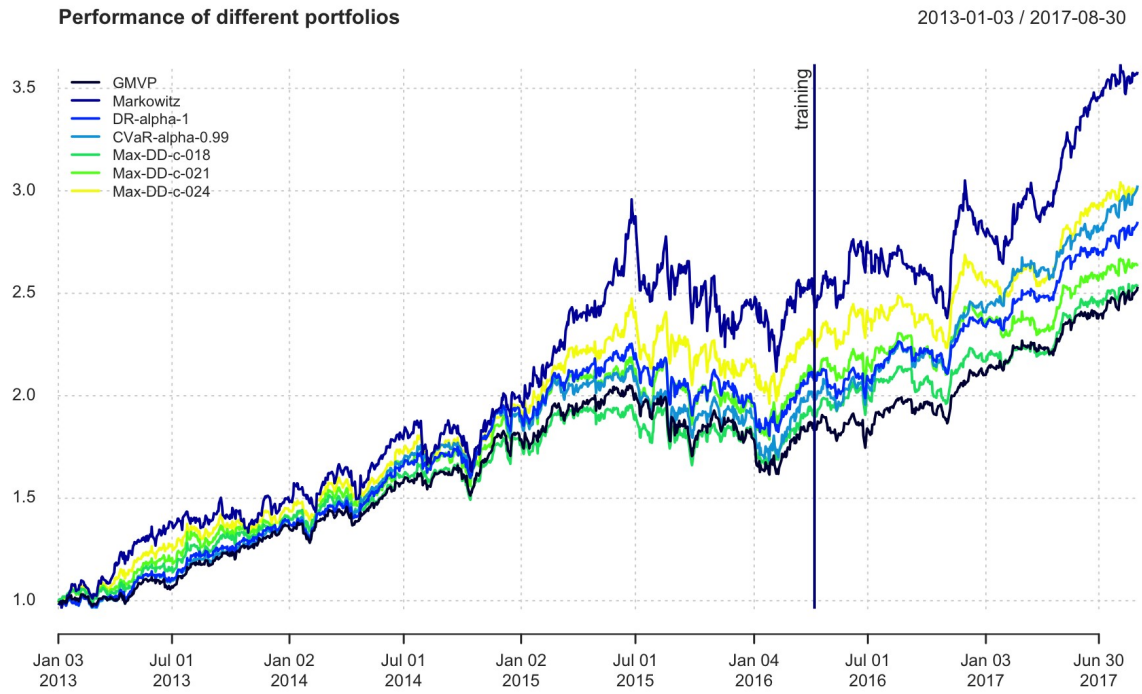
Let compare the performance (in sample vs out-of-sample):

```
# combine portfolios
w_all <- cbind(w_all,
               "Max-DD-c-018" = w_MaxDD_c018,
               "Max-DD-c-021" = w_MaxDD_c021,
               "Max-DD-c-024" = w_MaxDD_c024)

# compute returns of all portfolios
ret_all <- xts(X_lin %*% w_all, index(X_lin))
ret_all_trn <- ret_all[1:T_trn, ]
ret_all_tst <- ret_all[-c(1:T_trn), ]

# performance
t(table.AnnualizedReturns(ret_all_tst)[3, ])
#>                 Annualized Sharpe (Rf=0%)
#> GMVP                               2.1092
#> Markowitz                          1.6875
#> DR-alpha-1                         2.1906
#> CVaR-alpha-0.99                    2.7544
#> Max-DD-c-018                       1.8021
#> Max-DD-c-021                       1.4050
#> Max-DD-c-024                       1.6581
t(maxDrawdown(ret_all_trn))
#>                 Worst Drawdown
#> GMVP                 0.2123792
#> Markowitz            0.2838861
#> DR-alpha-1           0.1994105
#> CVaR-alpha-0.99      0.2151317
#> Max-DD-c-018         0.1583564
#> Max-DD-c-021         0.1831850
#> Max-DD-c-024         0.2073082
t(maxDrawdown(ret_all_tst))
#>                 Worst Drawdown
#> GMVP                0.08700167
#> Markowitz           0.13904334
#> DR-alpha-1          0.06968321
#> CVaR-alpha-0.99     0.07322740
#> Max-DD-c-018        0.08486694
#> Max-DD-c-021        0.08758975
#> Max-DD-c-024        0.10266292
```

We can see that the Max-DD designs indeed have a controlled maximum drawdown at least in-sample; however, out-of-sample it is not mantained. This is probably due to having not enough training samples. Also, the Sharpe ratio doesn't seem to be very good.

