ERC OPTIMIZATION WITH ADDITIONAL CONSTRAINTS

Using the SCRIP Algorithm to find the optimal portfolio under multiple user defined constraints

Introduction

Portfolio optimization has been an area that has been explored widely in the past few years, with many novel ideas and interesting optimization methods being created. One such portfolio is the Equal Risk Contribution (ERC) or Risk Parity (RP) portfolio, which operates with the intent of assigning an equal amount of risk to each instrument in the portfolio. However, the ERC optimization currently used at SG operates with a long-only constraint, that is, it doesn't allow short selling of any kind. Another constraint is that the risk contributions are constrained to be exactly equal for all the securities, and customization of the same isn't viable.
This modified ERC optimization, called the Successive Convex Optimization for Risk Parity Portfolios (SCRIP) Algorithm, allows for vast flexibility in allowing short selling, choosing a target volatility, target risk contributions, target expected return and more. While it does have some drawbacks, it is very versatile and can be used to great effect at SG.

The Original ERC Strategy

Markowitz optimization aims to solve the following optimization problem:

$$\begin{gathered} min_x \;\; \mathbf{x}^T \Sigma \mathbf{x} \\ s.t. \;\; x_i \geq 0 \\ \mathbf{x}^T \mathbf{1} = 1 \end{gathered}$$

with x being the weights and Σ being the covariance matrix. The optimal solution here turns out to be the one where the marginal risk contributions of each security is the same, i.e,

$$\frac{\partial \sigma}{\partial x_i} = \frac{\partial \sigma}{\partial x_j} \qquad \forall i, j \in \{1, 2, ..., n\}$$

The RP approach, however, involves optimizing the weights such that the contribution of each security to the total volatility is the same, i.e,

$$x_i . \frac{\partial \sigma}{\partial x_i} = x_j . \frac{\partial \sigma}{\partial x_j} \qquad \forall i, j \in \{1, 2, ..., n\}$$

The derivation of this result comes from Euler's Theorem, and can be found in [1]. The convex optimization problem to be solved to get the RP portfolio is the following:

$$\begin{gathered} min_x \quad \sum_{i=1,j=1}^{n} (x_i(\Sigma x)_i - x_j(\Sigma x)_j)^2 \\ s.t. \;\; x_i \geq 0 \\ \sum_{i=1}^{n} x_i = 1 \end{gathered}$$

where $\mathcal{R}(x) = \sum_{i=1,j=1}^{n} (x_i(\Sigma x)_i - x_j(\Sigma x)_j)^2$ is the **risk concentration function** counting the total deviation from the theoretical RP condition. This method guarantees a solution as the convex optimization always has a unique global minimum. The proof of this can be found in [1] as well. This method is currently being used in the SG strategies such as the Harmonia, and is only valid for non-negative weights. In the next section we extend this strategy and remove the constraints on the weights to make it more versatile, and also describe a new optimization method to solve the problem.

The Proposed RP Strategy

This method is credited to Profs. Feng and Palomar in [2].
In this general approach, the problem we look to solve is the following non-convex nonlinear optimization problem:

$$\begin{gathered} min_x \;\; \mathcal{R}(\mathbf{x}) + \lambda F(\mathbf{x}) \\ s.t. \;\; x_i \in \mathcal{X} \\ \mathbf{x}^T \mathbf{1} = 1 \end{gathered}$$

where:

- $\mathcal{R}(\mathbf{x})$ denotes the risk concentration function. $\mathcal{R}(\mathbf{x})$ is also written as $\sum_{i=1}^{n}(g_i(x))^2$, where $g_i(x)$ measures the risk concentration brought about by the i-th asset
- $\mathcal{X}$ is a convex set of possible values of $\mathbf{x}$. It represents the investor's profiles, capital limitations, market regulations, etc.
- $F(\mathbf{x})$ is a convex function and denotes the portfolio preference, i.e, what the investor prefers to minimize. It could be the expected loss $\mu^T\mathbf{x}$ or the variance $\mathbf{x}^T\Sigma\mathbf{x}$ or a combination of both, etc.
- $\lambda \geq 0$ is the trade off between the portfolio preference and risk concentration.

