TORI 於永明彩虹強積金計劃的回測 – Backtesting TORI in SunLife Rainbow Scheme

TORI (Trend- & Oversell-based Rebalancing Instructions) in SunLife Rainbow MPF Scheme

永明彩虹強積金應用 TORI 進行資產調配

Civil Servant Management fee pricing

公務員收費

Disclaimer

1. The author (jchan-gi) expressly stated that nothing my repositories and webpages constitutes any advices or recommendation on investing or allocating assets on any stock or financial products.

2. No content in my study have been prepared with respect to specific personal need and investment objects of individuals. All materials reveal to the public are applicable to the author (jchan-gi) only.
3. All investors shall obtain kind advices from professional financial consultants before making any decisions. Professional financial consultants should recommend products or decisions that suit your needs and objectives.

4. The author is not licensed nor professional in the field hence this studies are not professional advice and may not be suitable for anyone except myself.

5. You shall not invest on any financial products using the information on this code repository, website or any information made public by jchan-gi alone. You agree you have fully understand and willing to undertake the risks of this script in any circumstances involves in using this script.

6. English version shall be used if any discrepencies exists between English and Chinese version.

免責聲明

1. 本文作者現此聲明，本人的程式及網頁並非對閣下提供或作出對任何金融產品的投資或資產調配的建議。

2. 本文內容沒有針對閣下或其他人的實際個人需要而編撰。所有公開的內容只適用於本文作者。

3. 所有投資者應在作出任何決定前，先向專業理財顧問查詢及要求提供意見。只有理財顧問的意見才能符合閣下的實際理財需要及風險胃納。

4. 本文作者非專業人士或持牌從業者，因此，本文內容並非專業意見。而且，本文內容極有可能並不合適於除作者以外的任何人。
5. 在作出任何投資決定前，你不應單靠此程序作出決定。當你作出任何與本程序算法有關的投資決定時，你已完全知悉並願意接受本程序或算法所帶來的風險。

6. 本聲明如有歧義，將以英文版本作準。

最後更新日期 Last update: 2020/01/13.

Changelog

更新記錄

2020-01-13 v1
1. TORI is officially published and backtested with SunLife Rainbow MPF
1. 正式推出以永明彩虹強積金作回測的 TORI 介紹

WARNING: Please understand nature of TORI in this section before reading further:

This script relies on BIAS indicator and RSI indicator for investment rebalancing.
Please understand BIAS indicator and RSI indicator before reading further.
The script perform walk-forward validation only.
A model will be created for exactly next re-balance only based on most updated data for that moment.
There is no other validation or assessment on the data.
Please be reminded that backtesting and/or walk-forward validation is no way to be accurate in predicting future performance.
Using this script may exposed to various risk, including but not limited to overfitting, market, timing, etc.

警告：在應用 TORI 之前，請先於此節了解 TORI 的特性

TORI 根據乖離值及相對強弱指數作投資組合再配置。
閱讀下文前，請先了解乖離值及相對強弱指數。
此模式只使用了前移式回測 (walk-forward validation)。
每次資產調配都會基於擁有當時而言的最新資料。
然而，請切記過往表現不代表將來表現。
使用此代碼將會受到包括但不限於以下的風險所影響：過擬合風險、市場風險、時差風險等。

Background Introduction

Mandatory Provident Fund (MPF) is one of the legal pensions in Hongkong.
All Employers and some employees are required to pay at least 5% of monthly salary to MPF services provider.
MPF providers are required to provide approved funds for employees to invest, while the team could earn management fees.

However, the average annualized return of MPF is 3.1% only in from 2000-2016 (Mandatory Provident Fund Schemes Authority, 2016).
Most Hongkong employees feels they are forced to give their salary to MPF fund manager.
Indeed, the reasons of low return may due to inactive management by employees.
In this example, we will explore artifical neural network (ANN) to rise the annualized return in MPF-ANN.