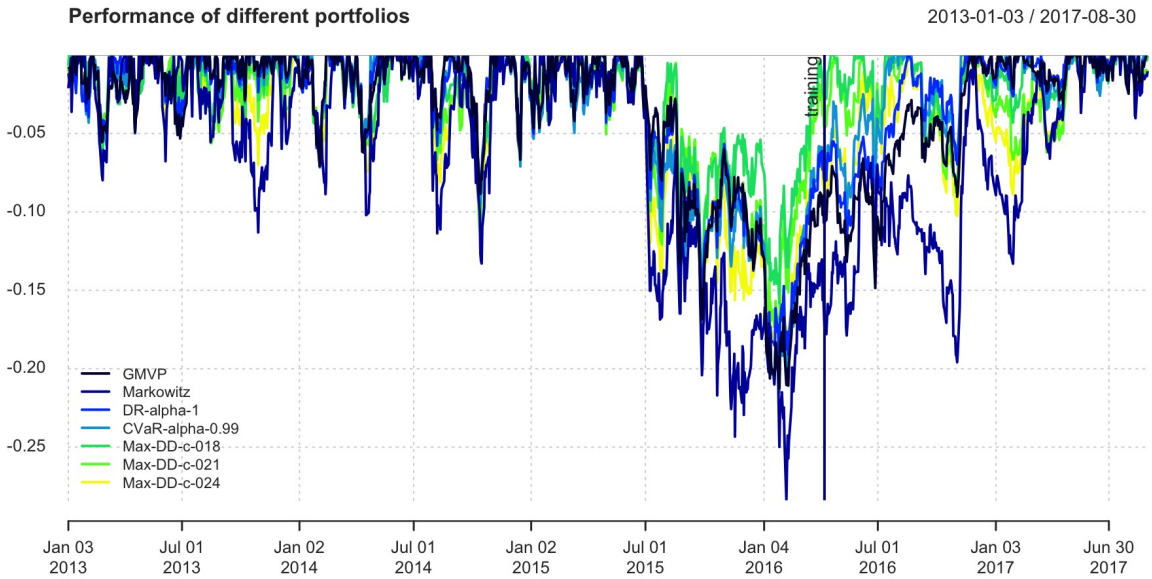
Let us plot the cumulative PnL over time:

```
{ chart.CumReturns(ret_all, main = "Performance of different portfolios",
                   wealth.index = TRUE, legend.loc = "topleft", colorset = rich10equal)
  addEventLines(xts("training", index(X_lin[T_trn])), srt=90, pos=2, lwd = 2, col = "darkblue") }
```



Now the drawdown:

```
{ chart.Drawdown(ret_all, main = "Performance of different portfolios",
                 legend.loc = "bottomleft", colorset = rich10equal)
  addEventLines(xts("training", index(X_lin[T_trn])), srt=90, pos=2, lwd = 2, col = "darkblue") }
```



Indeed, the Max-DD portfolio with $c = 0.18$ has the lowest drawdown (in the in-sample period). In terms of Sharpe ratio, however, it is not that good. We will then keep the Max-DD portfolio with $c = 0.18$:

```
w_all <- w_all[, !colnames(w_all) %in% c("Max-DD-c-021", "Max-DD-c-024")]
```

