Google Summer Of Code 2021 Project Proposal

13 April, 2021

Project Information

Project Title:

Addition of Conic Solver to PortfolioAnalytics Package with Application to Expected Quadratic Shortfall and Robust Portfolio Optimization

Short Description:

In this project we will implement second-order cone (SOCopt) optimization as a capability of the R ***PortfolioAnalytics*** package. Our approach will be to use the ***CVXR***, which supports second-order cone optimization through open source solvers ECOS and SCS and through commercial solvers GUROBI and MOSEK. Applications use of the ***PortfolioAnalytics*** implementation will include portfolio optimization using expected quadratic shortfall (EQS) and Robust Portfolio Optimization (RPO) as implemented in the Axioma product.

Student Information

I'm a PhD student in Economics at the University of Washington, Seattle. My background includes a Bachelor's degree in Computer Science & Engineering (2004) from Calcutta in India, and a Master's in Economics (2010) from Delhi School of Economics, India. After completion of my Bachelor's, I worked as a Software Engineer at Cognizant Technology Solutions for more than 3.5 years. Upon completion of my Master's degree I worked in the Business Intelligence Unit (BIU) of the ICICI Bank where I became proficient at handling large datasets using SAS. To further pursue my research interests, later I joined the National Institute of Public Finance and Policy (NIPFP) in India and worked as a consultant in the Macroeconomics and Finance Research Program. Later, I also worked at Roubini Global Economics (RGE) (India) Pvt. Ltd. as an Economist in the macroeconomics research team.

My research interests include Financial Econometrics, Financial Modeling, Quantitative Risk Analysis, Portfolio Optimization, Time-Series econometrics and R package development. I had been introduced to R while I was pursuing my Master's in Economics in India. I gained substantial experience in R programming when I was working as a researcher at NIPFP. My association with R grew stronger when I joined the PhD program in Economics at University of Washington, Seattle. I was fortunate to take courses from Professor Eric Zivot and Guy Yollin that dealt not only with computational methods using R but also provided a strong theoretical background. In the past, I also worked as a Teaching Assistant in

the Computational Finance & Risk Management (CFRM) program at University of Washington.

I was awarded Google Summer of Code 2017 funding in the amount of $6,000 for the project *Portfolio Construction and Risk Management with Unequal Returns Histories* I gained valuable experience in the combination of research and coding in R under the guidance of Prof. Douglas Martin, who was my lead mentor. I developed an R package that deals with unequal return histories of assets with two different methods – Multiple Imputation (MI) and Combined Backfilling (CBF). For the MI method I implemented two methods of computing standard errors, one a bootstrap method and the other a method described in the second edition of the 2002 book by Little and Rubin. The R package shows the comparative study of the efficacy of the two methods and supports portfolio construction and risk management for portfolios that have unequal returns histories.

As a Software Engineer at Cognizant I gained experience in all the phases of the Software Development Lifecycle (SDLC) such as requirements analysis, design, development and testing of software development projects. After I was promoted to the level of an Associate, I worked as an offshore project-lead for software development and enhancement projects and dealt with clients from the United States. While my technical expertise at Cognizant included Java and TIBCO suite of products, I also worked extensively on the Technical Design Documents of the projects. I was also a member of the Enterprise Application Integration Centre of Excellence (EAICoE), an intra-organization technical expertise group at Cognizant. I received Retail & Hospitality Excellence Award for outstanding contribution to EAI projects at Cognizant. Later as a researcher for the NIPFP–DEA (Department of Economic Affairs) program (a collaboration between NIPFP and the Ministry of Finance Government of India), I worked on the Seasonal Adjustment of time-series data using the X-12 ARIMA system developed by the U.S. Census Bureau. At NIPFP, I constructed the daily time-series of Foreign Institutional Investment (FII) holding in Indian companies from FII transaction level data published by Securities and Exchange Board of India (SEBI), the Business Cycle Indicator (coincident) to monitor real time economic activity in India and developed and maintained a repository of time-series macroeconomic and financial data. As an Economist at Roubini Global Economics I focused on the U.S. economy and contributed to research notes on U.S. growth and inflation developments, and dynamics of the different sectors of the economy. I also leveraged my quantitative skills in producing forecasts for key short term U.S. economic indicators, which were disseminated to clients on a weekly basis and constructed a U.S. economic activity index.

I also worked as an Economist Intern, for Amazon in the Summer of 2019 on a project ***"Cold Start" Forecasting***. In this project I worked with the Topline Forecasting team in Amazon's Supply Chain

organization on a demand forecasting project for a very short history ("cold start") daily seasonal time series. This was done using a multivariate approach where rich cross-sectional information from other related longer history time series have been utilized. Multi-horizon forecasts were produced by two methods - 1) Factor Model and 2) Panel Estimation. The forecasts showed higher accuracy vis-à-vis the current production model.

In the Summer of 2020, I worked as a Research Scientist Intern for the Core AI team in Amazon on a project ***Prediction of Supply Risk***. In this project, I worked with the Inventory Planning and Control (IPC) team in Amazon's Supply Chain organization on a forecasting project related to the prediction of supply risk due to events like COVID-19. This was done using tools from the factor model framework (used in asset pricing literature) such as - statistical factor model, fundamental factor model and time series factor model.

Throughout my professional career, I worked under multiple projects and it taught me to deliver under pressure within the given deadline, while not compromising with the quality of the product. In my opinion, this is the most important factor to be considered while developing any product and I am confident that I can carry forward the same in this project as well. This is an excellent project for me to showcase my skills and learn under the guidance of Professor Douglas Martin. I am confident that I will be able to accomplish the project goals within the given timeline while also broadening my knowledge in R.

Student's Contact Information

Student Name: Pushpak Sarathi Sarkar

Student postal address: 3927 Adams Lane NE, Mercer Court-D #502A, Seattle, Washington - 98105

Telephone(s): 206-422-4945

Email(s): pushpak@uw.edu, sarkar.pushpak@gmail.com

Other communications channels: Zoom/Skype/Google+

Student Affiliation

Institution: University of Washington, Seattle

Program: Economics (PhD)

Expected date of Graduation: August 2021

Contacts to verify enrollment:

- Simon Reeve-Parker, Graduate Program Counselor, Department of Economics, University of Wash-

ington (Email: simonrp@uw.edu)

- Prof. Doug Martin, a mentor for this project, Professor Emeritus of Applied Mathematics, University of Washington (Email: martinrd3d@gmail.com, doug@amath.washington.edu).