背景簡介

在香港，強積性公積金 (強積金, Mandatory Provident Fund, MPF) 是其中一種法定退休金。
僱主及部份僱員須扣除 5% 薪金供強積金營運商。
強積金營運商將每月收取管理費，並提供認可基金供僱員作投資儲蓄。
但是，強積金的平均回報僅 3.1%，因而被批評回報有三份之一被基金經理蠶食。
誠言，每位僱員如以主動方式管理強積金，有助提升回報而減少收費的影響。
這文記錄了作者以人工類神經網絡方式提升強積金回報的探索 (即 MPF-ANN)。

TORI used NMMA BIAS & Relative Strength Index (RSI)

The author used n-m MA BIAS indicator (NMMA BIAS) to extract trend and RSI for overbuy/oversell status in TORI.
This script combine two indicators and consider recent stock returns for the three outcome:
1. Tactical Asset Allocation (TAA) using TORI 2. Strategic Asset Allocation (SAA) using TORI and designated funds 3. Hedging mode (cash mode)

TORI 應用了時差平均線乖離率及相對強

作者嘗試以時差平均線乖離率 (n-m MA BIAS, NMMA BIAS) 及相對強弱指數 (RSI) 分配投資物。
時差平均線乖離率的預測考慮了投資物的趨勢 (Trend)。 相對強弱指數則考慮投資物的超買超賣狀態。
本程序結合兩項指標後，再考慮考慮最近三至六個月回報後，程式將自動決定以 TORI 佈置戰術性資產部署 (tactical asset allocation, TAA) 、或是以 TORI 結果配合債券及現金的策略性資產部署 (Strategic asset allocation)、或進入避險模式。

When to manage my portfolio using TORI?

The author collect historical pricing in the last day of every month to calculate NMMA BIAS and RSI in TORI.
The same day would be the day of successful rebalance.
In this example, MPF would have T+2 time delay for obtaining historical price, hence it is expected to have timing risks in TORI in MPF.

何時使用 TORI 管理及調配投資組合?

作者以每月最後一日取得投資物的歷史價格計算每月回報、NMMA BIAS 及相對強弱指數。
然後於同日重新配置投資物。
在本示例中，強積金大部分均有 T+2 的時延，因此實際操作將有時延誤差。

TORI in action: Walk-forwark validation results

TORI 前移式回測結果

單一供款 One-off MPF Installment:
年率化回報 Annualized Return: ~10.62%
算術年均回報 Mean Annual Return: ~10.17%
累積回報（等比）Cumulative Return (Geometric): ~531.79%
年率化標準誤差 Annualized Standard Deviation: ~8.60% (StdDev(monthly return) * sqrt(12))
夏普比率 Sharpe Ratio: 10.62%/8.60% = 1.2357
索丁諾比率 Sortino Ratio: 0.8732 (MAR = 0%)
預期損失 Expected Shortfall: 4.04% loss (0% Risk-free rate, 95% C.I.)

月供投資 Monthly MPF installment (或有誤差 maybe slightly differs):
每月供款 Monthly installment amount: 1500
總供款 Total contribution: 330000
最新結餘 Latest asset value: 904564.7
算術年均回報 Mean annual return: 9.61%
內部回報率 Internal Rate of Return (IRR): 9.53%
年率化標準誤差 Annualized Standard Deviation: 8.56%
夏普比率 Sharpe Ratio: 1.1221
索丁諾比率 Sortino Ratio: 0.7991 (MAR = 0%)

流程 Detailed Workflow

Package Preparation

1. Install necessary packages

```
r = getOption("repos")
r["CRAN"] = "https://cran.r-project.org/"
options(repos = r)


install.packages("zoo", type="binary")
install.packages("xts", type="binary")
install.packages("fBasics", type="binary")
install.packages("quantmod", dependencies = TRUE, type="binary")
install.packages("PerformanceAnalytics", dependencies = TRUE, type="binary")
install.packages("keras", type="binary")

install.packages("remotes")
remotes::install_github('rstudio/rmarkdown')
```

