Reproducible Research in Portfolio Selection

by Majeed Simaan

Abstract This article contributes to reproducible research in portfolio selection using R. In particular, we evaluate several portfolio rules used in the literature by utilizing different publicly available data sets. The primary analysis builds on a recent empirical design by Kan et al. (2022) (KWZ), which represents one of the standard frameworks pursued in the literature. Our empirical investigation reproduces 198 performance metrics reported by the original research, covering six public data sets, eleven portfolio rules, and three performance metrics. Additionally, our experiment can be generalized to accommodate additional data sets and portfolio rules. This generalization, however, presumes that the user is willing to put extra effort to (1) add new data sets to the main `data_list` object using the same format and (2) follow the logic behind the primary decision rule function (`DR_function`) to incorporate other portfolio rules with the same vector structure.

Introduction

The portfolio selection problem is typically viewed as a two steps problem. The first step concerns how the decision-maker (henceforth DM) assimilates information, e.g., evaluates underlying assets. The second step presumes that the DM has already acquired such prior/posterior beliefs about the assets in order to make an optimal portfolio choice. The seminal work by Markowitz (1952) focuses on the second step regarding how the DM can form optimal choices based on given preferences and beliefs. This paradigm is known today as the very famous mean-variance (MV) model. Despite its normative appeal, the model suffers from what is known as estimation risk (Simaan et al., 2018). Estimation risk corresponds to the impact of the first step in the portfolio decision process (ex-ante forming beliefs) on the second step (ex-post optimality).

Research on estimation risk in portfolio selection is not new and dates at least back to Klein and Bawa (1976). Nonetheless, the literature on this area is enormous. A quick search on Google Scholar for the exact terms "estimation risk" and "portfolio selection" corresponds to 2,650 results for the years ranging between 2000 and 2020. The corresponding number for the years ranging between 1950 and 1999 is 384. Not surprisingly, the increased popularity of this area can be attributed to many factors, one of which is computational and technological advancements, such as improved optimization algorithms, robust statistical tools, and open-source statistical/machine learning models.

In this article, we focus on one of the prominent frameworks in the literature by Kan et al. (2022). Given how competitive the literature on "estimation risk in portfolio selection" is, it is common for researchers to compare their proposed new decision rules with other existing benchmarks (see, e.g., Kan et al. (2022); Lassance et al. (2022)). For instance, the empirical design by Kan et al. (2022) considers more than a dozen of benchmarks to evaluate the effectiveness/robustness of their proposed decision rule. Motivated by competitive literature, we consider a similar experiment as in Kan et al. (2022) (henceforth KWZ) and replicate a subset of their results, focusing on six public data sets and eleven portfolio rules.

Specifically, the analysis pursued in this article replicates 198 statistics reported by KWZ, covering three performance metrics (certainty equivalent returns, Sharpe ratio, and portfolio turnover), six data sets, and eleven decision rules. Most of the decision rules studied in our article rely on a closed-form solution, whereas two require numerical optimization. We note that most of the results can be easily replicated to a large extent. At the same time, we witness a minor divergence between our results and KWZ. Such deviations could potentially be attributed to differences in software (especially for numerical optimization) as well as other statistical packages for shrinkage and potential changes in the publicly available data (see, e.g., Akey et al. (2022)),

We hope that the initiative undertaken in our research will make it more accessible and user-friendly for other/future researchers in pursuing their optimal portfolio. The rest of the article proceeds as follows. In Section 2.2, we briefly discuss the libraries used in the implementation. Section 2.3 illustrates the process of gathering publicly available data, which comes from two different sources. Section 2.4 discusses the implementation of eleven decision rules used by KWZ. The main empirical analysis is conducted using backtesting and illustrates the replication of the original results from KWZ, which we cover in Section 2.5. Finally, Section 2.6 concludes.