This is a very general problem and allows for vast manipulations, and this has been discussed in detail in [2].
What we look at is the direct extension of the long-only strategy to this case, which gives:

$$min_x \quad \sum_{i=1,j=1}^{n}(x_i(\Sigma x)_i - x_j(\Sigma x)_j)^2 - \lambda\mu^T\mathbf{x} + \theta\mathbf{x}^T\Sigma\mathbf{x}$$
$$s.t. \;\; a_i \leq x_i \leq b_i$$
$$\mathbf{x}^T\mathbf{1} = 1$$

with $\lambda \geq 0,\ \theta \geq 0,\ a_i, b_i \in \mathbb{R}$
This is a highly non-convex optimization problem, and thus faces a very important hurdle : the issue of local minima.
As this function is non-convex, it is not required to (and in most cases does not) have a unique global minimum, but rather possesses multiple local minima. The usual non-convex algorithms like SQP and IMP fail to tackle this issue concretely, and are very inefficient in solving this problem. Thus, a new algorithm was developed in [2] by Feng and Palomar called the SCRIP Algorithm which is much more efficient, although not perfect, at solving this problem. I provide a brief description of the alorithm below, but the details are left out (they can be found in [2]).

The SCRIP Algorithm

For $k \geq 0$, the algorithm solves the following problem P^k:

$$min_x \quad \sum_{i=1}^{n}(g_i(x^k) + (\nabla g_i(x^k))^T(x - x^k))^2 + \frac{\tau}{2}\left\|(x - x^k)\right\|_2^2 + \lambda F(\mathbf{x})$$
$$s.t. \;\; x_i \in \mathcal{X}$$
$$\mathbf{x}^T\mathbf{1} = 1$$

where $\tau \geq 0$ is the regularization parameter.
This method convexifies the original risk concentration function by linearizing it by adding the gradient and then adding the proximal term for convergence.
The neat feature of this is that the first term has the same gradient as the original risk concentration function, which is why the algorithm works efficiently.
This problem is a standard convex optimization and can be done in many ways by efficient solvers.

The SCRIP Algorithm is the following:

- Select $k = 0,\ \tau > 0,\ x^0 \in \{x^T\mathbf{1} = 1\} \cap \mathcal{X},\ \{\gamma^k\}_{k\in\mathbb{N}} > 0$
- Solve P^k repeatedly to get the optimal solution $\hat{x}^k$
- Set $x^{k+1} = x^k + \gamma^k(\hat{x}^k - x^k)$
- Repeat until convergence.

Convergence and Other Advanced Algorithms

The convergence analysis is done in detail in [2], and under some pretty standard constraints followed by almost all portfolio design objects, there is a global stationary point that the algorithm will converge to.
Other advanced versions of this algorithm exist and can be examined as well for any efficiency time saves in the future. Some of these are also available in [2].

Code and Simulation

There currently is a package called **'RiskPartyPortfolio'** in both R and Python that is able to perform the SCRIP optimization and return the optimal weights. The Python package, however, does not currently run on Windows PCs due to the lack of availability of a certain prerequisite package that it is built upon (jaxlib). The packages themselves can be found at [3] and [4].
I ran a small simulation of this strategy on a portfolio of 3 securities - the **EuroStoxx 50 Index**, the **Nikkei 225 Index** and **AAPL stock** (these were arbitrarily chosen). After acquiring the data on the performance of these securities in the past 5 years from public sources, I then calculated the covariance matrix using R's inbuilt statistics library (I did not use a rolling window, however that would be the next step up in complexity in a simulation).

I then ran the RiskParityPortfolio package on said covariance matrix, also accounting for the expected returns of the securities. I varied the Lagrange parameter from 10^{-5} to 100 and plotted the various risk concentrations versus expected returns.