Ave-DD portfolio

Similary, we can formulate the maximization of the expected return subject to an Ave-DD constraint as the following LP:

$$
\begin{aligned}
\underset{\mathbf{w},\{u_t\}}{\text{maximize}} \quad & \mathbf{w}^T\boldsymbol{\mu} \\
\text{subject to} \quad & \tfrac{1}{T}\sum_{t=1}^{T} u_t \leq \sum_{t=1}^{T} \mathbf{w}^T\mathbf{r}_t^{\text{cum}} + c \\
& \mathbf{w}^T\mathbf{r}_t^{\text{cum}} \leq u_t \\
& u_{t-1} \leq u_t \\
& \mathbf{1}^T\mathbf{w} = 1, \quad \mathbf{w} \geq \mathbf{0}.
\end{aligned}
$$

```
portfolioAveDD <- function(X, c = 0.2) {
  T <- nrow(X)
  N <- ncol(X)
  X <- as.matrix(X)
  X_cum <- apply(X, MARGIN = 2, FUN = cumsum)
  mu <- colMeans(X)
  # variables
  w <- Variable(N)
  u <- Variable(T)
  # problem
  prob <- Problem(Maximize(t(w) %*% mu),
                  constraints = list(w >= 0, sum(w) == 1,
                                     mean(u) <= mean(X_cum %*% w) + c,
                                     u >= X_cum %*% w,
                                     u[-1] >= u[-T]))
  result <- solve(prob)
  return(as.vector(result$getValue(w)))
}

w_AveDD_c004 <- portfolioAveDD(X_log_trn, c = 0.04)
w_AveDD_c006 <- portfolioAveDD(X_log_trn, c = 0.06)
w_AveDD_c008 <- portfolioAveDD(X_log_trn, c = 0.08)
```

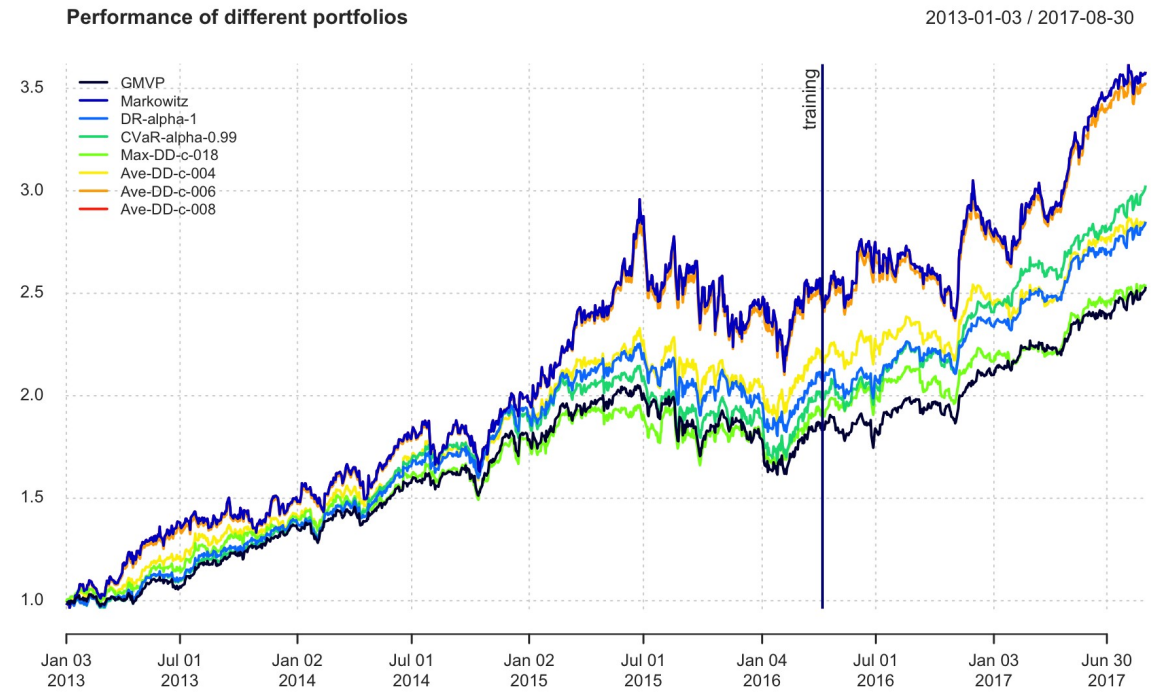
Let compare the performance (in sample vs out-of-sample):

```
# combine portfolios
w_all <- cbind(w_all,
               "Ave-DD-c-004" = w_AveDD_c004,
               "Ave-DD-c-006" = w_AveDD_c006,
               "Ave-DD-c-008" = w_AveDD_c008)

# compute returns of all portfolios
ret_all <- xts(X_lin %*% w_all, index(X_lin))
ret_all_trn <- ret_all[1:T_trn, ]
ret_all_tst <- ret_all[-c(1:T_trn), ]

# performance
t(table.AnnualizedReturns(ret_all_tst)[3, ])
#>                 Annualized Sharpe (Rf=0%)
#> GMVP                               2.1092
#> Markowitz                          1.6875
#> DR-alpha-1                         2.1906
#> CVaR-alpha-0.99                    2.7544
#> Max-DD-c-018                       1.8021
#> Ave-DD-c-004                       1.6757
#> Ave-DD-c-006                       1.7414
#> Ave-DD-c-008                       1.6875
t(maxDrawdown(ret_all_tst))
#>                 Worst Drawdown
#> GMVP                 0.08700167
#> Markowitz            0.13904334
#> DR-alpha-1           0.06968321
#> CVaR-alpha-0.99      0.07322740
#> Max-DD-c-018         0.08486694
#> Ave-DD-c-004         0.08915498
#> Ave-DD-c-006         0.12831001
#> Ave-DD-c-008         0.13904335
t(AverageDrawdown(ret_all_trn))
#>                 Average Drawdown
#> GMVP                  0.01964476
#> Markowitz             0.03199988
#> DR-alpha-1            0.02036526
#> CVaR-alpha-0.99       0.02059035
#> Max-DD-c-018          0.02234339
#> Ave-DD-c-004          0.02057560
#> Ave-DD-c-006          0.02861995
#> Ave-DD-c-008          0.03199988
t(AverageDrawdown(ret_all_tst))
#>                 Average Drawdown
#> GMVP                  0.01459725
#> Markowitz             0.02383452
#> DR-alpha-1            0.01388943
#> CVaR-alpha-0.99       0.01538195
#> Max-DD-c-018          0.01608110
#> Ave-DD-c-004          0.01697335
#> Ave-DD-c-006          0.02221282
#> Ave-DD-c-008          0.02284142
```

