CVXR for PortfolioAnalytics

Xinran Zhao

2022/8/26

Abstract

The purpose of this vignette is to demonstrate examples of optimization problems that can be solved in PortfolioAnalytics with CVXR and its many supported solvers. The problem types covered include not only Linear Programming(LP), Quadratic Programming(QP) but also Second-Order Cone Programming(SOCP). Multiple solvers supported by CVXR can be selected according to optimization types. SCS and ECOS can completely cover the types of problems that ROI can deal with, such as mean-variance and ES problem. In order to better understand the functions of PortfolioAnalytics, users are recommended to read the Vignette *Introduction to PortfolioAnalytics* first.

Contents

1 Getting Started

1.1 Load Packages

Load the necessary packages.

```
library(PortfolioAnalytics)
library(CVXR)
```

1.2 Solvers

There are many solvers supported by the CVXR. Different solvers could support different types of problems. Users could use `optimize_method=c("CVXR", {CVXRsolver})` in the function `optimize.portfolio` to specify the solver. If the argument is `optimize_method="CVXR"`, the default solver for LP and QP will be OSQP, and the default solver for SOCP will be SCS.

Solver	LP	QP	SOCP
GLPK	✓		
OSQP	✓	✓	
GUROBI	✓	✓	✓
SCS	✓	✓	✓
ECOS	✓	✓	✓

For more information about CVXR, please refer to the website https://cvxr.rbind.io/.

1.3 Data

The edhec data set from the PerformanceAnalytics package will be used as example data. It contains monthly returns for 13 assets from 1997-1 to 2019-11.

```
data(edhec)
# Use the first 4 columns in edhec for a returns object
returns <- edhec[, 1:4]
colnames(returns) <- c("CA", "CTAG", "DS", "EM")
print(head(returns, 5))
```

```
##                CA    CTAG      DS      EM
## 1997-01-31 0.0119  0.0393  0.0178  0.0791
## 1997-02-28 0.0123  0.0298  0.0122  0.0525
## 1997-03-31 0.0078 -0.0021 -0.0012 -0.0120
## 1997-04-30 0.0086 -0.0170  0.0030  0.0119
## 1997-05-31 0.0156 -0.0015  0.0233  0.0315
```

```
# Get a character vector of the asset names
funds <- colnames(returns)
```

1.4 Optimization Problems

All optimization problems treated will use linear constraints unless otherwise stated. The types of constraints can be queried from *Introduction to PortfolioAnalytics*. This Vignette will be organized by objective type and provide some visual examples.

2 Maximizing Mean Return

The objective to maximize mean return is a linear problem of the form:

$$\max_{w} \quad \hat{\boldsymbol{\mu}}'\boldsymbol{w}$$

$$s.t. \quad A\boldsymbol{w} \geq b$$

Where $\hat{\boldsymbol{\mu}}$ is the estimated asset returns mean vector and $\boldsymbol{w}$ is the vector of portfolio weights.

2.1 Portfolio Object

The first step in setting up a model is to create the portfolio object. Then add constraints and a return objective.

```
# Create portfolio object
pspec_maxret <- portfolio.spec(assets=funds)
# Add constraints to the portfolio object
pspec_maxret <- add.constraint(pspec_maxret, type="full_investment")
pspec_maxret <- add.constraint(portfolio = pspec_maxret, type = "box",
                               min = c(0.02, 0.05, 0.03, 0.02),
                               max = c(0.55, 0.6, 0.65, 0.5))
# Add objective to the portfolio object
```

```
pspec_maxret <- add.objective(portfolio = pspec_maxret,
                              type = "return", name = "mean")
pspec_maxret
```

```
## **************************************************
## PortfolioAnalytics Portfolio Specification
## **************************************************
##
## Call:
## portfolio.spec(assets = funds)
##
## Number of assets: 4
## Asset Names
## [1] "CA"   "CTAG" "DS"   "EM"
##
## Constraints
## Enabled constraint types
##      - full_investment
##      - box
##
## Objectives:
## Enabled objective names
##      - mean
```

2.2 Optimization

The next step is to run the optimization. Note that `optimize_method=c("CVXR", {CVXRsolver})` should be specified in the function `optimize.portfolio` to use CVXR solvers for the optimization, or use the default solver by giving `optimize_method="CVXR"`.

```
# Run the optimization
opt_maxret <- optimize.portfolio(R=returns, portfolio=pspec_maxret,
                                 optimize_method="CVXR", trace=TRUE)
opt_maxret
```

```
## ***********************************
## PortfolioAnalytics Optimization
## ***********************************
##
## Call:
## optimize.portfolio(R = returns, portfolio = pspec_maxret, optimize_method = "CVXR",
##     trace = TRUE)
##
## Optimal Weights:
##   CA CTAG   DS   EM
## 0.02 0.05 0.65 0.28
##
## Objective Measures:
##     mean
## 0.006371
```

2.3 Backtesting

An out of sample backtest is run with `optimize.portfolio.rebalancing`. In this example, an initial training period of 36 months is used and the portfolio is rebalanced quarterly.

```
bt_maxret <- optimize.portfolio.rebalancing(R=returns, portfolio=pspec_maxret,
                                            optimize_method="CVXR",
                                            rebalance_on="quarters",
                                            training_period=36)
```

The call to `optimize.portfolio.rebalancing` returns the `bt_maxret` object which is a list containing the optimal weights and objective measure at each rebalance period.

```
class(bt_maxret)
```

```
## [1] "optimize.portfolio.rebalancing"
```

```
names(bt_maxret)
```

```
## [1] "portfolio"       "R"               "call"            "elapsed_time"
## [5] "opt_rebalancing"
```

3 Minimizing Variance

The objective to minimize variance is a quadratic problem of the form:

$$\min_{w} \quad \boldsymbol{w}'\Sigma\boldsymbol{w}$$

$$s.t. \quad A\boldsymbol{w} \geq b$$

Where Σ is the estimated covariance matrix of asset returns and w is the set of weights. It is a quadratic problem.