Libraries

In terms of implementation, we utilize seven libraries in total:

```
library(xts) # for time series manipulation
library(lubridate) # for date manipulation
library(parallel) # for parallel processing across data sets
library(readxl) # reading excel files to load data
library(ggplot2) # for visualization
library(zipfR) # needed for one of the decision rules
library(RiskPortfolios) # used for covariance matrix shrinkage

rm(list = ls())
```

The **xts** library is used for downloading data and working with time series, whereas the **lubridate** library is used for date manipulation and formatting. The `parallel` library can be efficiently utilized on Linux machines to perform parallel computing using functional programming, e.g., working with `lapply`. We refer to the **readxl** library to read xlsx formatted files, which is relevant to two of six data sets studied in this article. The **ggplot2** library is used for results visualization and comparison. The last two libraries are used in the implementation of the decision rules. Specifically, we utilize the **zipfR** library to implement the incomplete beta function as advised by KWZ and employ the **RiskPortfolios** library for linear shrinkage of covariance matrices.

Getting the Data

In their paper, KWZ rely on eight different data sources to evaluate the different decision rules. Four of these sets come from Ken French's website (commonly referred to as Fama-French), and two come from Robert Novy-Marx's website (Novy-Marx and Velikov, 2016), whereas the remaining are constructed from the CRSP database. In this article, we focus on the first six data sets which are publicly available:

1. 10 momentum portfolios (Momentum); January 1927 to December 2018
2. 25 Fama-French 5 $\times$ 5 size and book-to-market ranked portfolios (Size-B/M); from January 1927 to December 2018
3. 25 portfolios formed on operating profitability and investment (OP-Inv); from July 1963 to December 2018
4. 49 industry portfolios (Industry); from July 1969 to December 2018
5. 16 low turnover anomalies by Novy-Marx and Velikov (2016); from July 1963 to December 2013
6. 46 all turnover anomalies by Novy-Marx and Velikov (2016); from July 1973 to December 2013 (after dropping missing values)

Fama-French Data

With some minimal tweaking, we can download the four Fama-French data in the following manner:

```
# MOMETUM -  January 1927 to December 2018
FF_file <- "https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/ftp/10_Portfolios_Prior_12_2_CSV.zip"
temp <- tempfile()
download.file(FF_file,temp)
unz_files <- unzip(temp)
ds <- read.csv(unz_files,skip = 10,stringsAsFactors = F)
flag_obs <- grep("Average Equal",ds[,1],ignore.case = T)
ds <- ds[1:(flag_obs-1),]
ds <- data.frame(apply(ds,2,as.numeric))
names(ds)[1] <- "date"
ds$date <- ds$date*100  + 1
ds$date <- ymd(ds$date)
ds$date <- ceiling_date(ds$date,"m") - 1
ds[,-1] <- ds[,-1]/100
ds <- ds[ds$date >= "1927-01-01",]
ds <- ds[ds$date <= "2018-12-31",]
ds1 <- ds
```

```
# 25BM - January 1927 to December 2018
FF_file <- "https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/ftp/25_Portfolios_5x5_CSV.zip"
temp <- tempfile()
download.file(FF_file,temp)
unz_files <- unzip(temp)
ds <- read.csv(unz_files,skip = 15,stringsAsFactors = F)
flag_obs <- grep("Average Equal",ds[,1],ignore.case = T)
ds <- ds[1:(flag_obs-1),]
ds <- data.frame(apply(ds,2,as.numeric))
names(ds)[1] <- "date"
ds$date <- ds$date*100  + 1
ds$date <- ymd(ds$date)
ds$date <- ceiling_date(ds$date,"m") - 1
ds[,-1] <- ds[,-1]/100
ds <- ds[ds$date >= "1927-01-01",]
ds <- ds[ds$date <= "2018-12-31",]
ds2 <- ds

# 25OP-Inv July 1963 to December 2018
FF_file <- "https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/ftp/25_Portfolios_OP_INV_5x5_CSV.zip"
temp <- tempfile()
download.file(FF_file,temp)
unz_files <- unzip(temp)
ds <- read.csv(unz_files,skip = 24,stringsAsFactors = F)
flag_obs <- grep("Average Equal",ds[,1],ignore.case = T)
ds <- ds[1:(flag_obs-1),]
ds <- data.frame(apply(ds,2,as.numeric))
names(ds)[1] <- "date"
ds$date <- ds$date*100  + 1
ds$date <- ymd(ds$date)
ds$date <- ceiling_date(ds$date,"m") - 1
ds[,-1] <- ds[,-1]/100
ds <- ds[ds$date >= "1963-07-01",]
ds <- ds[ds$date <= "2018-12-31",]
ds3 <- ds

# 49 Industry -  July 1969 to December 2018
FF_file <- "https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/ftp/49_Industry_Portfolios_CSV.zip"
temp <- tempfile()
download.file(FF_file,temp)
unz_files <- unzip(temp)
ds <- read.csv(unz_files,skip = 11,stringsAsFactors = F)
flag_obs <- grep("Average Equal",ds[,1],ignore.case = T)
ds <- ds[1:(flag_obs-1),]
ds <- data.frame(apply(ds,2,as.numeric))
names(ds)[1] <- "date"
ds$date <- ds$date*100  + 1
ds$date <- ymd(ds$date)
ds$date <- ceiling_date(ds$date,"m") - 1
ds[,-1] <- ds[,-1]/100
ds <- ds[ds$date >= "1969-07-01",]
ds <- ds[ds$date <= "2018-12-31",]
ds4 <- ds
```