The Ave-DD designs have a controlled average drawdown. But in terms of Sharpe ratio they are not especially outstanding.

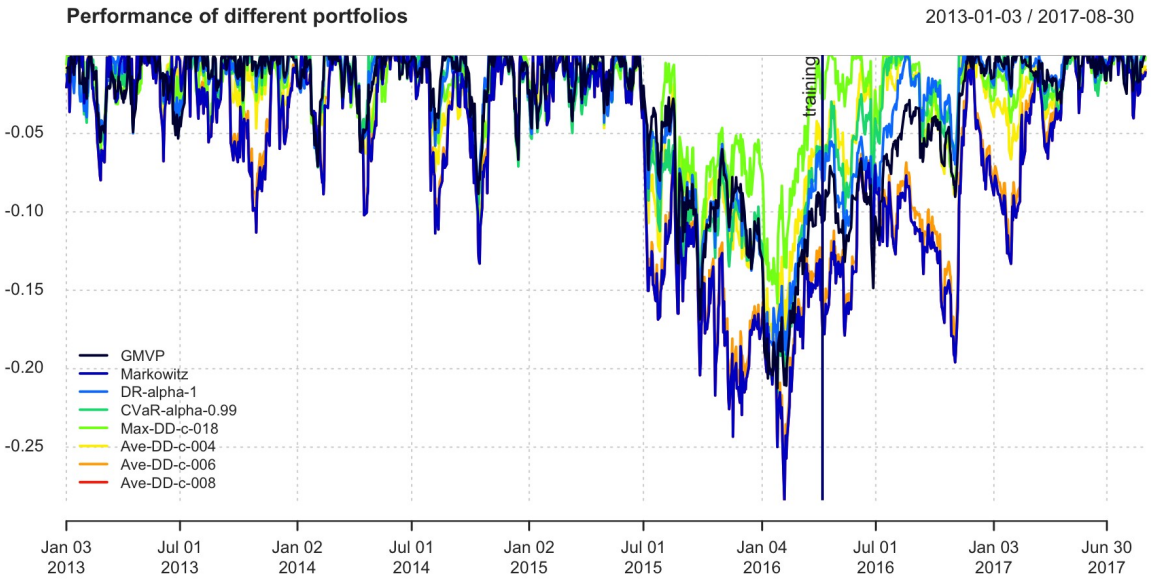
Let us plot the cumulative PnL over time:

```
{ chart.CumReturns(ret_all, main = "Performance of different portfolios",
                   wealth.index = TRUE, legend.loc = "topleft", colorset = rich8equal)
  addEventLines(xts("training", index(X_lin[T_trn])), srt=90, pos=2, lwd = 2, col = "darkblue") }
```