3.1 Global Minimum Variance Portfolio

3.1.1 Portfolio Object

In this example, the only constraint specified is the full investment constraint, therefore the optimization problem is solving for the global minimum variance portfolio.

```
# Create portfolio object
pspec_gmv <- portfolio.spec(assets=funds)
# Add full-investment constraint
pspec_gmv <- add.constraint(pspec_gmv, type="full_investment")
# Add objective of minimizing variance
pspec_gmv <- add.objective(portfolio = pspec_gmv, type = "risk", name = "var")
```

3.1.2 Optimization

```
opt_gmv <- optimize.portfolio(returns, pspec_gmv, optimize_method = "CVXR")
opt_gmv
```

```
## ***********************************
## PortfolioAnalytics Optimization
## ***********************************
##
## Call:
## optimize.portfolio(R = returns, portfolio = pspec_gmv, optimize_method = "CVXR")
##
## Optimal Weights:
##      CA    CTAG      DS      EM
##  0.3637  0.2887  0.5581 -0.2105
##
## Objective Measures:
##  StdDev
## 0.01187
```

3.1.3 Backtesting

```
bt_gmv <- optimize.portfolio.rebalancing(R=returns, portfolio=pspec_gmv,
                                          optimize_method="CVXR",
                                          rebalance_on="quarters",
                                          training_period=36)
```

3.2 Linearly Constrained Global Minimum Variance Portfolio

Various linear inequality constraint, such as box constraints and group box constraints can be used with GMV portfolio construction. Here we demonstrate the case of group box constraints.

```
# portfolio object
pspec_mv <- add.constraint(pspec_gmv, type = "long_only")
pspec_mv <- add.constraint(pspec_mv, type = "group",
                           groups=list(groupA=1,
                                       groupB=c(2, 3),
                                       groupC=4),
                           group_min=c(0, 0.25, 0.01),
                           group_max=c(0.45, 0.6, 0.5))
pspec_mv <- add.constraint(pspec_mv, type = "return", return_target=0.006)
pspec_mv
```

```
## **************************************************
## PortfolioAnalytics Portfolio Specification
## **************************************************
##
## Call:
## portfolio.spec(assets = funds)
##
## Number of assets: 4
## Asset Names
## [1] "CA"   "CTAG" "DS"   "EM"
##
## Constraints
## Enabled constraint types
##      - full_investment
##      - long_only
##      - group
##      - return
##
## Objectives:
## Enabled objective names
##      - var
```

```
# optimization
opt_mv <- optimize.portfolio(returns, pspec_mv, optimize_method = "CVXR")
opt_mv
```

```
## ***********************************
## PortfolioAnalytics Optimization
## ***********************************
##
## Call:
## optimize.portfolio(R = returns, portfolio = pspec_mv, optimize_method = "CVXR")
##
## Optimal Weights:
##     CA   CTAG     DS     EM
## 0.3900 0.0733 0.5267 0.0100
##
## Objective Measures:
##  StdDev
## 0.01452
```

```
# backtesting
bt_mv <- optimize.portfolio.rebalancing(R=returns, portfolio=pspec_mv,
                                        optimize_method="CVXR",
                                        rebalance_on="quarters",
                                        training_period=36)
```

4 Maximizing Quadratic Utility

Next we demonstrate the classical form of Markowitz's mean-variance model to maximize quadratic utility $\mathrm{QU}(\boldsymbol{w}) = \mu_{\mathrm{p}} - \lambda\sigma_{\mathrm{p}}^2 = \boldsymbol{\mu}'\boldsymbol{w} - \lambda\boldsymbol{w}'\Sigma\boldsymbol{w}$:

$$\max_{w} \quad \hat{\boldsymbol{\mu}}'\boldsymbol{w} - \lambda\boldsymbol{w}'\Sigma\boldsymbol{w}$$
$$s.t. \quad A\boldsymbol{w} \geq b$$

Where $\hat{\boldsymbol{\mu}}$ is the estimated mean asset returns, $0 \leq \lambda < \inf$ is the risk aversion parameter, Σ is the estimated covariance matrix of asset returns and $\boldsymbol{w}$ is the set of weights. Quadratic utility maximizes return while penalizing variance. The risk aversion parameter λ controls how much portfolio variance is penalized, and when $\lambda = 0$ it becomes a maximum mean return problem of Section 2, and as $\lambda \to \inf$, it becomes the minimum variance problem of Section 3.