Mentors

Mentor - Prof. Douglas Martin (Email: martinrd3d@gmail.com, doug@amath.washington.edu)

Mentor - Brian Peterson (Email: brian@braverock.com)

Have you been in touch with the mentors? When and how?

Yes, I've been working with Prof. Doug Martin with respect to this proposal. I had regular weekly meetings and email communication with Prof. Martin. I expect to have ongoing conversations and I'll continue to keep in touch with my mentors regularly, with a weekly update and email and/or phone discussions as needed.

I have done some preliminary exploration of the *CVXR* package and I'll continue to make myself acquainted with the package before the formal start of the coding session from June 7, 2021.

Motivation for PortfolioAnalytics SOCPortopt

Second Moment Coeherent Risk (SMCR) or Expected Qudratic Shortfall (EQS) portfolio optimization

The classical Mean variance Optimization (MVO) framework has been the workhorse model in the portfolio optimization since Markowitz (1952, 1959). But in this approach both positive and negative deviations from the expected level are penalized equally. But it does not always yield an adequate estimation of risk. Researchers have explored different measures of downside risk to rectify it. Value-at-Risk (VaR) and Conditional Value-at-Risk (CVaR) or Expected Shortfall (ES) are two such measures which have received attention in the context of optimal portfolio construction. But VaR does not satisfy the properties of coherent risk measures as shown by Artzner et al. (1999) and it is also non-convex. Rockafellar, Uryasev, and others (2000) and Rockafellar and Uryasev (2002) used expected shortfall as the risk estimate in the portfolio optimization context.

Krokhmal (2007) further extended these ideas by proposing a *Higher Moment Risk Measure* (HMCR) which is defined as

$$HMCR_{p,\alpha}(X) = \arg\min_{\eta\in\mathbb{R}} \eta + (1-\alpha)^{-1} \left\|(X-\eta)^+\right\|_p$$

$p \geq 1, \alpha \in (0,1)$

It also showed that the implementation of HMCR measures reduces to a p-order conic programming and can be approximated via linear programming.

The case $p = 2$ defines the *Second Moment Risk Measure* (SMCR)

$$SMCR_{\alpha}(X) = \arg\min_{\eta\in\mathbb{R}} \eta + (1-\alpha)^{-1} \left\|(X-\eta)^+\right\|_2$$

, $0 < \alpha < 1$

The $SMCR_{\alpha}(X)$ has similar properties as $CVaR_{\alpha}(X)$ but it measures risk in terms of the second moments of loss distributions. To formulate the SMCR measure into a mathematical programming problem second-order cone constraints should be used. The necessary tool to solve convex optimization problems involving second-order cone constraints is the second-order cone programming (SOCP). Krokhmal (2007) outlines an algorithm how a portfolio optimization problem of minimizing *HMCR* can be transformed into a linear programming problem with a single p-order cone constraint.

Robust Portfolio Optimization

It is a well known fact in the literature that Mean Variance optimizer is very sensitive to small variation in expected returns. Small perturbations in expected returns might lead to drastic changes in asset weights of a portfolio. For a thorough discussion on robust portfolio optimization with respect to errors in expected returns estimates, see Ceria and Stubbs (2006). Due to estimation error the actual efficient frontier can be far away from the true and estimated efficient frontiers.

Let's assume the n dimensional vector of true expected returns $\boldsymbol{\alpha}$ be normally distributed. Let $\bar{\boldsymbol{\alpha}}$ be the estimate of expected returns and let $\boldsymbol{\Sigma}$ be the covariance matrix of the estimates of expected returns. It is also assumed that the true expected returns lies within the confidence region

$$(\boldsymbol{\alpha} - \bar{\boldsymbol{\alpha}})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\alpha} - \bar{\boldsymbol{\alpha}}) \leq \kappa^2$$

with probability 100η per cent where $\kappa^2 = \chi_n^2(1-\eta)$ and χ_n^2 is the quantile function of the chi-squared distribution with n degrees of freedom. Consider the optimization problem of maximizing $\bar{\boldsymbol{\alpha}}^T w$ subject to $w^T Q w \leq \nu$ where Q is the covariance matrix of returns and ν is the target portfolio variance. The optimal holdings weight vector is

$$w = \sqrt{\frac{\nu}{\boldsymbol{\alpha}^T Q^{-1} \boldsymbol{\alpha}}} Q^{-1} \boldsymbol{\alpha}$$

.

Let $\boldsymbol{\alpha}^*$ be the true but unknown expected return vector, and let $\bar{\boldsymbol{\alpha}}$ be an expected return estimate. The true expected return of a portfolio on the estimated frontier is given by $\sqrt{\frac{\nu}{\bar{\boldsymbol{\alpha}}^T Q^{-1} \bar{\boldsymbol{\alpha}}}} \boldsymbol{\alpha}^{*\boldsymbol{T}} Q^{-1} \bar{\boldsymbol{\alpha}}$. Let $\tilde{w}$ be the optimal portfolio weight vector on the estimated frontier for a given target risk level. Then the maximum difference between the estimated expected return and the actual expected return is given as $\bar{\boldsymbol{\alpha}}^{\boldsymbol{T}} \tilde{w} - \boldsymbol{\alpha}^{*\boldsymbol{T}} \tilde{w}$. It can be shown (see Ceria and Stubbs (2006)) that the maximum difference between the estimated frontier and the actual frontier is $\boldsymbol{\kappa} \left\| \Sigma^{1/2} \tilde{w} \right\|$ where $\|.\|$ refers to 2-norm. We would like this difference to be as small as possible. To choose the optimal weights w, we solve the following long-only robust portfolio satisfying a budget constraint and a variance constraint

Maximize $\bar{\boldsymbol{\alpha}}^{\boldsymbol{T}} \boldsymbol{w} - \boldsymbol{\kappa} \left\| \Sigma^{1/2} w \right\|$ subject to $e^T w = 1$, $w^T Q w \leq \nu$, $w \geq 0$

The minimum volatility problem can be formulated as the following

Minimize $w^T Q w$ subject to $\bar{\boldsymbol{\alpha}}^{\boldsymbol{T}} \boldsymbol{w} - \boldsymbol{\kappa} \left\| \Sigma^{1/2} w \right\| \geq r$

The above problems can not be solved by a general purpose quadratic optimizer as the estimation-error term is 2-norm which contains a square root and can not be formulated as a pure quadratic problem. These type of robust optimization problem can be solved by a symmetric second-order cone optimizer. Hence adding conic solver functionality to *PortfolioAnalytics* will enable us to solve robust optimization problems, similar to the robust portfolio optimization tools offered by Axioma.