Now the drawdown:

```
{ chart.Drawdown(ret_all, main = "Performance of different portfolios",
                 legend.loc = "bottomleft", colorset = rich8equal)
  addEventLines(xts("training", index(X_lin[T_trn])), srt=90, pos=2, lwd = 2, col = "darkblue") }
```



The Ave-DD portfolio with $c = 0.04$ seems to have the best performance and we keep it for future comparisons:

```
w_all <- w_all[, !colnames(w_all) %in% c("Ave-DD-c-006", "Ave-DD-c-008")]
```

CDaR portfolio

Finally, we can consider the maximization of the mean return subject to a CDaR constraint:

$$\begin{array}{ll}\underset{\mathbf{w},\zeta}{\text{maximize}} & \mathbf{w}^T\boldsymbol{\mu} \\ \text{subject to} & \zeta + \frac{1}{1-\alpha}\frac{1}{T}\sum_{t=1}^{T}\left[\max_{1\le\tau\le t}\mathbf{w}^T\mathbf{r}_\tau^{\text{cum}} - \mathbf{w}^T\mathbf{r}_t^{\text{cum}} - \zeta\right]^+ \le c \\ & \mathbf{1}^T\mathbf{w} = 1, \quad \mathbf{w} \ge \mathbf{0},\end{array}$$

which can be conveniently reformulated as the following LP:

$$\begin{array}{ll}\underset{\mathbf{w},\{z_t\},\zeta,\{u_t\}}{\text{maximize}} & \mathbf{w}^T\boldsymbol{\mu} \\ \text{subject to} & \zeta + \frac{1}{1-\alpha}\frac{1}{T}\sum_{t=1}^{T} z_t \le c \\ & 0 \le z_t \ge u_t - \mathbf{w}^T\mathbf{r}_t^{\text{cum}} - \zeta, \quad t = 1,\ldots,T \\ & \mathbf{w}^T\mathbf{r}_t^{\text{cum}} \le u_t \\ & u_{t-1} \le u_t \\ & \mathbf{1}^T\mathbf{w} = 1, \quad \mathbf{w} \ge \mathbf{0}.\end{array}$$

```
portfolioCDaR <- function(X, c = 0.1, alpha = 0.95) {
  T <- nrow(X)
  N <- ncol(X)
  X <- as.matrix(X)
  X_cum <- apply(X, MARGIN = 2, FUN = cumsum)
  mu <- colMeans(X)
  # variables
  w <- Variable(N)
  z <- Variable(T)
  zeta <- Variable(1)
  u <- Variable(T)
  # problem
  prob <- Problem(Maximize(t(w) %*% mu),
                  constraints = list(w >= 0, sum(w) == 1,
                                     zeta + (1/(T*(1-alpha))) * sum(z) <= c,
                                     z >= 0, z >= u - X_cum %*% w - zeta,
                                     u >= X_cum %*% w,
                                     u[-1] >= u[-T]))
  result <- solve(prob)
  return(as.vector(result$getValue(w)))
}

w_CDaR095_c014 <- portfolioCDaR(X_log_trn, c = 0.14, alpha = 0.95)
w_CDaR099_c016 <- portfolioCDaR(X_log_trn, c = 0.16, alpha = 0.99)
w_CDaR095_c016 <- portfolioCDaR(X_log_trn, c = 0.16, alpha = 0.95)
w_CDaR099_c018 <- portfolioCDaR(X_log_trn, c = 0.18, alpha = 0.99)
```

