Using CMEtest with PortfolioAnalytics

Mohamed Ishmael Diwan Belghazi

March 26, 2014

Abstract

The purpose of this vignette is to show how to pass custom location and scatter estimators to PortfolioAnalytics using CMEtest.

Contents

1 Preliminaries

1.1 Loading packages

Loading Packages and sourcefiles. CMEtest is not yet under package format as some thinking is still required concerning its internal architecture.

```r
# Loading packages
suppressMessages(require(PortfolioAnalytics))
# Loading optimization packages
suppressMessages(require(ROI))
suppressMessages(require(ROI.plugin.glpk))
suppressMessages(require(ROI.plugin.quadprog))

################### Loading test ##

suppressMessages(library(CMEtest))
options(width = 60)
```

For now, the package tawny is loaded only for the dataset.

1.2 Loading Data

We take 50 observations for 10 assets.

```r
################### Loading data ##

# Loading data
data(sp500.subset)
returns <- sp500.subset[1:50, 1:10]
assets <- colnames(returns)
```

2 Defining portfolio

We define a Global minimum variance long only portfolio with Box Constraint. Since we are optimizing a var objective, we can use the ROI solver.

```r
## Specifying portfolio
port_gmv <- portfolio.spec(assets = assets)
## specifying long only constraing
long_const = 0
## specifying unform upper box constraints
upper_box <- 0.6

## Setting constraints
```

```r
port_gmv <- add.constraint(portfolio = port_gmv, type = "full_investment",
    enabled = TRUE)
port_gmv <- add.constraint(portfolio = port_gmv, type = "box",
    min = long_const, max = upper_box)
## Adding objective function
port_gmv <- add.objective(portfolio = port_gmv, type = "risk",
    name = "var")

## Showing portfolio specification
print(port_gmv)

## ***************************************************
## PortfolioAnalytics Portfolio Specification
## ***************************************************
##
## Call:
## portfolio.spec(assets = assets)
##
## Assets
## Number of assets: 10
##
## Asset Names
##  [1] "MMM"  "ABT"  "ANF"  "ADBE" "AMD"  "A"    "APD"  "AKS"
##  [9] "AA"   "AYE"
##
## Constraints
## Number of constraints: 2
## Number of enabled constraints: 2
## Enabled constraint types
##  - full_investment
##  - box
## Number of disabled constraints: 0
##
## Objectives
## Number of objectives: 1
## Number of enabled objectives: 1
## Enabled objective names
##  - var
## Number of disabled objectives: 0
```

We show the portfolio specifications

```
## Showing portfolio specification
print(port_gmv)

## ***************************************************
## PortfolioAnalytics Portfolio Specification
## ***************************************************
##
## Call:
## portfolio.spec(assets = assets)
##
## Assets
## Number of assets: 10
##
## Asset Names
##  [1] "MMM"  "ABT"  "ANF"  "ADBE" "AMD"  "A"    "APD"  "AKS"
##  [9] "AA"   "AYE"
##
## Constraints
## Number of constraints: 2
## Number of enabled constraints: 2
## Enabled constraint types
##  - full_investment
##  - box
## Number of disabled constraints: 0
##
## Objectives
## Number of objectives: 1
## Number of enabled objectives: 1
## Enabled objective names
##  - var
## Number of disabled objectives: 0
```

3 Using CMEtest

3.1 Construting specification

For the time being only robust estimation is implemented. Smoothing, shrinking and filtering will be implemented when the architecture will have matured.

We specify an mle (classical covariance) specification and call the summary function.

```
## Specifying sample cov

mleCovSpec <- CMEspec(smooth = 'None',
                      estim = 'mle',
                      shrink = 'None',
                      filter = 'None')
class(mleCovSpec)

## [1] "CMEspec"

## Showing summary
summary(mleCovSpec)

## /---------------------------------\
## |Matrix Covariance Estimators test |
## \---------------------------------/
##
##
##   Specification summary
##
##  Smoothing: None
##  Estimation: mle
##  Shrinking: None
##  Filtering: None
##
## -----------------------------------
```

We also specify an minimum covariance determinant robust estimator (mcd). The Minimum Covariance Determinant estimator is a robust estimator of a data sets covariance introduced by Rousseeuw(1984). The idea is to find a given proportion (h) of good observations which are not outliers and compute their empirical covariance matrix. This empirical covariance matrix is then rescaled to compensate the performed selection of observations (consistency step).

```
robCovSpec <- CMEspec(smooth = "None", estim = "mcd", shrink = "None",
    filter = "None")

class(robCovSpec)

## [1] "CMEspec"

## Showing summary
summary(robCovSpec)

## /----------------------------------\
## |Matrix Covariance Estimators test |
## \----------------------------------/
##
##
##   Specification summary
##
##  Smoothing: None
##  Estimation: mcd
##  Shrinking: None
##  Filtering: None
##
## ------------------------------------
```