2. Now load necessary packages.

```
library("zoo")
```

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
library("xts")
```

```
## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo
```

```
library("fBasics")
```

```
## Loading required package: timeDate

## Loading required package: timeSeries

##
## Attaching package: 'timeSeries'

## The following object is masked from 'package:zoo':
##
##     time<-
```

```
library("quantmod")
```

```
## Loading required package: TTR
```

```
##
## Attaching package: 'TTR'
```

```
## The following object is masked from 'package:fBasics':
##
##     volatility
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
## Version 0.4-0 included new data defaults. See ?getSymbols.
```

```
library("PerformanceAnalytics")
```

```
##
## Attaching package: 'PerformanceAnalytics'
```

```
## The following objects are masked from 'package:timeDate':
##
##     kurtosis, skewness
```

```
## The following object is masked from 'package:graphics':
##
##     legend
```

Load Prices and Calculate Return

0. Parameters

```
# Determine the n-m MA BIAS percentage
nmBIAS_Percent <- 1

top <- 2
RSI_Overbuy <- 0.85
nm <- 1
MA_long <- 6
MA_Short <- 2
RSI_Period <- 18
Min_NMMA <- 0.00002
```

1. Load the price into zoo format

```
MPF.SunLife <-
  as.xts(
    read.zoo(
      "~/OneDrive/MPF/Sun Life Rainbow/SunLife.csv",
      format = "%d/%m/%Y",
      header = TRUE,
      read = read.csv,
      na.strings = "0"
    )
  )
daily <- index(MPF.SunLife)
```

2. Calculate Simple Moving Average (SMA)

```
MPF.SunLife.SMA.hy <- na.fill(apply(MPF.SunLife, 2, function(x) SMA(x, n=21*MA_long)), 0)
MPF.SunLife.SMA.hy <- as.xts(MPF.SunLife.SMA.hy, order.by = daily)
MPF.SunLife.SMA.mt <- na.fill(apply(MPF.SunLife, 2, function(x) SMA(x, n=21*MA_Short)), 0)
MPF.SunLife.SMA.mt <- as.xts(MPF.SunLife.SMA.mt, order.by = daily)
```

3. Calculate Relative Strength Index (RSI)

```
MPF.SunLife.RSI <- na.fill(apply(MPF.SunLife, 2, function(x) RSI(x, n=21*RSI_Period)),0)
MPF.SunLife.RSI <- as.xts(MPF.SunLife.RSI, order.by = daily)
```

4. Calculate Returns

```
MPF.SunLife.AE <- monthlyReturn(as.xts(MPF.SunLife$AE), type="log")
MPF.SunLife.B <- monthlyReturn(as.xts(MPF.SunLife$B), type="log")
MPF.SunLife.CA <- monthlyReturn(as.xts(MPF.SunLife$CA), type="log")
MPF.SunLife.CE <- monthlyReturn(as.xts(MPF.SunLife$CE), type="log")
MPF.SunLife.HKT <- monthlyReturn(as.xts(MPF.SunLife$FTSE.HK), type="log")
MPF.SunLife.G <- monthlyReturn(as.xts(MPF.SunLife$G), type="log")
MPF.SunLife.GB <- monthlyReturn(as.xts(MPF.SunLife$GB), type="log")
MPF.SunLife.GE <- monthlyReturn(as.xts(MPF.SunLife$GE), type="log")
MPF.SunLife.HKB <- monthlyReturn(as.xts(MPF.SunLife$HKB), type="log")
MPF.SunLife.HKE <- monthlyReturn(as.xts(MPF.SunLife$HKE), type="log")
MPF.SunLife.MPFC <- monthlyReturn(as.xts(MPF.SunLife$MPFC), type="log")
MPF.SunLife.SFP <- monthlyReturn(as.xts(MPF.SunLife$SFP), type="log")

MPF.SunLife.returns <- merge(MPF.SunLife.AE, MPF.SunLife.B, MPF.SunLife.CA, MPF.SunLife.CE, MPF.SunLife