Let compare the performance (in sample vs out-of-sample):

```r
# combine portfolios
w_all <- cbind(w_all,
               "CDaR095-c-014" = w_CDaR095_c014,
               "CDaR099-c-016" = w_CDaR099_c016,
               "CDaR095-c-016" = w_CDaR095_c016,
               "CDaR099-c-018" = w_CDaR099_c018)

# compute returns of all portfolios
ret_all <- xts(X_lin %*% w_all, index(X_lin))
ret_all_trn <- ret_all[1:T_trn, ]
ret_all_tst <- ret_all[-c(1:T_trn), ]

# performance
t(table.AnnualizedReturns(ret_all_tst)[3, ])
#>                 Annualized Sharpe (Rf=0%)
#> GMVP                               2.1092
#> Markowitz                          1.6875
#> DR-alpha-1                         2.1906
#> CVaR-alpha-0.99                    2.7544
#> Max-DD-c-018                       1.8021
#> Ave-DD-c-004                       1.6757
#> CDaR095-c-014                      1.7179
#> CDaR099-c-016                      1.9595
#> CDaR095-c-016                      1.4371
#> CDaR099-c-018                      1.7642
t(maxDrawdown(ret_all_tst))
#>                 Worst Drawdown
#> GMVP                 0.08700167
#> Markowitz            0.13904334
#> DR-alpha-1           0.06968321
#> CVaR-alpha-0.99      0.07322740
#> Max-DD-c-018         0.08486694
#> Ave-DD-c-004         0.08915498
#> CDaR095-c-014        0.08541869
#> CDaR099-c-016        0.08609170
#> CDaR095-c-016        0.08882975
#> CDaR099-c-018        0.08804789
t(AverageDrawdown(ret_all_tst))
#>                 Average Drawdown
#> GMVP                  0.01459725
#> Markowitz             0.02383452
#> DR-alpha-1            0.01388943
#> CVaR-alpha-0.99       0.01538195
#> Max-DD-c-018          0.01608110
#> Ave-DD-c-004          0.01697335
#> CDaR095-c-014         0.01711395
#> CDaR099-c-016         0.01646761
#> CDaR095-c-016         0.01896366
#> CDaR099-c-018         0.01709859
t(CDD(ret_all_trn))
#>                 Conditional Drawdown 5%
#> GMVP                          0.04947406
#> Markowitz                     0.07620952
#> DR-alpha-1                    0.05004854
#> CVaR-alpha-0.99               0.05132041
#> Max-DD-c-018                  0.05135137
#> Ave-DD-c-004                  0.04667023
#> CDaR095-c-014                 0.04576644
#> CDaR099-c-016                 0.06335329
#> CDaR095-c-016                 0.05229455
#> CDaR099-c-018                 0.04541764
t(CDD(ret_all_tst))
#>                 Conditional Drawdown 5%
#> GMVP                          0.02772373
#> Markowitz                     0.06244467
#> DR-alpha-1                    0.02615751
#> CVaR-alpha-0.99               0.03690619
```