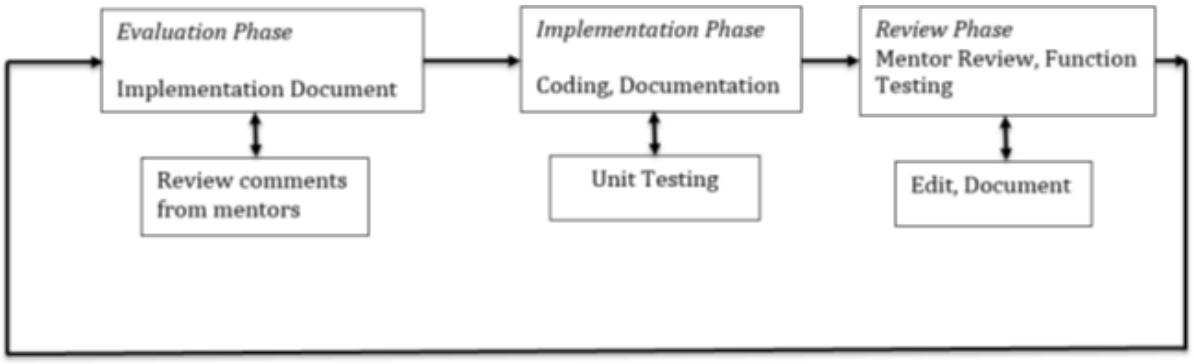
What is CVXR?

CVXR is an R package that provides a flexible object-oriented modeling language for convex optimization. It allows the user to formulate convex optimization problems in a natural mathematical syntax. The user can specify an objective and set of constraints by combining constants, variables, and parameters using a library of functions. CVXR verifies the problem's convexity and converts the problem into standard conic form and passes it to an appropriate solver such as the QP and LP solver *OSQP*, or a cone solver such as *ECOS* or *SCS*. For details see Fu, Narasimhan, and Boyd (2020).

Coding Plan and Methods

I intend to use an iterative and incremental software development framework with sets of two week sprints. At the beginning of each week, I will communicate my plan for the sprint and get feedback from the mentors. I will present to them an implementation document which will list the tasks I plan to accomplish in the next two weeks - listing all the functions, relevant testing and any documentation to be done, so that they can review my plan and make appropriate changes if required.

Each sprint should lead to finished portions of the project and hence minimizing the chances of work getting piled up in the end. At the completion of each sprint, I will review the progress of the sprint, document any tasks that were not completed and pass it to the mentors for review and get their feedback and document any changes suggested, so that they can be incorporated in the next sprint. The workflow in each sprint will be as shown below.



Key Deliverables

1. Implement the static and dynamic (with rebalancing) portfolio optimization by minimizing expected quadratic shortfall using the R package *CVXR*.
2. I'll be testing EQS on weekly and daily returns, where EQS can take advantage of controlling downside risk associated with the greater non-normality of weekly and daily returns relative to monthly returns. I'll be able to obtain this data via my access to CRSP data via the UW Business School WRDS license (through the end of 2021).
3. The next step would be to get the SOCopt working in *PortfolioAnalytics* (by adding the conic solver functionality) focusing on Expected Quadratic Shortfall (EQS).
4. Create a very extensive and detailed vignette for the EQS implementation that includes a substantial

number of usage examples.

5. The next step would be to get the Robust Portfolio Optimization (RobPortopt) working in *PortfolioAnalytics* (by adding the conic solver functionality).

6. Create a very extensive and detailed vignette for the RobPortOpt implementation that includes a substantial number of usage examples.

Implementation

The proposed package will be developed using GitHub as the development platform. This way the development process will take advantage of the facilities for version control, daily build and check, test cases, collaboration with mentors, etc.

Testing

As a Software Engineer, to deliver an efficient product, I have always been taught to do the testing of every element of the application before performing the overall test. The same methodology would be applied here. The testing can be differentiated as the white box testing and black box testing. White box testing is done by me during the implementation phase where I check every line of code through unit testing and the black box testing at the end of the sprint to check the overall functionality of the software. Each outcome will be documented and reviewed by the mentors at the end of each sprint.

We will use the facilities of test package to test the major package functions. During each sprint, tests will be added to the test suite to validate modified code. We will also cross-check the output results with the examples available in textbooks like Ruppert's Statistics and data analysis for financial engineering and Zivot, E., & Wang's Modeling Financial Time Series with S-PLUS etc. We could also compare our results with that generated from a commercially available package for greater accuracy.

Documentation

All functions and datasets will be documented using Roxygen2. For better understanding to the user, I will write code examples of function usage, and a vignette describing the equations and some relevant theory will be written to guide the user in the overall use of the function.

Timeline

Sprint 1: 6/7 - 6/13

Implement the static portfolio optimization by minimizing expected quadratic shortfall using the R package *CVXR*, perform unit testing and document the progress.

Sprint 2: 6/14 – 6/27

Implement the dynamic (with rebalancing) portfolio optimization by minimizing EQS using the R package *CVXR* second order cone capability with ECOS and SCS, perform unit testing and document the progress, backlogs.

Sprint 3: 6/28 – 7/18

Get SOCopt working in *PortfolioAnalytics* (by adding the conic solver functionality) focusing on Expected Quadratic Shortfall (EQS), perform unit testing and document the progress, backlogs.