MPF.SunLife.original.cost <- c(0.0201, 0.0187, 0.0085, 0.0206, 0.0102, 0.0183, 0.0202, 0.0204, 0.0180,
MPF.SunLife.cs.cost <- c(0.0096, 0.0089, 0.0095, 0.0096, 0.0096, 0.0089, 0.0089, 0.0096, 0.0089, 0.0089

for(col in 1:length(MPF.SunLife.returns[1,])) {
  MPF.SunLife.returns[,col] <- MPF.SunLife.returns[,col] - MPF.SunLife.cs.cost[col] + MPF.SunLife.origi
}

monthly <- index(MPF.SunLife.returns)
colnames(MPF.SunLife.returns) <- c("AE", "B", "CA", "CE", "FTSE HK", "G", "GB", "GE", "HKB", "HKE", "MP


rm(MPF.SunLife.AE, MPF.SunLife.B, MPF.SunLife.CA, MPF.SunLife.CE, MPF.SunLife.HKT, MPF.SunLife.G, MPF.Su
```

Calculate average RSI of the month, and then adjustment factor

```
MPF.SunLife.RSI.month <- as.xts(do.call(rbind, lapply(split(as.xts(MPF.SunLife.RSI), "months"), functio

MPF.SunLife.RSI.p <- MPF.SunLife.returns
MPF.SunLife.RSI.p[,] <- 0

for (col in 1:length(MPF.SunLife.RSI.month[1,])) {
  if (col != 11) {
    for (row in 1:length(MPF.SunLife.RSI.month[,col])) {
      percentile <- ecdf(as.numeric(MPF.SunLife.RSI.month[1:row,col]))
      if (percentile(MPF.SunLife.RSI.month[row,col]) >= (RSI_Overbuy - 1/length(1:row)^(1/2))) {
        MPF.SunLife.RSI.p[row,col] <- 0.2
      } else {
        MPF.SunLife.RSI.p[row,col] <- 1.2-(percentile(MPF.SunLife.RSI.month[row,col])^(3))
      }
    }
  } else {
    MPF.SunLife.RSI.p[,col] <- 1
  }
}

MPF.SunLife.RSI.sum <- as.xts(rowSums(MPF.SunLife.RSI.p), order.by = monthly)

for (row in 1:length(MPF.SunLife.RSI.p[,col])) {
  MPF.SunLife.RSI.p[row,] <- apply(MPF.SunLife.RSI.p[row,], 2, function(x) (x/MPF.SunLife.RSI.sum[row,1
}
MPF.SunLife.RSI.sum <- as.xts(rowSums(MPF.SunLife.RSI.p), order.by = monthly)
```

Allocate MPF for n-m MA BIAS

```
rm(MPF.SunLife.BIAS.diff, MPF.SunLife.BIAS.diff.qu, MPF.SunLife.BIAS.month, MPF.SunLife.BIAS.p)
MPF.SunLife.BIAS.diff <- (MPF.SunLife.SMA.mt - MPF.SunLife.SMA.hy) / MPF.SunLife.SMA.hy
MPF.SunLife.BIAS.diff.qu <- MPF.SunLife.BIAS.diff - lag(MPF.SunLife.BIAS.diff, k=21*nm)
MPF.SunLife.BIAS.diff.qu <- na.fill(MPF.SunLife.BIAS.diff.qu, 0)


MPF.SunLife.BIAS.month <- do.call(rbind, lapply(split(as.xts(MPF.SunLife.BIAS.diff.qu), "months"), func

MPF.portf.weight <- MPF.SunLife.returns
MPF.portf.weight[,] <- NA
#colnames(MPF.portf.weight) <- c("AP", "B", "CA", "CE", "EE", "G", "GB", "HSIT", "HKCE", "MPFC", "NA",

MPF.SunLife.stock.return <- as.xts(rowSums(MPF.SunLife.BIAS.month), order.by=monthly)
MPF.SunLife.stock.return[] <- NA