4.1 Portfolio Object

In this case the objectives of the portfolio should be both return and risk, and for this example we will use a risk aversion parameter λ to be 5 by setting `risk_aversion = 5`.

```
pspec_mvo <- portfolio.spec(assets=funds)
pspec_mvo <- add.constraint(pspec_mvo, type="full_investment")
pspec_mvo <- add.constraint(pspec_mvo, type="long_only")
# Add objectives
pspec_mvo <- add.objective(portfolio = pspec_mvo, type = "return", name = "mean")
pspec_mvo <- add.objective(portfolio = pspec_mvo, type = "risk", name = "var",
                           risk_aversion = 5)
```

4.2 Optimization

The optimization result `opt_mvo` shows the call, optimal weights, and the objective measure. Objective measure contains quadratic utility, mean return and standard deviation.

```
opt_mvo <- optimize.portfolio(returns, pspec_mvo, optimize_method = "CVXR")
opt_mvo
```

```
## ***********************************
## PortfolioAnalytics Optimization
## ***********************************
##
## Call:
## optimize.portfolio(R = returns, portfolio = pspec_mvo, optimize_method = "CVXR")
##
## Optimal Weights:
##     CA   CTAG     DS     EM
## 0.0000 0.0559 0.9441 0.0000
##
## Objective Measures:
## optimal value
##     -0.001042
##
##
##     mean
## 0.006484
##
##
##  StdDev
## 0.01597
```

5 Minimizing Expected Shortfall

Expected Shortfall(ES) is also called Conditional Value-at-Risk(CVaR) and Expected Tail Loss(ETL). The ES of a portfolio is

$$ES_\gamma(w) = -E(r_P | r_P \leq q_\gamma(\boldsymbol{w})) = -E(\boldsymbol{w'r} | \boldsymbol{w'r} \leq q_\gamma(\boldsymbol{w}))$$

where q_γ is γ-quantile.

Consider the conditional expectation $E(r|r \leq q_{r,\gamma})$ for a random variable r whose γ-quantile is $q_\gamma = q_{r,\gamma}$. R is defined as a continuous random variable, and r is the realized value of R, f is the probability density function of R. Then

$$E(R|R \leq q_\gamma) = \int_{-\infty}^{\infty} r \cdot f(r|R \leq q_\gamma)dr$$

The conditional distribution function is

$$\begin{aligned} F(r|R \leq q_\gamma) &= P(R \leq r|R \leq q_\gamma) \\ &= \frac{P(R \leq r \cap R \leq q_\gamma)}{P(R \leq q_\gamma)} \\ &= \begin{cases} 1, r > q_\gamma \\ \dfrac{1}{\gamma}F(r), r \leq q_\gamma \end{cases} \end{aligned} \tag{1}$$

Then,

$$E(R|R \leq q_\gamma) = \frac{1}{\gamma}\int_{r \leq q_\gamma} r \cdot f(r)dr$$

Hence,

$$\begin{aligned} E(\boldsymbol{w'r}|\boldsymbol{w'r} \leq q_\gamma) &= \frac{1}{\gamma}\int_{\boldsymbol{w'r} \leq q_\gamma} \boldsymbol{w'r} \cdot f(\boldsymbol{r})d\boldsymbol{r} \\ &= q_\gamma - \frac{1}{\gamma}\int [q_\gamma - \boldsymbol{w'r}]^+ \cdot f(\boldsymbol{r})d\boldsymbol{r} \end{aligned} \tag{2}$$

Replacing q_γ with the free variable t, then the $ES_\gamma(\boldsymbol{w})$ will be:

$$F_\gamma(\boldsymbol{w}, t) = -t + \frac{1}{\gamma}\int [t - \boldsymbol{w'r}]^+ \cdot f(\boldsymbol{r})d\boldsymbol{r}$$

It was shown by Rockafellar and Uryasev (2000) that:

$$\min_{\boldsymbol{w}} ES_\gamma(\boldsymbol{w}) = \min_{\boldsymbol{w},t} F_\gamma(\boldsymbol{w}, t)$$

Because

$$F_\gamma(\boldsymbol{w}, t) \to \hat{F}_\gamma(\boldsymbol{w}, t) = -t + \frac{1}{n \cdot \gamma}\sum_{i=1}^{n}[t - \boldsymbol{w'r_i}]^+$$

the problem minimizing ES is equivalent to solving a linear programming problem

$$\max_{w_i, \{e_i\}, t} \quad t - \frac{1}{n \cdot r}\sum_{i=1}^{n} e_i$$

Therefore, the objective goal is to minimize ES, and it is in the form of:

$$\min_{\eta} \quad \eta + (1-\alpha)^{-1}E(X-\eta)^+$$

Where $0 < \alpha < 1$ is the confidence parameter, and η is the value from which shortfalls are measured in the optimal solution.

5.1 Portfolio Object

The default probability is $\alpha = 95\%$. Specified probability could be given by `arguments`.

```
pspec_es <- portfolio.spec(assets=funds)
pspec_es <- add.constraint(pspec_es, type="full_investment")
# Add objective of minimizing ES
pspec_es <- add.objective(portfolio = pspec_es, type = "risk", name = "ES",
                          arguments = list(p=0.85))
```

5.2 Optimization

```
opt_es <- optimize.portfolio(returns, pspec_es, optimize_method = "CVXR")
opt_es
```

```
## ***********************************
## PortfolioAnalytics Optimization
## ***********************************
##
## Call:
## optimize.portfolio(R = returns, portfolio = pspec_es, optimize_method = "CVXR")
##
## Optimal Weights:
##      CA    CTAG      DS      EM
##  0.4873  0.2215  0.4853 -0.1942
##
## Objective Measures:
##      ES
## 0.01407
```

6 Minimizing Expected Quadratic Shortfall

Expected Quadratic Shortfall(EQS) is also called Second-Moment Coherent Risk Measure(SMCR). The objective to minimize EQS is in the form of:

$$\underset{\eta}{minimize} \quad \eta + (1-\alpha)^{-1}||(X-\eta)^{+}||_2$$

Where α is the confidence parameter and $0 < \alpha < 1$, η is the value from which quadratic shortfalls are measured in the optimal solution. The default probability is $\alpha = 95\%$. Minimizing EQS could be incorporated into a convex problem as a second-order cone constraints. For Second-Order Cone Optimization(SOCopt), PortfolioAnalytics uses the CVXR package with SCS or ECOS solvers.