I did this in 3 cases :

1. The standard ERC algorithm, with the long-only constraint
2. The RP strategy with lower and upper bounds of -1 and 1 respectively
3. The RP strategy, but scaled up to allow lower and upper bounds of -20 and 20 respectively.

The code and the plotted results are shown in the Appendix.
The plots show that at risk concentrations of about 10^{-5}, Case 1 and Case 2 have expected returns of about **0.65%** and **1.4%** respectively. Case 3 is simply a scaled up version of Case 2, allowing for larger returns at proportionately higher risk contributions.
Further evidence of the new RP strategy's efficacy is found when we fit a volatility cap of **2.7%** (this was manually done and the number was arbitrarily chosen). The results are summarised in the 3 tables below.

Thus it is clearly visible that cases (2) and (3) greatly outperformed case (1) and have a wider variety of expected returns obtainable based on the investors' anticipations, needs and desires.

Statistic	1Y	3Y	5Y
Return	-0.194%	1.27%	0.81%
Volatility	1.53%	1.05%	1.04%
Sharpe Ratio	-0.127	1.216	0.784

Table 1: The Original Long Only ERC Strategy

Statistic	1Y	3Y	5Y
Return	0.375%	3.14%	1.9%
Volatility	2.65%	2.27%	2.03%
Sharpe Ratio	0.141	1.384	0.94

Table 2: The New RP Strategy with bounds of -1 and 1

Statistic	1Y	3Y	5Y
Return	0.195%	3.73%	2.02%
Volatility	2.24%	2.66%	2.13%
Sharpe Ratio	0.087	1.4	0.946

Table 3: The New RP Strategy with bounds of -20 and 20

Conclusion

To conclude, I believe that this optimization technique has the potential to outperform the current techniques in many of SG's flagship indices and allow for more versatility, with the possibility of higher returns at a specified risk concentration and volatility.

The salient feature of this algorithm is that even in the worst case scenario, it performs at a level equal to that which is currently in practice in SG Harmonia, and can only produce better (or equally as good) returns.

Introducing this optimization strategy into SG's indices will open up new possibilities that were previously unavailable, and thus, this technique may be used to achieve a performance better than that of the current indices.

References

1. Bai X. *et al.*. Least-squares approach to risk parity in portfolio selection. *Industrial and Systems Engineering Department, Lehigh University, Bethlehem, PA, USA. Goldman Sachs Asset Management, New York, NY, USA.*

2. Feng Y., Palomar D.P. SCRIP: Successive Convex Optimization Methods for Risk Parity Portfolio Design. *IEEE Transactions on Signal Processing, Vol. 63, No. 19, October 1, 2015*

3. https://mirca.github.io/riskparity.py/index.html

4. https://cran.r-project.org/web/packages/riskParityPortfolio/index.html

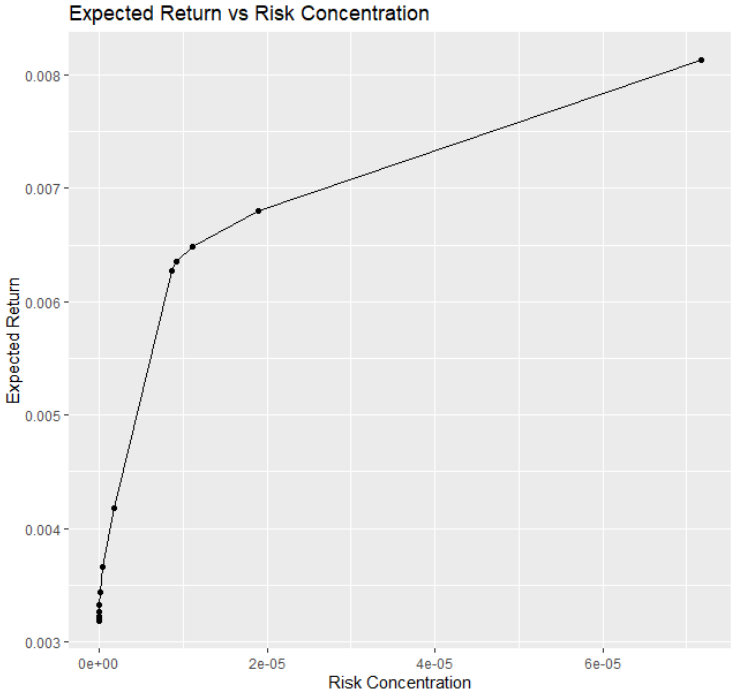
5. https://cran.r-project.org/web/packages/riskParityPortfolio/vignettes/RiskParityPortfolio.html