MPF.portf.return <- as.xts(rowSums(MPF.SunLife.BIAS.month), order.by=monthly)
MPF.portf.return[] <- NA

hedge <- FALSE
```

```r
round_percent <- function(x) {
  x <- x * 100
  result <- floor(x)    # Find integer bits
  remain <- x - result
  rsum <- sum(result)   # Find out how much we are missing
  i <- 1
  if (rsum < 100) {
    o <- order(remain, decreasing = TRUE)
    while (rsum < 100) {
      if (i > length(remain)) i <- 1
      idx <- o[i]
      if (result[idx] == 0) {
        i <- i+1
        next
      }
      result[idx] <- result[idx] + 1
      rsum <- sum(result)
      i <- i+1
    }
  }
  result <- result/100
  return(result)
}



MPF.SunLife.returns.mat <- as.matrix(MPF.SunLife.returns)
MPF.SunLife.stock.mean <- 0
for (row in 1:length(MPF.SunLife.BIAS.month[,1])) {
  MPF.SunLife.BIAS.month[row,] <- (MPF.SunLife.BIAS.month[row,]) * MPF.SunLife.RSI.p[row,]#* VIX.month[

  MPF.SunLife.stock.mean <- 0
  i <- 0
  for (col in 1:length(MPF.SunLife.BIAS.month[1,])) {
        if (col != 2 && col != 3 && col != 7 && col != 9 && col != 11 && col != 12) {
      if (!is.na(MPF.SunLife.returns.mat[row, col]) &&
          MPF.SunLife.returns.mat[row, col] != 0) {
        MPF.SunLife.stock.mean <- MPF.SunLife.stock.mean + MPF.SunLife.returns.mat[row, col]
          i <- i + 1
        }
        if (MPF.SunLife.BIAS.month[row, col] < 1e-6) {
          MPF.SunLife.BIAS.month[row, col] <- 0
        }
    } else {
      if (MPF.SunLife.BIAS.month[row, col] < 0) {
        MPF.SunLife.BIAS.month[row, col] <- 0
      }
    }

    # if (MPF.SunLife.BIAS.month[row,col] < Min_NMMA){
    #   MPF.SunLife.BIAS.month[row,col] <- 0
    # }
  }
```

```
MPF.SunLife.stock.return[row] <- MPF.SunLife.stock.mean / i

if (is.nan(MPF.SunLife.BIAS.month[row,col])) {
  MPF.SunLife.BIAS.month[row,col] <- 0
}

# Retain two most increasing fund
last <- length(MPF.SunLife.BIAS.month[1,]) - top

order <- order(MPF.SunLife.BIAS.month[row,])[1:last]
for (col in order) {
  MPF.SunLife.BIAS.month[row,col] <- 0
}

MPF.SunLife.sum <- sum(MPF.SunLife.BIAS.month[row,])
MPF.SunLife.BIAS.p <- MPF.SunLife.BIAS.month[row,]
MPF.SunLife.BIAS.p[] <- 0


if (row > 8 && MPF.SunLife.stock.return[row] <
    quantile(na.omit(MPF.SunLife.stock.return), c(.35)) &&
    MPF.SunLife.stock.return[row - 3] <
    quantile(na.omit(MPF.SunLife.stock.return), c(.45))) {
  up <- FALSE
}

if (row > 8 && hedge &&
    MPF.SunLife.stock.return[row] >
      quantile(na.omit(MPF.SunLife.stock.return), c(.45))  &&
    MPF.SunLife.stock.return[row - 3] >
      quantile(na.omit(MPF.SunLife.stock.return), c(.35))) {
  hedge <- FALSE
  up <- TRUE
}

if (row > 8 &&  (MPF.SunLife.stock.return[row] < 0 &&
                 MPF.SunLife.stock.return[row-1] >
                 quantile(na.omit(MPF.SunLife.stock.return), c(.85)))) {
  hedge <- TRUE
}