Sprint 4: 7/19 – 8/8

Get RobPortopt working in *PortfolioAnalytics* (by adding the conic solver functionality) focusing on Robust Portfolio Optimization, perform unit testing and document the progress, backlogs.

Sprint 5: 8/9 – 8/15

Create a very extensive and detailed vignette that includes a substantial number of usage examples., perform unit testing and document the progress, backlogs.

Sprint 6: 8/16 – 8/23

Wrap up any remaining issues, test all the functionalities and incorporate mentor feedback.

What is your contingency plan for things not going to schedule?

As we follow an iterative and incremental software development procedure, any delays will be addressed every other week, thus avoiding a pile-up of falling behind schedule. Also, we follow an adaptive method, so that necessary changes can be incorporated on the go when required and make appropriate corrections to the future schedules. I'm also prepared to continue working on the project past the end date if necessary to deliver the key objectives.

Management of Coding Project

How do you propose to ensure code is submitted / tested?

I'll be meeting with one or more of the mentors every week or two to plan and review each sprint, and discuss related design issues, particularly those that have to do with efficiency of computation and usability. The proposed package will be developed using GitHub as the development platform. This way the development process will take advantage of the facilities for version control, daily build and check, collaboration with mentors, etc. With the guidance from mentors, a checklist can be prepared in the implementation phase for each functionality to be implemented in the next sprint, to make sure all the functionalities of the implemented function have been currently implemented and tested properly at the testing phase. The effectiveness of the work will be reviewed by the mentors at the review phase i.e. at the end of each sprint. Any feedback from mentors, failing test case scenarios etc. should be documented, so that they can be fixed in the next sprint.

Sample Code

Following is a basic layout of prototype code implementing basic portfolio optimization problems using the R packages *CVXR* and *PortfolioAnalytics*. This is a preliminary version of the code and it will be refined as we go along.

The following code snippets show how to use *CVXR* and *PortfolioAnalytics* to solve portfolio optimization problems based on different strategies of minimizing the risk of the portfolio - a) Global Minimum Variance (GMV), b) Global Minimum Expected Shortfall (GMES), c) Global Minimum Expected Quadratic Shortfall (GMEQS). When using *CVXR*, we also choose different solvers besides using the default solver.

1 GMV Portfolio Optimization

In this section we implement GMV using both *CVXR* and *PortfolioAnalytics* and use the We use 10 MidCap stocks from the CRSP data. First, we use *CVXR* and then *PortfolioAnalytics*. The following categories of GMV problem are solved - unconstrained GMV, long-only GMV, GMV with box constraint, GMV with box and short constraints.

1.1 Optimize portfolio using CVXR

In the section below we use *CVXR* with the default solver i.e. we don't explicitly specify the solver and let the package choose it while solving the problem. We also print the name of the solver which is used. *CVXR* uses default solver 'OSQP' for solving quadratic problem. The computed optimal weights are displayed in a table.

```
rm(list = ls())


# Load the necessary packages
library(CVXR)
library(xts)
library(kableExtra)


# Load the MidCap CRSP data
load('C:\\GoogleDrivePushpakUW\\Spillover\\Copula\\Optimization_Tail_Risk\\Data\\PCRM\\midcap.ts.rda')


# Select the first 10 stocks; compute the mean vector and covariance matrix of the returns data
returns <- midcap.ts[, 1:10]
n <- ncol(returns)
mu <- apply(returns, 2, mean)
mu <- matrix(mu, nrow = n)
Sigma <- cov(returns)
```

```
# Unconstrained GMV
## Form problem
opt_uc_w <- Variable(n)
ret <- t(mu) %*% opt_uc_w
risk <- quad_form(opt_uc_w, Sigma)
constraints <- list(sum(opt_uc_w) == 1)


objective <- risk
prob <- Problem(Minimize(objective), constraints)
result <- solve(prob)
```

```r
cat("Status of the solution: ", result$status)
```

Status of the solution: optimal

```r
cat("Solver used: ", result$solver)
```

Solver used: OSQP

```r
## Evaluate risk/return for current solution
opt_uc_risk <- result$getValue(sqrt(risk))
opt_uc_ret <- result$getValue(ret)

opt_uc_w <- result$getValue(opt_uc_w)
opt_uc_w <- t(opt_uc_w)
colnames(opt_uc_w) <- colnames(returns)

################################################################################
# Long only GMV
## Form problem
opt_lo_w <- Variable(n)
ret <- t(mu) %*% opt_lo_w
risk <- quad_form(opt_lo_w, Sigma)
constraints <- list(opt_lo_w >= 0, sum(opt_lo_w) == 1)

objective <- risk
prob <- Problem(Minimize(objective), constraints)
result <- solve(prob)
cat("Status of the solution: ", result$status)
```

Status of the solution: optimal

```r
cat("Solver used: ", result$solver)
```

Solver used: OSQP

```r
## Evaluate risk/return for current solution
opt_lo_risk <- result$getValue(sqrt(risk))
opt_lo_ret <- result$getValue(ret)

opt_lo_w <- result$getValue(opt_lo_w)
opt_lo_w <- t(opt_lo_w)
colnames(opt_lo_w) <- colnames(returns)

################################################################################
# Box GMV
## Form problem
opt_box_w <- Variable(n)
ret <- t(mu) %*% opt_box_w
risk <- quad_form(opt_box_w, Sigma)
constraints <- list(opt_box_w >= 0.03, opt_box_w <= 0.25, sum(opt_box_w) == 1)

objective <- risk
prob <- Problem(Minimize(objective), constraints)
result <- solve(prob)
cat("Status of the solution: ", result$status)
```

Status of the solution: optimal

```r
cat("Solver used: ", result$solver)
```

Solver used: OSQP

```r
## Evaluate risk/return for current solution
opt_box_risk <- result$getValue(sqrt(risk))
opt_box_ret <- result$getValue(ret)

opt_box_w <- result$getValue(opt_box_w)
opt_box_w <- t(opt_box_w)
colnames(opt_box_w) <- colnames(returns)
```

```
################################################################################
# Short box GMV
## Form problem
opt_shbox_w <- Variable(n)
ret <- t(mu) %*% opt_shbox_w
risk <- quad_form(opt_shbox_w, Sigma)
constraints <- list(opt_shbox_w >= -0.03, opt_shbox_w <= 0.25, sum(opt_shbox_w) == 1)


objective <- risk
prob <- Problem(Minimize(objective), constraints)
result <- solve(prob)
cat("Status of the solution: ", result$status)
```