6.1 Portfolio Object

The default probability is $\alpha = 95\%$. Specified probability could be given by `arguments`.

```
pspec_eqs <- portfolio.spec(assets=funds)
pspec_eqs <- add.constraint(pspec_eqs, type="full_investment")
# Add objective of minimizing EQS
pspec_eqs <- add.objective(portfolio = pspec_eqs, type = "risk", name = "EQS",
                           arguments = list(p=0.85))
```

6.2 Optimization

```r
opt_eqs <- optimize.portfolio(returns, pspec_eqs, optimize_method = "CVXR")
opt_eqs
```

```
## ***********************************
## PortfolioAnalytics Optimization
## ***********************************
##
## Call:
## optimize.portfolio(R = returns, portfolio = pspec_eqs, optimize_method = "CVXR")
##
## Optimal Weights:
##      CA    CTAG      DS      EM
##  0.0125  0.4055  0.7451 -0.1631
##
## Objective Measures:
##     EQS
## 0.02353
```

6.3 Backtesting

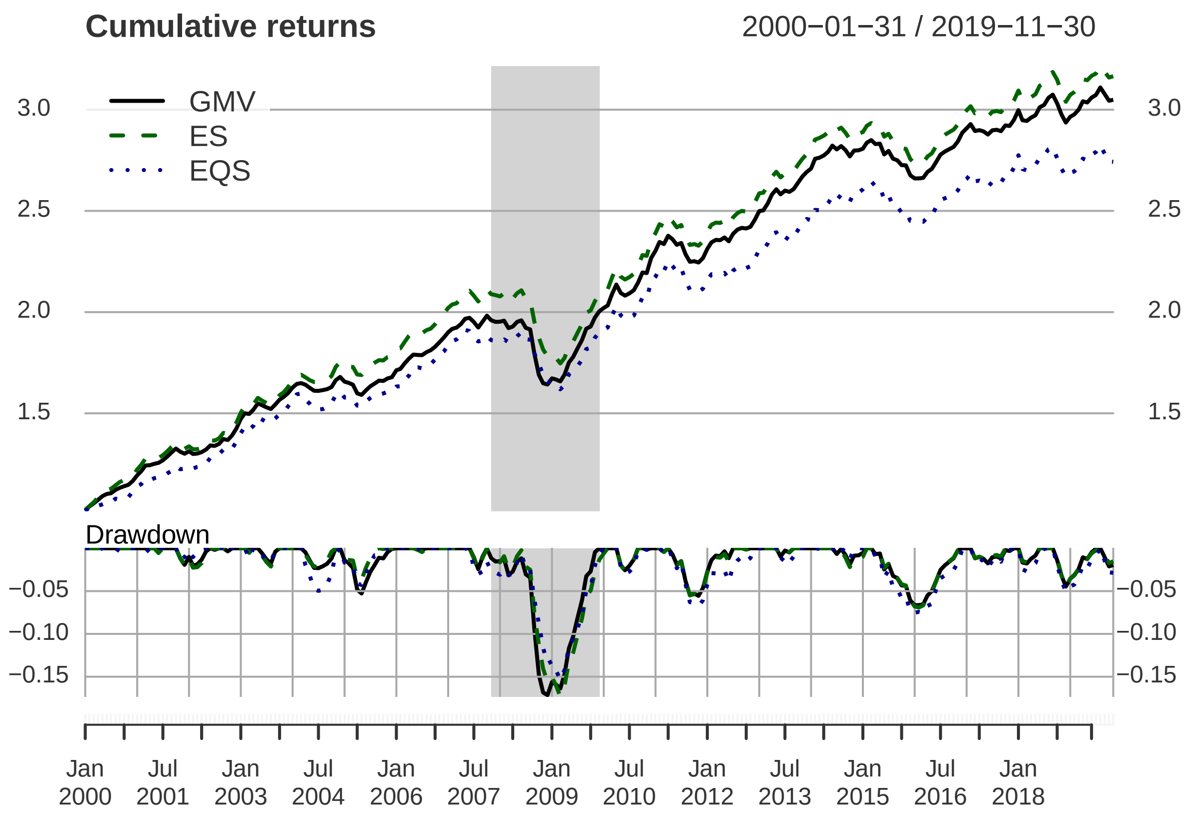
In this example, a comparative backtesting among GMV, ES and EQS portfolio is generated.

```r
# Optimize Portfolio at Quarterly Rebalancing and 36-Month Training
bt_es <- optimize.portfolio.rebalancing(R=returns, portfolio=pspec_es,
                                        optimize_method="CVXR",
                                        rebalance_on="quarters",
                                        training_period=36)
bt_eqs <- optimize.portfolio.rebalancing(R=returns, portfolio=pspec_eqs,
                                         optimize_method="CVXR",
                                         rebalance_on="quarters",
                                         training_period=36)
```