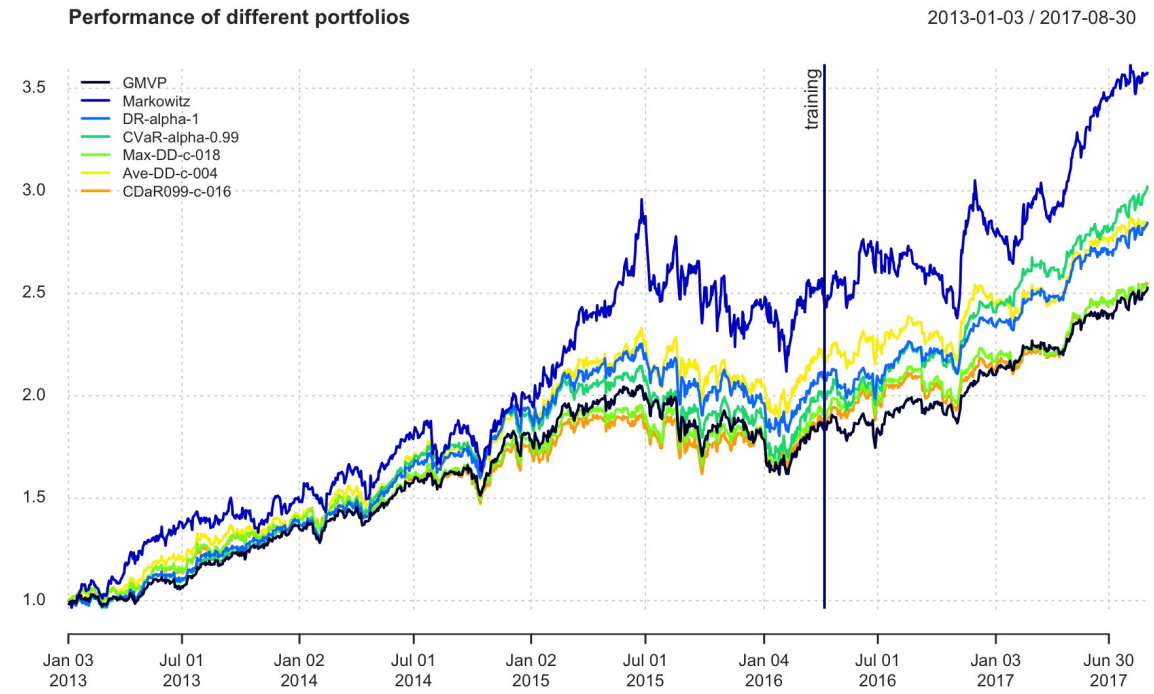
```
#> Max-DD-c-018                0.03554439
#> Ave-DD-c-004                0.06102026
#> CDaR095-c-014               0.04218441
#> CDaR099-c-016               0.02944872
#> CDaR095-c-016               0.06152001
#> CDaR099-c-018               0.04421545
```

The CDaR portfolio with $\alpha = 0.99$ and $c = 0.16$ seems to have the best performance and we keep it for future comparisons:

```
w_all <- w_all[, !colnames(w_all) %in% c("CDaR095-c-014", "CDaR095-c-016", "CDaR099-c-018")]

# recompute returns of all portfolios
ret_all <- xts(X_lin %*% w_all, index(X_lin))
ret_all_trn <- ret_all[1:T_trn, ]
ret_all_tst <- ret_all[-c(1:T_trn), ]

# final performance
t(table.AnnualizedReturns(ret_all_tst)[3, ])
#>                 Annualized Sharpe (Rf=0%)
#> GMVP                               2.1092
#> Markowitz                          1.6875
#> DR-alpha-1                         2.1906
#> CVaR-alpha-0.99                    2.7544
#> Max-DD-c-018                       1.8021
#> Ave-DD-c-004                       1.6757
#> CDaR099-c-016                      1.9595
t(maxDrawdown(ret_all_tst))
#>                 Worst Drawdown
#> GMVP                0.08700167
#> Markowitz           0.13904334
#> DR-alpha-1          0.06968321
#> CVaR-alpha-0.99     0.07322740
#> Max-DD-c-018        0.08486694
#> Ave-DD-c-004        0.08915498
#> CDaR099-c-016       0.08609170
t(AverageDrawdown(ret_all_tst))
#>                 Average Drawdown
#> GMVP                 0.01459725
#> Markowitz            0.02383452
#> DR-alpha-1           0.01388943
#> CVaR-alpha-0.99      0.01538195
#> Max-DD-c-018         0.01608110
#> Ave-DD-c-004         0.01697335
#> CDaR099-c-016        0.01646761
t(CDD(ret_all_tst))
#>                 Conditional Drawdown 5%
#> GMVP                           0.02772373
#> Markowitz                      0.06244467
#> DR-alpha-1                     0.02615751
#> CVaR-alpha-0.99                0.03690619
#> Max-DD-c-018                   0.03554439
#> Ave-DD-c-004                   0.06102026
#> CDaR099-c-016                  0.02944872
```