3.2 Creating moment functions for PortfolioAnalytics

Creating moment functions for Portfolio Analytics is straightforward using CMEtest. It is enough to call the MakeMomentFUN function on the specification object.

```
## Generating Moment functions. These functions will
## dynamically compute location and scatter when passed to
## optimize.portfolio.

MleMomentFUN <- MakeMomentFUN(mleCovSpec)
RobMomentFUN <- MakeMomentFUN(robCovSpec)
```

Note that for RobMomenFun, only the location and scatter are robust. The third and fourth moment are not computed using robust method. It does not matter if the optimization method and/or objective use at most the first two moments. An example

where it matters, is when using the Edgeworth or Cornish-Fisher approximation of a non-normal distribution. The latter approximations being based on the four first cumulant.

4 Optimizing the portfolios!

4.1 Dynamic moment functions and optimization

We are now ready to passe the generated moment function. let us optimize the portfolios!

```
################################### Let's Optimize the potfolios! ##

# mle version
opt_gmv_mle <- optimize.portfolio(R = returns, portfolio = port_gmv,
    optimize_method = "ROI", momentFUN = "MleMomentFUN", trace = TRUE)

## Robust version
opt_gmv_rob <- optimize.portfolio(R = returns, portfolio = port_gmv,
    optimize_method = "ROI", momentFUN = "RobMomentFUN", trace = TRUE)
```

Let us extract the weights For the mle estimator:

```
extractWeights(opt_gmv_mle)

##        MMM        ABT        ANF       ADBE        AMD
##  3.333e-01  3.309e-01  0.000e+00 -5.761e-18  8.891e-20
##          A        APD        AKS         AA        AYE
##  7.747e-02  1.202e-17  6.127e-02  1.349e-17  1.970e-01
```

For the mcd estimator:

```
extractWeights(opt_gmv_rob)

##        MMM        ABT        ANF       ADBE        AMD
##  3.564e-01  3.799e-01  1.912e-17 -1.877e-17  0.000e+00
##          A        APD        AKS         AA        AYE
##  1.348e-01  1.761e-02  1.113e-01  2.445e-17 -5.923e-17
```

4.2 Computed moment functions and optimization

Robust moment computing can be expensive. Thankfully, It is also possible to generate pre-computed moment functions. The process is very straightforward, instead of calling the moment making function on a CMEtest specification object it is enough to call it on a CMEtest estimation object.

Let us first explicitly compute the covariance. In order to do so, one has to use the Estimate() generic function on the specification object and on the choosen dataset.

```
## Let us compute the covariances
mleCovEst <- Estimate(mleCovSpec, returns)
class(mleCovEst)

## [1] "CMEest"

robCovEst <- Estimate(robCovSpec, returns)
class(robCovEst)

## [1] "CMEest"
```

Althought it is not necessary, let's get the empirical and robust correlations just for fun.

MLE estimator correlation:

```
## Using correlation getter on the estimation object.
GetCorr(mleCovEst)

##            MMM      ABT     ANF     ADBE    AMD       A
## MMM  1.000000  0.38502 0.60797  0.47238 0.4823 0.48520
## ABT  0.385018  1.00000 0.14786  0.39328 0.1054 0.23548
## ANF  0.607971  0.14786 1.00000  0.46896 0.6639 0.35549
## ADBE 0.472381  0.39328 0.46896  1.00000 0.4371 0.51525
## AMD  0.482334  0.10536 0.66387  0.43709 1.0000 0.48161
## A    0.485197  0.23548 0.35549  0.51525 0.4816 1.00000
## APD  0.608784  0.31163 0.44302  0.39229 0.5097 0.32069
## AKS  0.006833 -0.20216 0.02507 -0.01556 0.0716 0.09742
## AA   0.405074  0.00953 0.19542  0.35568 0.2540 0.31496
## AYE  0.386649  0.27902 0.14487  0.32792 0.1103 0.27224
##          APD        AKS      AA    AYE
## MMM  0.6088  0.006833 0.40507 0.3866
## ABT  0.3116 -0.202156 0.00953 0.2790
```

```
## ANF  0.4430  0.025067 0.19542 0.1449
## ADBE 0.3923 -0.015555 0.35568 0.3279
## AMD  0.5097  0.071597 0.25399 0.1103
## A    0.3207  0.097422 0.31496 0.2722
## APD  1.0000  0.390666 0.54052 0.4898
## AKS  0.3907  1.000000 0.48921 0.2614
## AA   0.5405  0.489209 1.00000 0.4067
## AYE  0.4898  0.261419 0.40675 1.0000
```