Status of the solution: optimal

```
cat("Solver used: ", result$solver)
```

Solver used: OSQP

```
## Evaluate risk/return for current solution
opt_shbox_risk <- result$getValue(sqrt(risk))
opt_shbox_ret <- result$getValue(ret)


opt_shbox_w <- result$getValue(opt_shbox_w)
opt_shbox_w <- t(opt_shbox_w)
colnames(opt_shbox_w) <- colnames(returns)


################################################################################
# View the optimal weights of all portfolios
cvxr_wts <- rbind(opt_uc_w, opt_lo_w, opt_box_w, opt_shbox_w)
cvxr_wts <- round(cvxr_wts, digits = 3)
row.names(cvxr_wts) <- c("UNCONSTRAINED","LONG-ONLY","BOX","SHORT-BOX")
```

```
################################################################################
# View the mean return, std dev and Sharpe ratio
mean_ret <- c(opt_uc_ret, opt_lo_ret, opt_box_ret, opt_shbox_ret)
std_dev <- c(opt_uc_risk, opt_lo_risk, opt_box_risk, opt_shbox_risk)
sharpe_ratio <- mean_ret / std_dev


cvxr_ports <- cbind(mean_ret, std_dev, sharpe_ratio)
cvxr_ports <- round(cvxr_ports, digits = 3)
colnames(cvxr_ports) <- c("Mean Return","StdDev","Sharpe Ratio")
row.names(cvxr_ports) <- c("UNCONSTRAINED","LONG-ONLY","BOX","SHORT-BOX")
```

1.2 Optimize portfolio using PortfolioAnalytics

In the section below we use *PortfolioAnalytics* to solve the following categories of GMV problem - unconstrained GMV, long-only GMV, GMV with box constraint, GMV with box and short constraints. The computed optimal weights are displayed in a table.

```
# Portfolio optimization using PortfolioAnalytics
library(PortfolioAnalytics)
library(ROI)
library(ROI.plugin.quadprog)
library(ROI.plugin.glpk)


###
opt.outputMvo <- function(opt, returns, digits = NULL, names = NULL, rf = 0) {
    wts = opt$weights
    sigmasq = as.numeric(t(wts)%*%var(returns)%*%wts)
    sigma = sqrt(sigmasq)
    mu.ret = apply(returns,2,mean)
    mu = as.numeric(t(wts)%*%mu.ret)
    sr = (mu-rf)/sigma
    if(is.null(digits))
    {names(sigma) = "sigma"
```

```
    names(mu) = "mu"
    output = c(wts,mu,sigma)} else
    {if(is.null(names))
    {output = list(wts=wts,mean=mu,stdev=sigma,sr=sr)
    output = lapply(output,round,digits)}
      else
      {output = list(wts,mu,sigma,sr)
      names(output) = names
      output = lapply(output,round,digits)}
    }

    return(output)
  }
```

```
funds <- colnames(returns)
pspec.base <- portfolio.spec(funds)

# Unconstrained GMV
pspec.fi <- add.constraint(portfolio = pspec.base, type = "full_investment")
pspec.uc <- add.objective(portfolio = pspec.fi, type = "risk",name = "var")

# Optimize unconstrained GMV
opt.uc <- optimize.portfolio(R = returns, portfolio = pspec.uc,
                             optimize_method = "quadprog")
names <- c("WTS.UC","MU.UC","SD.UC","SR.UC")
out.uc <- opt.outputMvo(opt.uc,returns,names=names,digits = 3)

# Long only GMV
pspec.lo <- add.constraint(portfolio = pspec.uc, type = "long_only")

# Box GMV
pspec.box <- add.constraint(portfolio = pspec.lo, type = "box", min = .03,
```

```
                                max = .25, indexnum = 2)


# Box GMV with shorting
pspec.shbox <- add.constraint(portfolio = pspec.lo, type = "box", min = -.03,
                                  max = .25, indexnum = 2)


# Optimize long only GMV
opt.lo <- optimize.portfolio(returns, pspec.lo, optimize_method = "quadprog")
names <- c("WTS.LO","MU.LO","SD.LO","SR.LO")
out.lo <- opt.outputMvo(opt.lo,returns,names=names,digits = 3)


# Optimize box GMV
opt.box <- optimize.portfolio(returns, pspec.box, optimize_method = "quadprog")
names <- c("WTS.BOX","MU.BOX","SD.BOX","SR.BOX")
out.box <- opt.outputMvo(opt.box,returns,names=names,digits = 3)


# Optimize box GMV with shorting
opt.shbox <- optimize.portfolio(returns, pspec.shbox, optimize_method = "quadprog")
names <- c("WTS.SHBOX","MU.SHBOX","SD.SHBOX","SR.SHBOX")
out.shbox <- opt.outputMvo(opt.shbox,returns,names=names,digits = 3)


# View the optimal weights
compWts <- rbind(out.uc[[1]],out.lo[[1]],out.box[[1]],out.shbox[[1]])
compWts <- data.frame(compWts)
row.names(compWts) <- c("UNCONSTRAINED","LONG-ONLY","BOX","SHORT-BOX")


# View the mean return, StdDev and Sharpe ratio
compGmvPorts <- rbind(out.uc[2:4],out.lo[2:4],out.box[2:4],out.shbox[2:4])
compGmvPorts <- data.frame(compGmvPorts)
names(compGmvPorts) <- c("Mean Return","StdDev","Sharpe Ratio")
row.names(compGmvPorts) <- c("UNCONSTRAINED","LONG-ONLY","BOX","SHORT-BOX")
```

Below we show the GMV optimal weights computed by both PortfolioAnalytics and CVXR packages. We see the optimal weights from both packages are identical to three significant digits.

```
# Print the optimal weights table from CVXR
kable(cvxr_wts, caption = "GMV optimal weights using CVXR", digits = 3,
      format = "pandoc")
```