Appendix

Plots of the Simulation

Please find the plots and the code overleaf.

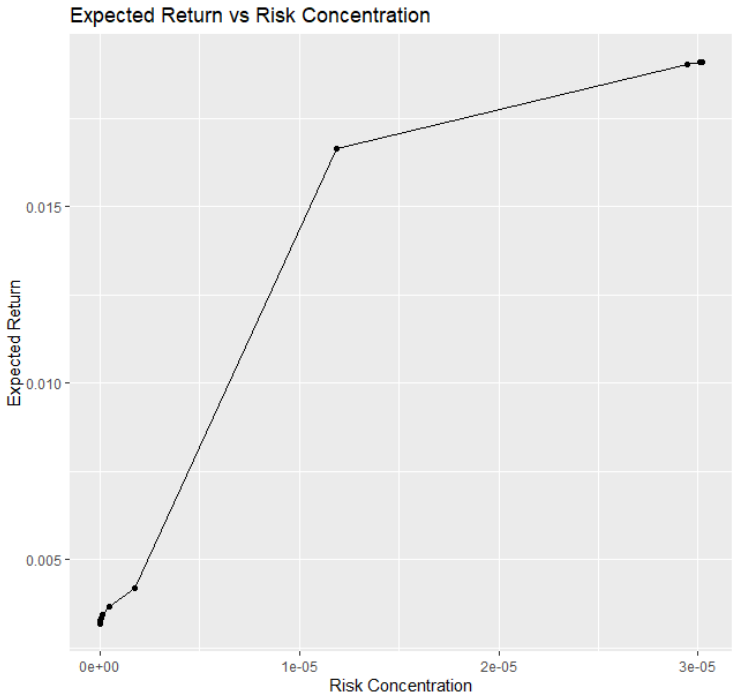
Figure 1: The Old ERC Strategy




```
library(riskParityPortfolio)
library(dplyr)
library(ggplot2)
# create covariance matrix
x <- read.csv('AAPL.csv')
x <- x %>% slice(1:1243)
mu = (as.numeric(tail(x, n = 1)/head(x, n = 1))) - 1
x <- log(x)
Sig <- cov(x)
lmd_sweep <- 10^seq(-5, 2, .25)
mean_return <- c()
risk_concentration <- c()
for (lmd_mu in lmd_sweep) {
  rpp <- riskParityPortfolio(Sig, mu = mu, lmd_mu = lmd_mu,
                             formulation = "rc-over-sd vs b-times-sd")
  mean_return <- c(mean_return, rpp$mean_return)
  risk_concentration <- c(risk_concentration, rpp$risk_concentration)
}

ggplot(data.frame(risk_concentration, mean_return),
       aes(x = risk_concentration, y = mean_return)) +
  geom_line() + geom_point() +
  labs(title = "Expected Return vs Risk Concentration",
       x = "Risk Concentration", y = "Expected Return")
```