MPF.SunLife.sum <- sum(MPF.SunLife.BIAS.month[row, ])
MPF.SunLife.BIAS.p[] <- 0


if (row <= 12 || MPF.SunLife.sum == MPF.SunLife.BIAS.month[row,11] ||
    MPF.SunLife.sum < 1e-6 || hedge == TRUE) {
  if (row >= 102) {
    MPF.SunLife.BIAS.p[7] <- 0.3
    MPF.SunLife.BIAS.p[11] <- 0.7
  } else {
    MPF.SunLife.BIAS.p[11] <- 1
  }
```

```
  } else if (min(MPF.SunLife.stock.return[(row-3):row]) < -0.08) {
    if (row >= 102) {
      MPF.SunLife.BIAS.p[] <- MPF.SunLife.BIAS.month[row, ] / MPF.SunLife.sum / 10 * 7
      MPF.SunLife.BIAS.p[11] <- MPF.SunLife.BIAS.p[11]+0.18
      MPF.SunLife.BIAS.p[7] <- MPF.SunLife.BIAS.p[7]+0.12
    } else {
      MPF.SunLife.BIAS.p[ ] <- MPF.SunLife.BIAS.month[row, ] / MPF.SunLife.sum / 10 * 7
      MPF.SunLife.BIAS.p[11] <- MPF.SunLife.BIAS.p[11]+0.3
    }
  } else {
    MPF.SunLife.BIAS.p[ ] <- MPF.SunLife.BIAS.month[row, ] / MPF.SunLife.sum
  }


  MPF.portf.weight[row,] <-
    round_percent(MPF.SunLife.BIAS.p)

  portf.rebal.fm <-
    Return.portfolio(
      na.fill(MPF.SunLife.returns, 0),
      weight = MPF.portf.weight[1:row,],
      geometric = TRUE,
      rebalance_on = "months"
    )

  MPF.portf.return[row] <-
    tail(na.omit(portf.rebal.fm), 1)
  MPF.portf.drawdown <- Drawdowns(MPF.portf.return,
                                  geometric = TRUE)

  if (tail(na.omit(MPF.portf.drawdown), 1) < -0.065 && up == FALSE) {
    hedge = TRUE
  }
}
```

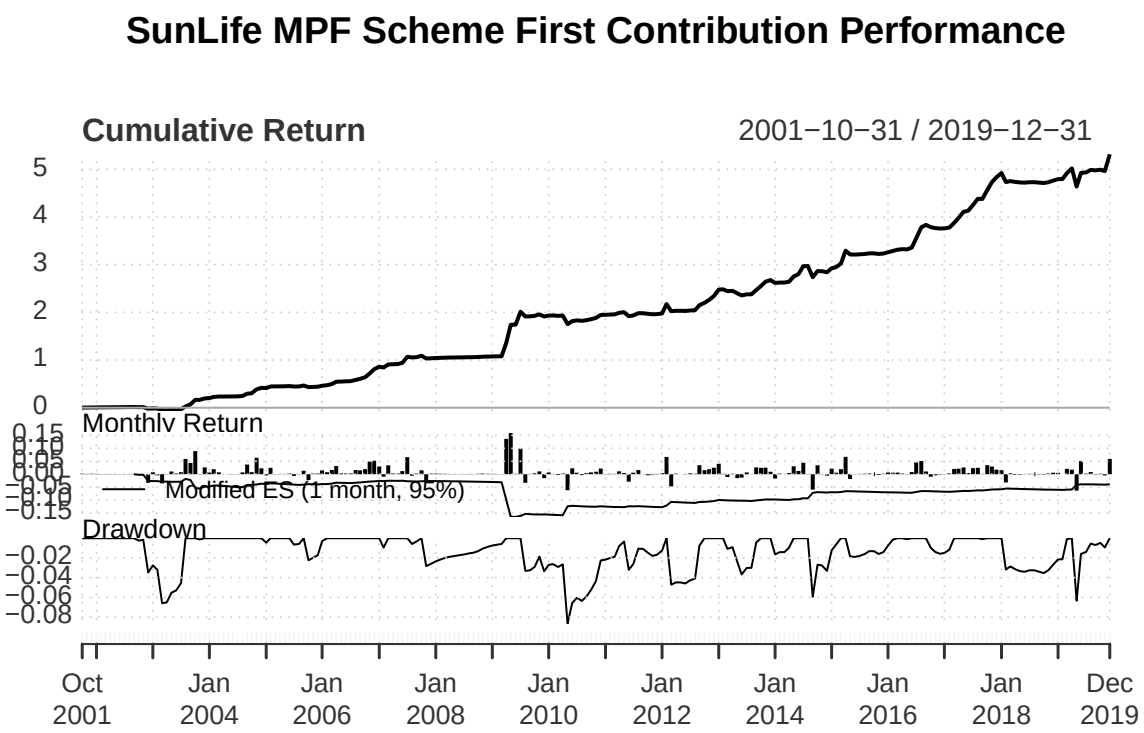
Performance Analysis