Table 1: GMV optimal weights using CVXR

	MAT	EMN	LEG	AAPL	UTR	HB	BNK	APA	LNCR	BMET
UNCONSTRAINED	0.264	-0.059	-0.085	0.151	0.407	0.166	-0.039	0.016	0.034	0.145
LONG-ONLY	0.224	0.000	0.000	0.132	0.322	0.164	0.000	0.011	0.011	0.136
BOX	0.200	0.030	0.030	0.124	0.250	0.161	0.030	0.030	0.030	0.115
SHORT-BOX	0.250	-0.030	-0.030	0.149	0.250	0.201	-0.015	0.030	0.040	0.155

```
# Print the optimal weights table from PortfolioAnalytics
kable(compWts, caption = "GMV optimal weights using PortfolioAnalytics", digits = 3,
      format = "pandoc")
```

Table 2: GMV optimal weights using PortfolioAnalytics

	MAT	EMN	LEG	AAPL	UTR	HB	BNK	APA	LNCR	BMET
UNCONSTRAINED	0.264	-0.059	-0.085	0.151	0.407	0.166	-0.039	0.016	0.034	0.145
LONG-ONLY	0.224	0.000	0.000	0.132	0.322	0.164	0.000	0.011	0.011	0.136
BOX	0.200	0.030	0.030	0.124	0.250	0.161	0.030	0.030	0.030	0.115
SHORT-BOX	0.250	-0.030	-0.030	0.149	0.250	0.201	-0.015	0.030	0.040	0.155

The tables below show the portfolio mean return, portfolio volatility and Sharpe ratio computed from both packages. These are identical to three significant digits.

```
# Print the mean return, volatility and Sharpe ratio from CVXR
kable(cvxr_ports, caption = "CVXR: Portfolio diagnostics", digits = 3,
      format = "pandoc")
```

Table 3: CVXR: Portfolio diagnostics

	Mean Return	StdDev	Sharpe Ratio
UNCONSTRAINED	0.017	0.052	0.319
LONG-ONLY	0.015	0.053	0.294
BOX	0.015	0.054	0.284
SHORT-BOX	0.017	0.053	0.316

```
# Print the mean return, volatility and Sharpe ratio from PortfolioAnalytics
kable(compGmvPorts, caption = "PortfolioAnalytics: Portfolio diagnostics", digits = 3,
      format = "pandoc")
```

Table 4: PortfolioAnalytics: Portfolio diagnostics

	Mean Return	StdDev	Sharpe Ratio
UNCONSTRAINED	0.017	0.052	0.319
LONG-ONLY	0.015	0.053	0.294
BOX	0.015	0.054	0.284
SHORT-BOX	0.017	0.053	0.316

2 Maximize Mean Return Subject to a Variance Constraint

Here we maximize portfolio mean return subject to a given variance constraint. This variance specifies the upper bound of the portfolio variance the investor will tolerate. To solve this problem we use three conic solvers of *CVXR* - ECOS, SCS and MOSEK.

```
# Maximize Mean return subject to a given variance constraint
# Risk tolerance - upper bound of the variance of the portfolio
risk_tol <- 0.005

## Form problem
opt_maxmean_w <- Variable(n)
```

```r
ret <- t(mu) %*% opt_maxmean_w

risk <- quad_form(opt_maxmean_w, Sigma)

constraints <- list(sum(opt_maxmean_w) == 1,

                    quad_form(opt_maxmean_w, Sigma) <= risk_tol)


objective <- ret

prob <- Problem(Minimize(objective), constraints)


result_mosek <- solve(prob)

cat("Status of the solution: ", result_mosek$status)
```

Status of the solution: optimal

```r
cat("Solver used: ", result_mosek$solver)
```

Solver used: MOSEK

```r
# We observe that default conic solver is MOSEK


# Evaluate risk/return for current solution

risk_mosek <- result_mosek$getValue(sqrt(risk))

mosek_ret <- result_mosek$getValue(ret)

mosek_wts <- result_mosek$getValue(opt_maxmean_w)

mosek_wts <- t(mosek_wts)

colnames(mosek_wts) <- colnames(returns)


# Specify 'SCS' solver

result_scs <- solve(prob, solver = 'SCS')

cat("Status of the solution: ", result_scs$status)
```

Status of the solution: optimal

```r
cat("Solver used: ", result_scs$solver)
```

Solver used: SCS

```r
scs_wts <- result_scs$getValue(opt_maxmean_w)
scs_wts <- t(scs_wts)
colnames(scs_wts) <- colnames(returns)

# Specify 'ECOS' solver
result_ecos <- solve(prob, solver = 'ECOS')
cat("Status of the solution: ", result_ecos$status)
```

Status of the solution: optimal

```r
cat("Solver used: ", result_ecos$solver)
```

Solver used: ECOS

```r
ecos_wts <- result_ecos$getValue(opt_maxmean_w)
ecos_wts <- t(ecos_wts)
colnames(ecos_wts) <- colnames(returns)

all_solver_wts <- rbind(mosek_wts, scs_wts, ecos_wts)
row.names(all_solver_wts) <- c("MOSEK", "SCS", "ECOS")
```

The table below shows the optimal weights of max-mean portfolio subject to a variance constraint. We see the weights are identical to four significant digits from three different solvers used by *CVXR*- ECOS, SCS and MOSEK.

```r
kable(all_solver_wts, caption = "MaxMean optimal weights using CVXR", digits = 3,
      format = "pandoc")
```

Table 5: MaxMean optimal weights using CVXR

	MAT	EMN	LEG	AAPL	UTR	HB	BNK	APA	LNCR	BMET
MOSEK	0.299	0.27	-0.056	0.063	0.487	0.222	0.031	0.015	-0.167	-0.164
SCS	0.299	0.27	-0.056	0.063	0.487	0.222	0.031	0.015	-0.167	-0.164
ECOS	0.299	0.27	-0.056	0.063	0.487	0.222	0.031	0.015	-0.167	-0.164

3 Global Minimum Expected Shortfall (GMES) Portfolio Optimization

We solve Global Minimum Expected Shortfall (GMES) portfolio optimization problem using both *CVXR* and *PortfolioAnaytics* packages. We use 10 SmallCap stocks from the CRSP data.