The performance of backtesting could be shown as a plot of cumulative returns and a plot of drawdown.

```r
# Extract time series of portfolio weights
wts_gmv <- extractWeights(bt_gmv)
wts_gmv <- wts_gmv[complete.cases(wts_gmv),]

wts_es <- extractWeights(bt_es)
wts_es <- wts_es[complete.cases(wts_es),]

wts_eqs <- extractWeights(bt_eqs)
wts_eqs <- wts_eqs[complete.cases(wts_eqs),]

# Compute cumulative returns of portfolios
port_gmv <- Return.rebalancing(returns, wts_gmv)
port_es <- Return.rebalancing(returns, wts_es)
port_eqs <- Return.rebalancing(returns, wts_eqs)
```

```r
# Combine cumulative returns
ret.comb <- na.omit(merge(port_gmv, port_es, port_eqs, all=F))
names(ret.comb) = c("GMV", "ES", "EQS")

# Compute cumulative geometric portfolios returns
R <- ret.comb
geometric = TRUE
c.xts <- if ( geometric ) {
  cumprod(1+R)
} else {
  1 + cumsum(R)
}

# Cumulative returns panel (Peter Carl)
p <- xts::plot.xts(c.xts[,1], col="black", main = "Cumulative returns",
                   grid.ticks.lwd=1, grid.ticks.lty = "solid", grid.ticks.on = "years",
                   labels.col="grey20", cex.axis=0.8, format.labels = "%b\n%Y",
                   ylim = c(min(c.xts), max(c.xts)))
p <- xts::addSeries(c.xts[,2], on=1, lwd=2, col="dark green", lty="dashed")
p <- xts::addSeries(c.xts[,3], on=1, lwd=2, col="dark blue", lty="dotted")
p <- xts::addLegend("topleft", on = 1,
                    legend.names = names(c.xts),
                    lty = c(1, 2, 3), lwd = rep(2, NCOL(c.xts)),
                    col = c("black", "dark green", "dark blue"),
                    bty = "o", box.col = "white",
                    bg=rgb(t(col2rgb("white")), alpha = 200,
                           maxColorValue = 255) )

## Drawdowns panel(Peter Carl)
d.xts <- PerformanceAnalytics::Drawdowns(R)
p <- xts::addSeries(d.xts[,1], col="black", lwd=2, main="Drawdown",
                    ylim = c(min(d.xts), 0), lty="solid")
p <- xts::addSeries(d.xts[,2], on=2, lwd=2, col="dark green", lty="dashed")
p <- xts::addSeries(d.xts[,3], on=2, lwd=2, col="dark blue", lty="dotted")

## panel 1 and 2 ylim
ylim1 <- c(p$Env$ylim[[2]][1], p$Env$ylim[[2]][2])
ylim2 <- c(p$Env$ylim[[4]][1], p$Env$ylim[[4]][2])
ylim <- c(ylim1, ylim2)
# get longest drawdown dates for xts object
dt <- table.Drawdowns(R, top = 1) # just want to find the worst drawdown
dt2 <- t(dt[,c("From", "To")])
x <- as.vector(dt2[,NCOL(dt2)])
y <- as.xts(matrix(rep(ylim, length(x)),ncol=length(ylim), byrow=TRUE), order.by=as.Date(x))
i=1
p <- xts::addPolygon(y[i:(i+1),1:2], on=-1, col="lightgrey") # top panel
p <- xts::addPolygon(y[i:(i+1),3:4], on=-2, col="lightgrey") # lower panel

p
```



7 Maximizing Mean Return Per Unit Risk

There are three kinds of risk measurements: variance or standard deviation, ES and EQS. The problem maximizing mean return per unit risk could be transferred into minimizing risk with a target return constraint. For these types of problems, both return and risk objectives should be given. Then an argument should be given to the `optimize.portfolio` to specify the type of problem.

7.1 Maximum Sharpe Ratio Portfolios

The Sharpe Ratio of a random return r_P of a portfolio P is defined as:

$$\frac{E(r_P) - r_f}{\sqrt{Var(r_P)}}$$

The problem of maximizing the Sharpe Ratio can be formulated as a quadratic problem with a budget normalization constraint. It is shown in Cornuejols, Pena and Tutuncu, that the problem is:

$$\begin{aligned} \underset{w}{minimize} \quad & w'\Sigma w \\ s.t. \quad (\hat{\mu} - r_f\mathbf{1})^T w &= 1 \\ \mathbf{1}^T w &= \kappa \\ \kappa &> 0 \end{aligned} \tag{3}$$

with the maximized Sharpe ratio given by $\tilde{w}^* = w^*/\kappa^*$.