```
portf.rebal.fm <- Return.portfolio(
  MPF.SunLife.returns,
  weight = MPF.portf.weight,
  geometric = TRUE,
  rebalance_on = "months"
)

mean.annual.return <-
  mean(do.call(rbind, lapply(split(portf.rebal.fm, "years"), function(x)
    colMeans(x))) * 12)

charts.PerformanceSummary(
  portf.rebal.fm,
  methods = "ModifiedES",
  geometric = TRUE,
```

```
  p = .95,
  main = "SunLife MPF Scheme First Contribution Performance"
)
```




```
Return.cumulative(portf.rebal.fm, geometric=TRUE)
```

```
##                   portfolio.returns
## Cumulative Return          5.317949
```

```
portf.rebal.fm.sharpe <- Return.annualized(portf.rebal.fm, geometric=TRUE) / (StdDev.annualized(portf.r
rownames(portf.rebal.fm.sharpe) <- "Sharpe Ratio"
Return.annualized(portf.rebal.fm, geometric=TRUE)
```

```
##                   portfolio.returns
## Annualized Return         0.1062854
```

```
mean.annual.return
```

```
## [1] 0.1017046
```

```
portf.rebal.fm.sharpe
```

```
##              portfolio.returns
## Sharpe Ratio           1.23568
```

```
portf.rebal.fm.sharpe.mean <- mean.annual.return / (StdDev.annualized(portf.rebal.fm))
rownames(portf.rebal.fm.sharpe.mean) <- "Sharpe Ratio (Mean annual return)"
portf.rebal.fm.sharpe.mean
```

```
##                                   portfolio.returns
## Sharpe Ratio (Mean annual return)          1.182424
```

```
StdDev.annualized(portf.rebal.fm)
```

```
##                                portfolio.returns
## Annualized Standard Deviation        0.08601368
```

```
SortinoRatio(portf.rebal.fm)
```

```
##                           portfolio.returns
## Sortino Ratio (MAR = 0%)         0.8731692
```

```
ES(portf.rebal.fm, method="historical")
```

```
##    portfolio.returns
## ES       -0.04040675
```

```
tail(MPF.portf.weight, n=6)
```

```
##              AE B CA   CE FTSE HK G   GB GE HKB  HKE MPFC SFP
## 2019-07-31 0.00 0  0 0.00       0 0 0.30  0   0 0.00 0.70   0
## 2019-08-30 0.00 0  0 0.00       0 0 0.30  0   0 0.00 0.70   0
## 2019-09-30 0.00 0  0 0.00       0 0 0.94  0   0 0.00 0.06   0
## 2019-10-31 0.48 0  0 0.52       0 0 0.00  0   0 0.00 0.00   0
## 2019-11-29 0.51 0  0 0.49       0 0 0.00  0   0 0.00 0.00   0
## 2019-12-31 0.55 0  0 0.00       0 0 0.00  0   0 0.45 0.00   0
```

Monthly Installment