Robust Mcd estimator correlation:

```
GetCorr(robCovEst)

##           MMM       ABT      ANF      ADBE      AMD          A
## MMM   1.00000  0.402067 0.63051  0.60784  0.53246  0.851319
## ABT   0.40207  1.000000 0.47076  0.73479  0.29216  0.403987
## ANF   0.63051  0.470757 1.00000  0.83040  0.71515  0.688068
## ADBE  0.60784  0.734789 0.83040  1.00000  0.61130  0.699586
## AMD   0.53246  0.292158 0.71515  0.61130  1.00000  0.644952
## A     0.85132  0.403987 0.68807  0.69959  0.64495  1.000000
## APD   0.60226  0.359932 0.67166  0.70283  0.58239  0.615828
## AKS  -0.02091 -0.319985 0.05689 -0.07264 -0.04467 -0.003436
## AA    0.41691 -0.005988 0.54314  0.38914  0.53884  0.487581
## AYE   0.45098  0.362403 0.53969  0.56107  0.47591  0.571584
##          APD       AKS         AA    AYE
## MMM  0.6023 -0.020914  0.416906 0.4510
## ABT  0.3599 -0.319985 -0.005988 0.3624
## ANF  0.6717  0.056887  0.543136 0.5397
## ADBE 0.7028 -0.072641  0.389140 0.5611
## AMD  0.5824 -0.044665  0.538835 0.4759
## A    0.6158 -0.003436  0.487581 0.5716
## APD  1.0000  0.439340  0.514510 0.8142
## AKS  0.4393  1.000000  0.482009 0.4572
## AA   0.5145  0.482009  1.000000 0.3892
## AYE  0.8142  0.457196  0.389208 1.0000
```

Similarly, we can get the first moment. For the classical estimation:

```
GetLoc(mleCovEst)
```

```
##         MMM         ABT         ANF        ADBE         AMD
## -0.0015708  0.0010227 -0.0040658  0.0003023 -0.0090793
##           A         APD         AKS          AA         AYE
##  0.0022011 -0.0011369 -0.0077219 -0.0054409 -0.0024925
```

For the robust estimation:

```
GetLoc(robCovEst)

##         MMM         ABT         ANF        ADBE         AMD
## -0.0048518  0.0001974 -0.0093319 -0.0003093 -0.0089676
##           A         APD         AKS          AA         AYE
## -0.0004503 -0.0029707 -0.0030716 -0.0095160 -0.0033690
```

4.3 Generating precomputed moment functions

Now that we have the estimation object, generating the precomputed moment functions is as easy as before. One has just to call the MakeMomentFUN function on the CMEtest estimation object.

```
## Now we create Precomputed moment functions
MlePrecompMomentFUN <- MakeMomentFUN(mleCovEst)
RobPrecompMomentFUN <- MakeMomentFUN(robCovEst)
```

4.4 Optimizing portfolio with precomputed moment functions

```
# mle version
opt_gmv_mle <- optimize.portfolio(R = returns, portfolio = port_gmv,
    optimize_method = "ROI", momentFUN = "MlePrecompMomentFUN",
    trace = TRUE)

## Robust version
opt_gmv_rob <- optimize.portfolio(R = returns, portfolio = port_gmv,
    optimize_method = "ROI", momentFUN = "RobPrecompMomentFUN",
    trace = TRUE)


extractWeights(opt_gmv_mle)
```

```
##         MMM         ABT         ANF        ADBE         AMD
##  3.333e-01  3.309e-01  0.000e+00 -5.761e-18  8.891e-20
##           A         APD         AKS          AA         AYE
##  7.747e-02  1.202e-17  6.127e-02  1.349e-17  1.970e-01
```

```
extractWeights(opt_gmv_rob)
```

```
##         MMM         ABT         ANF        ADBE         AMD
##  2.169e-01  5.015e-01  4.754e-18 -5.108e-18  2.600e-17
##           A         APD         AKS          AA         AYE
##  1.491e-01  4.862e-17  1.325e-01  1.767e-19  0.000e+00
```