When creating the portfolio, both return and risk objectives should be given. Then the argument `maxSR = TRUE` should be specified in the function `optimize.portfolio` to distinguish from the mean-variance optimization. The default argument is `maxSR = FALSE` and the default action for dealing with both mean and var/StdDev objectives is to maximize quadratic utility.

```
# Create portfolio object
pspec_sr <- portfolio.spec(assets=funds)
## Add constraints of maximizing Sharpe Ratio
pspec_sr <- add.constraint(pspec_sr, type="full_investment")
pspec_sr <- add.constraint(pspec_sr, type="long_only")
## Add objectives of maximizing Sharpe Ratio
pspec_sr <- add.objective(pspec_sr, type = "return", name = "mean")
pspec_sr <- add.objective(pspec_sr, type="risk", name="var")

# Optimization
optimize.portfolio(returns, pspec_sr, optimize_method = "CVXR", maxSR=TRUE)
```

```
## ***********************************
## PortfolioAnalytics Optimization
## ***********************************
##
## Call:
## optimize.portfolio(R = returns, portfolio = pspec_sr, optimize_method = "CVXR",
##     maxSR = TRUE)
##
## Optimal Weights:
##     CA   CTAG     DS     EM
## 0.2036 0.2422 0.5542 0.0000
##
## Objective Measures:
##     mean
## 0.005797
##
##
##  StdDev
## 0.01313
##
##
## Sharpe Ratio
##       0.4414
```

7.2 Maximum ES ratio Portfolios

The ES ratio(ESratio), which is also called STARR in PortfolioAnalytics, is defined as:

$$\frac{E(r_P) - r_f}{ES_\alpha(r_P)}$$

Similar to maximizing Sharpe Ratio, the problem maximizing the ES ratio can be formulated as a minimizing ES problem with a budget normalization constraint.

When creating the portfolio, both return and ES objectives should be given. When solving the problem, the default argument `ESratio=TRUE` in the function `optimize.portfolio` specify the problem type. Besides, this argument is equivalent to `maxSTARR=TRUE`, which is used in other Vignettes. If one of these two arguments specified as FALSE, the action will be to minimize ES ignoring the return objective.

```r
# Create portfolio object
pspec_ESratio <- portfolio.spec(assets=funds)
## Add constraints of maximizing return per unit ES
pspec_ESratio <- add.constraint(pspec_ESratio, type="full_investment")
pspec_ESratio <- add.constraint(pspec_ESratio, type="long_only")
## Add objectives of maximizing return per unit ES
pspec_ESratio <- add.objective(pspec_ESratio, type = "return", name = "mean")
pspec_ESratio <- add.objective(pspec_ESratio, type="risk", name="ES")

# Optimization
optimize.portfolio(returns, pspec_ESratio, optimize_method = "CVXR", ESratio=TRUE)
```

```
## ***********************************
## PortfolioAnalytics Optimization
## ***********************************
##
## Call:
## optimize.portfolio(R = returns, portfolio = pspec_ESratio, optimize_method = "CVXR",
##     ESratio = TRUE)
##
## Optimal Weights:
##     CA   CTAG     DS     EM
## 0.0000 0.4288 0.5712 0.0000
##
## Objective Measures:
##     mean
## 0.005565
##
##
##      ES
## 0.02317
##
##
## ES ratio
##   0.2402
```

7.3 Maximum EQS ratio Portfolios

The EQS ratio of a random return r_P of a portfolio P is defined as:

$$\frac{E(r_P) - r_f}{EQS_\alpha(r_P)}$$

Similar to maximizing Sharpe Ratio, the problem maximizing EQS ratio could be formulated as a minimizing EQS problem with a budget normalization constraint.

When creating the portfolio, both return and EQS objectives should be given. The argument `EQSratio=` is used to specify the problem type and the default value is `EQSratio=TRUE`. If `EQSratio=FALSE`, the action will be to minimize EQS ignoring the return objective. The default $\alpha = 0.95$, and it can be specified by `arguments`.

```r
# Create portfolio object
pspec_EQSratio <- portfolio.spec(assets=funds)
```

```
## Add constraints of maximizing return per unit EQS
pspec_EQSratio <- add.constraint(pspec_EQSratio, type="full_investment")
pspec_EQSratio <- add.constraint(pspec_EQSratio, type="long_only")
## Add objectives of maximizing return per unit EQS
pspec_EQSratio <- add.objective(pspec_EQSratio, type = "return", name = "mean")
pspec_EQSratio <- add.objective(pspec_EQSratio, type="risk", name="EQS",
                                arguments = list(p=0.95))

# Optimization
optimize.portfolio(returns, pspec_EQSratio, optimize_method = "CVXR", EQSratio=TRUE)
```

```
## ***********************************
## PortfolioAnalytics Optimization
## ***********************************
##
## Call:
## optimize.portfolio(R = returns, portfolio = pspec_EQSratio, optimize_method = "CVXR",
##     EQSratio = TRUE)
##
## Optimal Weights:
##     CA   CTAG     DS     EM
## 0.0000 0.4518 0.5482 0.0000
##
## Objective Measures:
##     mean
## 0.005509
##
##
##     EQS
## 0.02804
##
##
## EQS ratio
##    0.1965
```

8 Efficient Frontier

Generate efficient frontiers with MVO, mean-ES and mean-EQS portfolios.