3.1 GMES: optimize portfolio using PortfolioAnaytics

```
load('C:\\GoogleDrivePushpakUW\\Spillover\\Copula\\Optimization_Tail_Risk\\Data\\PCRM\\smallcapM.rda')


returns <- smallcapM[,1:10]
funds <- colnames(returns)


pspec <- portfolio.spec(assets=funds)
pspec.fi <- add.constraint(pspec, type="full_investment")
pspec.lo <- add.constraint(pspec.fi, type="long_only")
pspec.gmetlLo <- add.objective(pspec.lo, type="risk", name="ES",
                               arguments=list(p=0.95))


bt.gmetlLo <- optimize.portfolio(returns, pspec.gmetlLo,
                                 optimize_method="glpk")


# Extract time series of portfolio weights
wts.gmetlLo <- extractWeights(bt.gmetlLo)
```

3.2 GMES: optimize portfolio using CVXR

To solve the GMES portfolio optimization problem, we show the usage of *CVXR* package with different solvers. Along with the default solver, we also use SCS, ECOS, GLPK and MOSEK solvers.

```
alpha <- 0.95
t <- nrow(returns)
N <- ncol(returns)
X <- as.matrix(returns)
```

```r
mu <- colMeans(returns)


# variables
w <- Variable(N)
z <- Variable(t)
zeta <- Variable(1)


# Objective
objective_es <- zeta + (1/(t*(1-alpha))) * sum(z)


# problem
prob_es <- Problem(Minimize(objective_es),
                   constraints = list(z >= 0,
                                      z >= -X %*% w - zeta,
                                      w >= 0, sum(w) == 1))


# Using default solver
result_es <- solve(prob_es)
cat("Status of the solution: ", result_es$status)
```

Status of the solution: optimal

```r
cat("Solver used: ", result_es$solver)
```

Solver used: OSQP

```r
cvxr_es_wts <- result_es$getValue(w)
cvxr_es_wts <- t(cvxr_es_wts)
colnames(cvxr_es_wts) <- colnames(returns)


# Using SCS solver
es_result_scs <- solve(prob_es, solver = 'SCS')
cat("Status of the solution: ", es_result_scs$status)
```

Status of the solution: optimal

```r
cat("Solver used: ", es_result_scs$solver)
```

Solver used: SCS

```r
cvxr_scs_es_wts <- es_result_scs$getValue(w)
cvxr_scs_es_wts <- t(cvxr_scs_es_wts)
colnames(cvxr_scs_es_wts) <- colnames(returns)

# Using ECOS solver
es_result_ecos <- solve(prob_es, solver = 'ECOS')
cat("Status of the solution: ", es_result_ecos$status)
```

Status of the solution: optimal

```r
cat("Solver used: ", es_result_ecos$solver)
```

Solver used: ECOS

```r
cvxr_ecos_es_wts <- es_result_ecos$getValue(w)
cvxr_ecos_es_wts <- t(cvxr_ecos_es_wts)
colnames(cvxr_ecos_es_wts) <- colnames(returns)

# Using GLPK solver
es_result_glpk <- solve(prob_es, solver = 'GLPK')
cat("Status of the solution: ", es_result_glpk$status)
```

Status of the solution: optimal

```r
cat("Solver used: ", es_result_glpk$solver)
```

Solver used: GLPK

```r
cvxr_glpk_es_wts <- es_result_glpk$getValue(w)
cvxr_glpk_es_wts <- t(cvxr_glpk_es_wts)
colnames(cvxr_glpk_es_wts) <- colnames(returns)

# Using MOSEK solver
es_result_mosek <- solve(prob_es, solver = 'MOSEK')
```

```
cat("Status of the solution: ", es_result_mosek$status)
```

Status of the solution: optimal

```
cat("Solver used: ", es_result_mosek$solver)
```

Solver used: MOSEK

```
cvxr_mosek_es_wts <- es_result_mosek$getValue(w)
cvxr_mosek_es_wts <- t(cvxr_mosek_es_wts)
colnames(cvxr_mosek_es_wts) <- colnames(returns)

all_es_wts <- rbind(wts.gmetlLo, cvxr_es_wts, cvxr_scs_es_wts, cvxr_ecos_es_wts,
                    cvxr_mosek_es_wts, cvxr_glpk_es_wts)

column_names <- colnames(all_es_wts)
row_names <- c("PortfolioAnalytics", "CVXR-default", "CVXR-SCS",
                           "CVXR-ECOS", "CVXR-MOSEK", "CVXR-GLPK")
all_es_wts <- t(all_es_wts)
colnames(all_es_wts) <- row_names
row.names(all_es_wts) <- column_names
```

The following table displays the optimal weights from the above portfolio optimization results using PortfolioAnalytics and CVXR (with different solvers). The results of PortfolioAnalytics and CVXR with ECOS, MOSEK and GLPK solvers are identical to four significant digits. The results of CVXR with OSQP (default) and SCS solver are slightly different but they are same if we consider up to two significant digits.

```
kable(all_es_wts, caption = "GMES: optimal weights", digits = 4,
      format = "pandoc")
```

Table 6: GMES: optimal weights

	PortfolioAnalytics	CVXR-default	CVXR-SCS	CVXR-ECOS	CVXR-MOSEK	CVXR-GLPK
PLXS	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
BWINB	0.2405	0.2439	0.2448	0.2405	0.2405	0.2405

	PortfolioAnalytics	CVXR-default	CVXR-SCS	CVXR-ECOS	CVXR-MOSEK	CVXR-GLPK
HGIC	0.1068	0.1054	0.1050	0.1068	0.1068	0.1068
WTS	0.2221	0.2225	0.2224	0.2221	0.2221	0.2221
HTLD	0.2589	0.2567	0.2568	0.2589	0.2589	0.2589
PLAB	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
OFG	0.0193	0.0192	0.0192	0.0193	0.0193	0.0193
WSBC	0.0977	0.0989	0.0991	0.0977	0.0977	0.0977
CCC	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
SUR	0.0547	0.0532	0.0527	0.0547	0.0547	0.0547