NMV Data

The other two data sets are referred to as NMV (abbreviation for Novy-Marx and Velikov (2016)). The data can be downloaded from Robert Novy-Marx's website. By design, the data covers different decile portfolios that are formed based on different stock characteristics. Similar to KWZ, the first data set focuses on the low turnover anomaly strategies, which correspond to

```
low_to_strat <- c("Size","Gross Profitability",
                  "Value","ValProf",
                  "Accruals","Asset Growth",
                  "Investment","Piotroski's F-score")

low_to_strat
```

```
#> [1] "Size"                "Gross Profitability" "Value"
#> [4] "ValProf"             "Accruals"            "Asset Growth"
#> [7] "Investment"          "Piotroski's F-score"
```

The data can be downloaded and prepared in the following way:

```
file.i <- "http://rnm.simon.rochester.edu/data_lib/ToAatTC/Simple_Strategies_Returns.xlsx"
temp <- tempfile()
download.file(file.i,temp)
ds_sheets_all <- sort(excel_sheets(temp))
ds_sheets <- ds_sheets_all[ds_sheets_all %in% low_to_strat]
NM_data_list <- list()
for (sheet_i in ds_sheets) {
  ds_i <- data.frame(read_xlsx(temp,sheet = sheet_i))
  ds_i <- ds_i[,c(1,2,ncol(ds_i))]
  names(ds_i)[-1] <- paste(sheet_i,names(ds_i)[-1],sep = "_")
  ds_i$date <- ds_i$Month
  ds_i$date <- ymd(ds_i$date*100 + 1)
  ds_i$date <- ceiling_date(ds_i$date,"m") - 1
  ds_i$Month <- NULL
  ds_i[,1:2] <- ds_i[,1:2]/100
  NM_data_list <- c(NM_data_list,list(ds_i))
}
NM_data_LT <- Reduce(merge,NM_data_list)
NM_data_LT <- na.omit(NM_data_LT)
```

We note that even though we consider eight characteristics, we have 16 portfolios. Similar to KWZ, we include both the long (top decile) and short (bottom decile) portfolios resulting in 606 monthly observations with 17 columns (16 columns for long and short portfolios, and 1 (first) column for the date):

```
dim(NM_data_LT)
```

```
#> [1] 606  17
```

Additionally, KWZ consider all 23 anomalies corresponding to low, medium, and high turnover strategies. Same as above, we consider the top and bottom deciles, resulting in 46 portfolios/assets:

```
mid_to_strat <- c("Net Issuance (rebal.-A)","Return-on-book equity","Failure Probability",
                  "ValMomProf","ValMom","Idiosyncratic Volatility",
                  "Momentum","PEAD (SUE)","PEAD (CAR3)")
high_to_start <- c("Industry Momentum","Industry Relative Reversals","High-frequency Combo",
                   "Short-run Reversals","Seasonality","IRR (Low Vol)")
all_to_start <- unique(sort(c(low_to_strat,mid_to_strat,high_to_start)))
length(all_to_start)
```

```
#> [1] 23
```