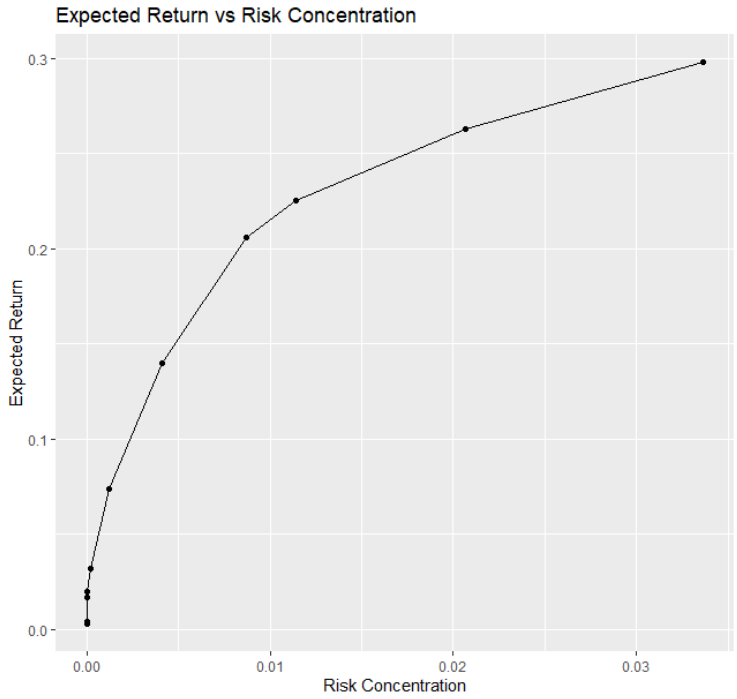
Figure 2: The New RP Strategy




```r
library(riskParityPortfolio)
library(dplyr)
library(ggplot2)
# create covariance matrix
x <- read.csv('AAPL.csv')
x <- x %>% slice(1:1243) #number of working days in the time period
mu = (as.numeric(tail(x, n = 1)/head(x, n = 1))) - 1 # calculating total return of the three securities
x <- log(x) #annualizing the returns
Sig <- cov(x)
vec1 <- c(-1, -1, -1)
vec2 <- c(1, 1, 1)
lmd_sweep <- 10^seq(-5, 2, .25)
mean_return <- c()
risk_concentration <- c()
for (lmd_mu in lmd_sweep) {
  rpp <- riskParityPortfolio(Sig, w_lb = vec1, w_ub = vec2, mu = mu, lmd_mu = lmd_mu,
                             formulation = "rc-over-sd vs b-times-sd")
  mean_return <- c(mean_return, rpp$mean_return)
  risk_concentration <- c(risk_concentration, rpp$risk_concentration)
}

ggplot(data.frame(risk_concentration, mean_return),
       aes(x = risk_concentration, y = mean_return)) +
  geom_line() + geom_point() +
  labs(title = "Expected Return vs Risk Concentration",
       x = "Risk Concentration", y = "Expected Return")
```

Figure 3: The New Scaled Up RP Strategy




```
library(riskParityPortfolio)
library(dplyr)
library(ggplot2)
# create covariance matrix
x <- read.csv('AAPL.csv')
#x <- x %>% slice(1:1243)
mu = (as.numeric(tail(x, n = 1)/head(x, n = 1))) - 1
x <- log(x)
Sig <- cov(x)
vec1 <- c(-20, -20, -20)
vec2 <- c(20, 20, 20)
lmd_sweep <- 10^seq(-5, 2, .25)
mean_return <- c()
risk_concentration <- c()
for (lmd_mu in lmd_sweep) {
  rpp <- riskParityPortfolio(Sig, w_lb = vec1, w_ub = vec2, mu = mu, lmd_mu = lmd_mu,
                             formulation = "rc-over-sd vs b-times-sd")
  mean_return <- c(mean_return, rpp$mean_return)
  risk_concentration <- c(risk_concentration, rpp$risk_concentration)

}

ggplot(data.frame(risk_concentration, mean_return),
       aes(x = risk_concentration, y = mean_return)) +
  geom_line() + geom_point() +
  labs(title = "Expected Return vs Risk Concentration",
       x = "Risk Concentration", y = "Expected Return")
```