4 Second Moment Coeherent Risk (SMCR) or Expected Qudratic Shortfall (EQS) Portfolio Optimization

To solve the EQS portfolio optimization problem (following Krokhmal (2007)), we use *CVXR* package with different conic solvers - SCS, ECOS, MOSEK and GUROBI. We find the default solver used by *CVXR* is MOSEK.

```
# EQS
alpha <- 0.95
t <- nrow(returns)
N <- ncol(returns)
X <- as.matrix(returns)
mu <- colMeans(returns)


# variables
eqs_wts <- Variable(N)
z <- Variable(t)
zeta <- Variable(1)


# Objective
objective_eqs <- zeta + (1/(1-alpha)) * p_norm(z, p=2)
```

```
# problem
prob_eqs <- Problem(Minimize(objective_eqs),
                    constraints = list(z >= 0,
                                       z >= -X %*% eqs_wts - zeta,
                                       eqs_wts >= 0, sum(eqs_wts) == 1))
result_eqs <- solve(prob_eqs)
cat("Status of the solution: ", result_eqs$status)
```

Status of the solution: optimal

```
cat("Solver used: ", result_eqs$solver)
```

Solver used: MOSEK

```
cvxr_eqs_wts <- result_eqs$getValue(eqs_wts)
cvxr_eqs_wts <- t(cvxr_eqs_wts)
colnames(cvxr_eqs_wts) <- colnames(returns)


###
result_eqs_scs <- solve(prob_eqs, solver = 'SCS')
cat("Status of the solution: ", result_eqs_scs$status)
```

Status of the solution: optimal

```
cat("Solver used: ", result_eqs_scs$solver)
```

Solver used: SCS

```
cvxr_scs_eqs_wts <- result_eqs_scs$getValue(eqs_wts)
cvxr_scs_eqs_wts <- t(cvxr_scs_eqs_wts)
colnames(cvxr_scs_eqs_wts) <- colnames(returns)


###
result_eqs_ecos <- solve(prob_eqs, solver = 'ECOS')
cat("Status of the solution: ", result_eqs_ecos$status)
```

Status of the solution: optimal

```
cat("Solver used: ", result_eqs_ecos$solver)
```

Solver used: ECOS

```
cvxr_ecos_eqs_wts <- result_eqs_ecos$getValue(eqs_wts)
cvxr_ecos_eqs_wts <- t(cvxr_ecos_eqs_wts)
colnames(cvxr_ecos_eqs_wts) <- colnames(returns)


###
result_eqs_gurobi <- solve(prob_eqs, solver = 'GUROBI')
cat("Status of the solution: ", result_eqs_gurobi$status)
```

Status of the solution: optimal

```
cat("Solver used: ", result_eqs_gurobi$solver)
```

Solver used: GUROBI

```
cvxr_gurobi_eqs_wts <- result_eqs_gurobi$getValue(eqs_wts)
cvxr_gurobi_eqs_wts <- t(cvxr_gurobi_eqs_wts)
colnames(cvxr_gurobi_eqs_wts) <- colnames(returns)


# Combine all weights
all_eqs_wts <- rbind(cvxr_eqs_wts, cvxr_scs_eqs_wts, cvxr_ecos_eqs_wts,
                     cvxr_gurobi_eqs_wts)


es_eqs_wts <- rbind(wts.gmetlLo, all_eqs_wts)
row.names(es_eqs_wts) <- c("GMES", "GMEQS-MOSEK", "GMEQS-SCS", "GMEQS-ECOS",
                           "GMEQS-GUROBI")
```

The table below shows the optimal weights of 10 stocks obtained by GMES and GMEQS portfolio optimization. For GMES we used PortfolioAnalytics and for GMEQS we used CVXR with four solvers - MOSEK, SCS, ECOS and GUROBI. It is interesting that all four of MOSEK, SCS, ECOS and GUROBI give identical answers to four significant digits, and they all set equal to zero two (stocks 'OFG' and 'WSBC') of the small non-zero coefficients of the GMES solution.

```
kable(es_eqs_wts, caption = "GMES and GMEQS: optimal weights", digits = 4,
format = "pandoc")
```

Table 7: GMES and GMEQS: optimal weights

	PLXS	BWINB	HGIC	WTS	HTLD	PLAB	OFG	WSBC	CCC	SUR
GMES	0	0.2405	0.1068	0.2221	0.2589	0	0.0193	0.0977	0	0.0547
GMEQS-MOSEK	0	0.2374	0.0000	0.3455	0.2829	0	0.0000	0.0000	0	0.1342
GMEQS-SCS	0	0.2374	0.0000	0.3454	0.2829	0	0.0000	0.0000	0	0.1342
GMEQS-ECOS	0	0.2374	0.0000	0.3455	0.2829	0	0.0000	0.0000	0	0.1342
GMEQS-GUROBI	0	0.2374	0.0000	0.3455	0.2829	0	0.0000	0.0000	0	0.1342

References

Artzner, Philippe, Freddy Delbaen, Jean-Marc Eber, and David Heath. 1999. "Coherent Measures of Risk." *Mathematical Finance* 9 (3): 203–28.

Ceria, Sebastián, and Robert A Stubbs. 2006. "Incorporating Estimation Errors into Portfolio Selection: Robust Portfolio Construction." *Journal of Asset Management* 7 (2): 109–27.

Fu, Anqi, Balasubramanian Narasimhan, and Stephen Boyd. 2020. "CVXR: An R Package for Disciplined Convex Optimization." *Journal of Statistical Software* 94 (14): 1–34. https://doi.org/10.18637/jss.v094.i14.

Krokhmal, Pavlo A. 2007. "Higher Moment Coherent Risk Measures."

Martin, R Douglas, Andrew Clark, and Christopher G Green. 2010. "Robust Portfolio Construction." In *Handbook of Portfolio Construction*, 337–80. Springer.

Rockafellar, R Tyrrell, and Stanislav Uryasev. 2002. "Conditional Value-at-Risk for General Loss Distributions." *Journal of Banking & Finance* 26 (7): 1443–71.

Rockafellar, R Tyrrell, Stanislav Uryasev, and others. 2000. "Optimization of Conditional Value-at-Risk." *Journal of Risk* 2: 21–42.