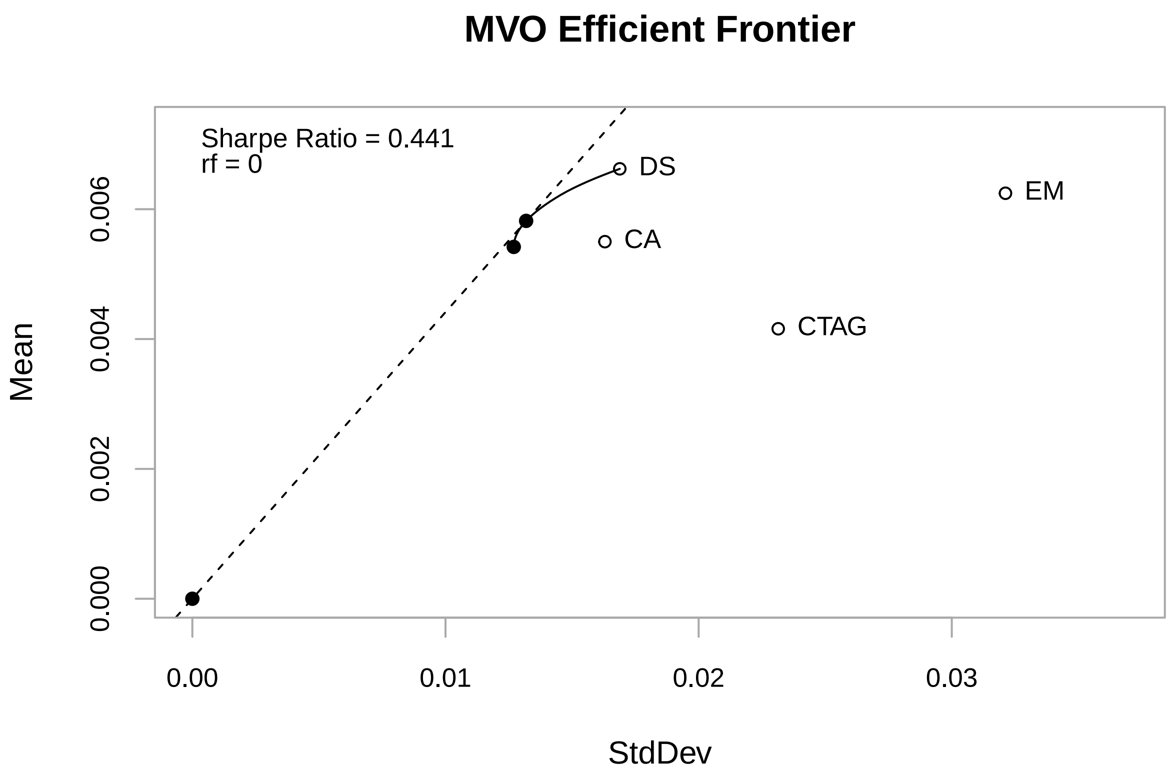
```
init <- portfolio.spec(assets=funds)
init <- add.constraint(portfolio=init, type="full_investment")
init <- add.constraint(portfolio=init, type="long_only")

# MVO
meanvar.ef <- create.EfficientFrontier(R=returns, portfolio=init, type="mean-StdDev")
```

```
## Registered S3 method overwritten by 'ROI':
##   method           from
##   print.constraint PortfolioAnalytics
```

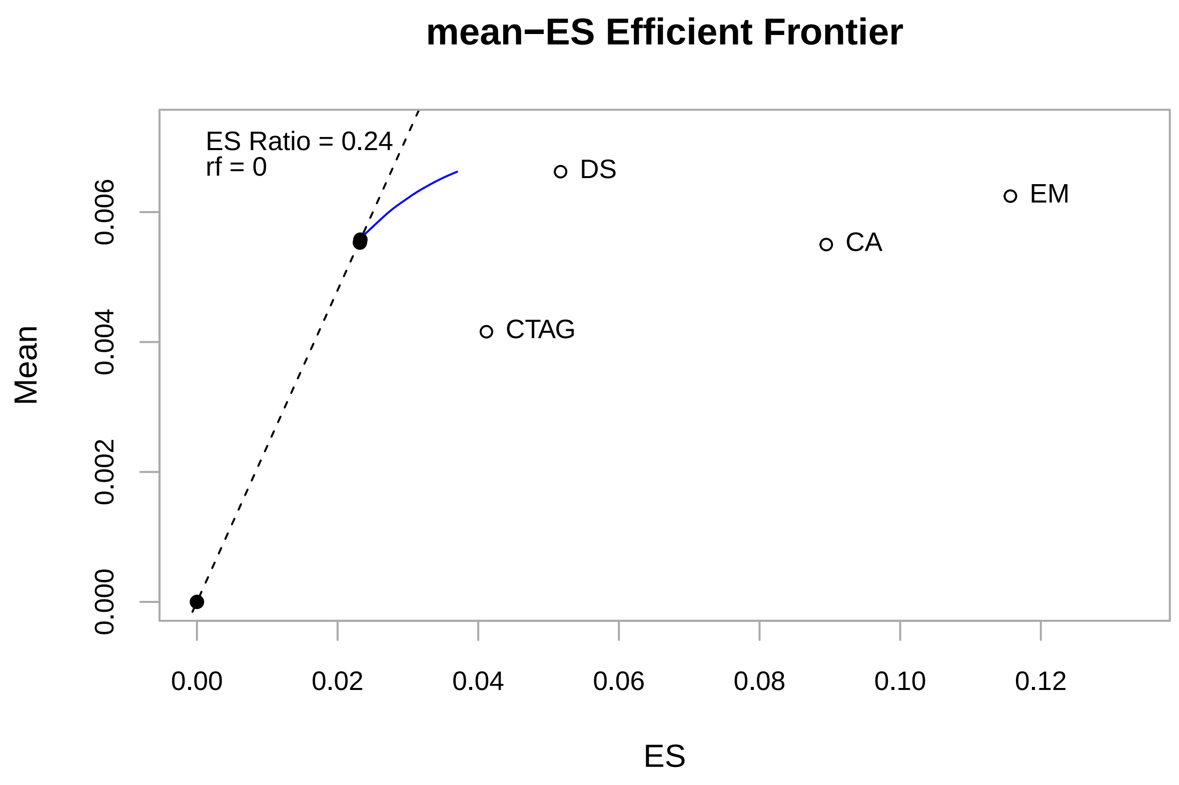
```
meanvar.ef
```

```
## **************************************************
## PortfolioAnalytics Efficient Frontier
## **************************************************
##
## Call:
## create.EfficientFrontier(R = returns, portfolio = init, type = "mean-StdDev")
##
## Efficient Frontier Points: 25
##
## **************************************************
## PortfolioAnalytics Portfolio Specification
## **************************************************
##
## Call:
## portfolio.spec(assets = funds)
##
## Number of assets: 4
## Asset Names
## [1] "CA"   "CTAG" "DS"   "EM"
##
## Constraints
## Enabled constraint types
##      - full_investment
##      - long_only
```