```
MPF.SunLife.units <- MPF.SunLife.returns
MPF.SunLife.units[, ] <- 0

MPF.monthly.asset <- MPF.SunLife.returns
MPF.monthly.asset[, ] <- 0

MPF.monthly.returns <-
  as.xts(rowSums(MPF.SunLife.returns), order.by = monthly)
MPF.monthly.returns[] <- 0

MPF.time <- 0:length(MPF.SunLife.returns[, 1]) / 12
MPF.pay <- -1500 + 0 * MPF.time
```

```
for (row in 1:length(MPF.SunLife.returns[, 1])) {
#for (row in 1:13) {
  this.price <- as.matrix(MPF.SunLife[monthly[row]])
  MPF.SunLife.units[row, ] <- this.price

  if (row == 1) {
    last.value <- 1500
    this.value <- last.value * this.price[11]
    MPF.monthly.returns[row] <- log(sum(na.fill(this.value,0)) / 1500)
    MPF.monthly.asset[row,] <-
      na.fill(((this.value + 1500) / this.price * MPF.portf.weight[row, ]) / this.price,0)
    last.price <- this.price
  } else {
    #last.value <-
    #  as.numeric(sum(na.fill(last.price * MPF.monthly.asset[row - 1, ], 0)))
    #this.value <-
    #  as.numeric(sum(na.fill(this.price * MPF.monthly.asset[row - 1, ], 0)))
    #MPF.monthly.returns[row] <- log(this.value / last.value)

    last.value <- na.fill(MPF.monthly.asset[row-1,] * last.price,0)
    this.value <- na.fill(MPF.monthly.asset[row-1,] * this.price,0)
    MPF.monthly.returns[row] <- log(sum(this.value) / sum(last.value))

    MPF.monthly.asset.old <- this.value / sum(na.fill(this.value,0))
    MPF.monthly.asset.propose <- as.matrix(MPF.portf.weight[row,]) -
       as.matrix(MPF.monthly.asset.old)
    MPF.monthly.unit.change <- MPF.monthly.asset.propose * sum(na.fill(this.value,0)) /this.price

    this.unit <- as.matrix(MPF.monthly.asset[row-1,]) + as.matrix(MPF.monthly.unit.change)
    this.unit <- as.matrix(this.unit) + (1500 / as.matrix(this.price) * MPF.portf.weight[row,])

    MPF.monthly.asset[row,] <- na.fill(this.unit,0)
    last.price <- this.price
  }
}


total.asset.value <- sum(MPF.monthly.asset[row, ] * this.price)
total.contribution <- 1500 * length(MPF.SunLife.returns[, 1])

MPF.pay[row + 1] <- total.asset.value
IRR.f <- function(r)
  sum(MPF.pay * exp(-r * MPF.time))
IRR.root <- uniroot(IRR.f, c(0, 1))


total.asset.value
```

```
## [1] 904564.7
```

```
total.contribution
```

```
## [1] 330000
```

```r
mean.monthly.annual.return <-
  mean(do.call(rbind, lapply(split(MPF.monthly.returns, "years"), function(x)
    colMeans(x))) * 12)
mean.monthly.annual.return
```

```
## [1] 0.09696108
```

```r
IRR.root$root
```

```
## [1] 0.09584039
```

```r
stddev.monthly <- (StdDev(MPF.monthly.returns) * sqrt(12))
monthly.installment.sharpe.ratio <-
  mean.monthly.annual.return / stddev.monthly
rownames(monthly.installment.sharpe.ratio) <-
  "Sharpe Ratio (mean annual return)"

StdDev.annualized(MPF.monthly.returns)
```

```
##                                      [,1]
## Annualized Standard Deviation 0.08597236
```

```r
monthly.installment.sharpe.ratio
```

```
##                                        [,1]
## Sharpe Ratio (mean annual return) 1.127817
```

```r
SortinoRatio(MPF.monthly.returns)
```

```
##                               [,1]
## Sortino Ratio (MAR = 0%) 0.795728
```

```r
ES(MPF.monthly.returns, method = "historical")
```

```
##           [,1]
## ES -0.04094886
```