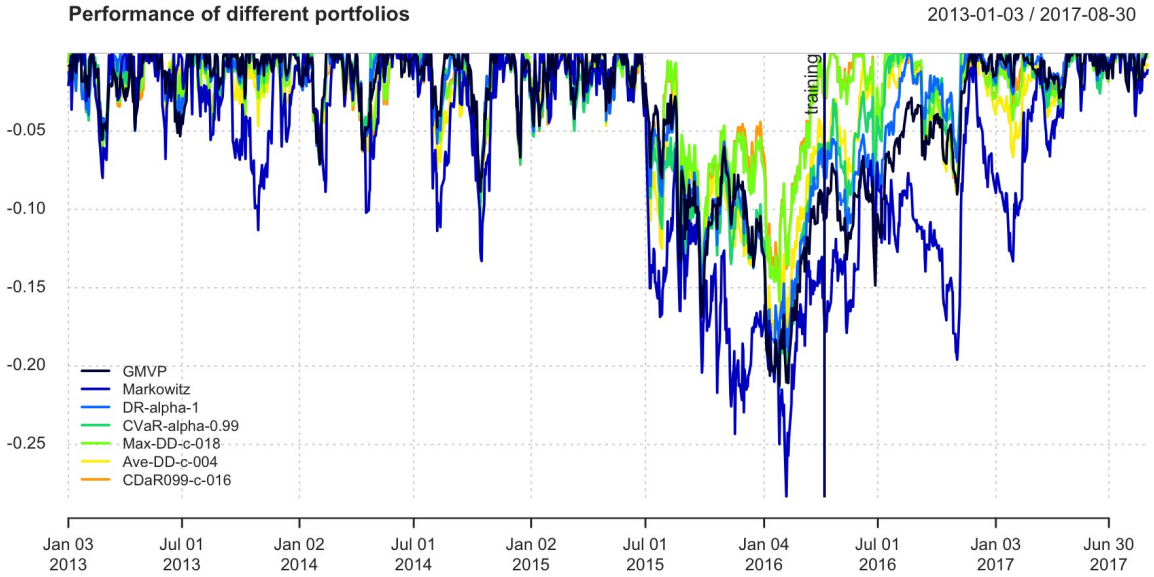
Let us plot the cumulative PnL over time:

```
{ chart.CumReturns(ret_all, main = "Performance of different portfolios",
                   wealth.index = TRUE, legend.loc = "topleft", colorset = rich8equal)
  addEventLines(xts("training", index(X_lin[T_trn])), srt=90, pos=2, lwd = 2, col = "darkblue") }
```



Now the drawdown:

```
{ chart.Drawdown(ret_all, main = "Performance of different portfolios",
                 legend.loc = "bottomleft", colorset = rich8equal)
  addEventLines(xts("training", index(X_lin[T_trn])), srt=90, pos=2, lwd = 2, col = "darkblue") }
```



Conclusion

We can conclude with the following points:

- Markowitz mean-variance porftolio, while it started the field of modern portfolio theory in 1952, has not been embraced by practitioners among other reasons because
  - the variance is not a good measure of risk.
- Alternative measures of risk exist such as the downside risk (e.g., semivariance), VaR, CVaR, and drawdown.
- In particular, we have explored several alternative portfolio formulations that are convex and easily solvable:
  - mean-downside risk portfolio with $\alpha = 1$, which is an LP
  - mean-downside risk portfolio with $\alpha = 2$, which is a QP

    - mean-downside risk portfolio with $\alpha = 3$, which is convex problem
    - mean-CVaR portfolio, which is an LP.
    - mean-DD portfolios (Max-DD, Aver-DD, CDaR), which are all LP.
- The mean-CVaR portfolio (and also the mean-CDaR portfolio) seems to perform well, but more exhaustive backtests should be conducted.
- As a note of caution, one has to be careful with the drawdown portfolios due to the sensitivity to the used data (if not enough data they will not be reliable).

---

1. There are alternative ways to express the CVaR conveniently. For example, for elliptical distributions one can always rewrite the CVaR in terms of the variance.↩