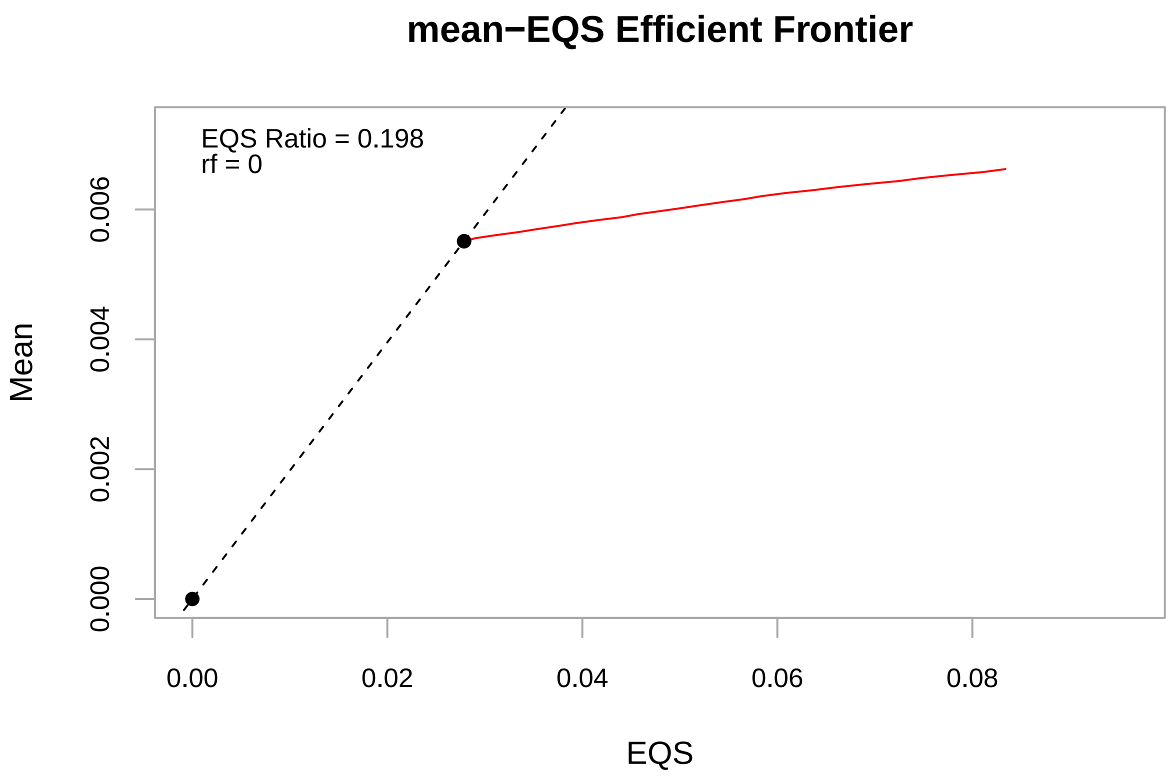
```
chart.EfficientFrontier(meanvar.ef, match.col="StdDev", type="l",
                        main="MVO Efficient Frontier",
                        RAR.text="Sharpe Ratio", pch=4)
```


MVO Efficient Frontier
Sharpe Ratio = 0.441
rf = 0
DS
EM
CA
CTAG
Mean
0.000
0.002
0.004
0.006
0.00
0.01
0.02
0.03
StdDev

```r
# mean-ES
meanetl.ef <- create.EfficientFrontier(R=returns, portfolio=init, type="mean-ES")
chart.EfficientFrontier(meanetl.ef, match.col="ES",
                        main="mean-ES Efficient Frontier",
                        type="l", col="blue", RAR.text="ES Ratio")
```



```
# mean-EQS
meaneqs.ef <- create.EfficientFrontier(R=returns, portfolio=init, type="mean-EQS")
chart.EfficientFrontier(meaneqs.ef, match.col="EQS", chart.assets = FALSE,
                        main="mean-EQS Efficient Frontier",
                        type="l", col="red", RAR.text="EQS Ratio")
```


mean−EQS Efficient Frontier
EQS Ratio = 0.198
rf = 0
Mean
EQS