Let us repeat the same loop as before to create long and short portfolios:

```
ds_sheets <- ds_sheets_all[ds_sheets_all %in% all_to_start]
NM_data_list <- list()
for (sheet_i in ds_sheets) {
  ds_i <- data.frame(read_xlsx(temp,sheet = sheet_i))
  ds_i <- ds_i[,c(1,2,ncol(ds_i))]
  names(ds_i)[-1] <- paste(sheet_i,names(ds_i)[-1],sep = "_")
  ds_i$date <- ds_i$Month
  ds_i$date <- ymd(ds_i$date*100 + 1)
  ds_i$date <- ceiling_date(ds_i$date,"m") - 1
  ds_i$Month <- NULL
  ds_i[,1:2] <- ds_i[,1:2]/100
  NM_data_list <- c(NM_data_list,list(ds_i))
}
```

```
NM_data_ALL <- Reduce(merge,NM_data_list)
NM_data_ALL <- na.omit(NM_data_ALL)
dim(NM_data_ALL)
```

```
#> [1] 486  47
```

Data List

Finally, we load the six data sets in a single list, so we can easily generalize the analysis for each set:

```
ds_list <- list(ds1,ds2,ds3,ds4,NM_data_LT,NM_data_ALL)
names(ds_list) <- c("MOM10","BM25","OPIN25","IND49","NMV16","NMV46")
sapply(ds_list,nrow)
```

```
#>  MOM10   BM25 OPIN25  IND49  NMV16  NMV46
#>   1104   1104    666    594    606    486
```

Note that such a generalization should run smoothly as long as each set follows the same format, even though each data set has a different number of observations/columns. Nonetheless, it may be preferable to ensure all sets have the exact dates or cover the same period for consistent comparison. For instance, this could help the researcher understand why specific decision rules work for one data set but not another. Nonetheless, in this article, we follow the original analysis by KWZ for the objective of reproducible research.

As a final tweak, we sort the order of the data sets in line with the one reported by KWZ, which is as follows:

```
ds_list <- ds_list[c("MOM10","BM25","OPIN25","NMV16","NMV46","IND49")]
sapply(ds_list,ncol) - 1
```

```
#>  MOM10   BM25 OPIN25  NMV16  NMV46  IND49
#>     10     25     25     16     46     49
```

The only two data sets missing in our analysis are the IVOL ($N = 10$) and Stocks ($N = 100$), where N denotes the number of assets. These two data sets are user-based, such that the replication exercise is subjected to additional degrees of freedom depending on the researcher's ability to construct the data. In contrast, the above six data sets are publicly available and require minimal preparation for reproducible research.

Excess Returns

Before we move to the main analysis, note that KWZ consider the excess returns on all portfolios. To adjust the asset returns accordingly, we download the risk-free rate, represented by the one-month Treasury bill rate, from the Fama-French three factors data:

```
FF_file <- "https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/ftp/F-F_Research_Data_Factors_CSV.zip"
temp <- tempfile()
download.file(FF_file,temp)
unz_files <- unzip(temp)
ds <- read.csv(unz_files,skip = 3)
flag_obs <- grep("Annual",ds[,1],ignore.case = T)
ds <- ds[1:(flag_obs-1),]
names(ds)[1] <- "date"
ds <- data.frame(apply(ds, 2, as.numeric))
ds$date <- ceiling_date(ymd(ds$date*100+ 01),"m")-1
ds <- ds[,c("date","RF")]
ds$RF <- ds$RF/100
ds_rf <- ds
rm(ds)
```

Now that we have the risk-free rate let us merge it with the data list and adjust the returns accordingly using the following function:

```
adjust_rf <- function(ds_i) {
  ds_i <- merge(ds_i,ds_rf, by = "date")
  RF <- ds_i$RF
  ds_i[,-1] <- ds_i[,-1] - RF
  ds_i$RF <- NULL
  return(ds_i)
}
ds_list <- lapply(ds_list,adjust_rf)
```

The object `ds_list` contains the excess returns on six data sets. Users, who are interested in considering additional data sets, should be able to efficiently utilize the above commands and amend additional data sets into the `ds_list` list. However, this presumes that the additional data set follows the same format.

Decision Rules

Now that we covered the six main data sets, we move to discuss the portfolio rules used in our investigation. We consider eleven decision rules in total. Without any position constraints, most decision rules studied by KWZ can be implemented using closed-form solutions that rely on matrix algebra and basic statistics. However, in the case of short-sales constraints, we need to solve for optimal portfolios numerically.[1] To do so, we refer to the following codes/functions:

```
U <- function(X,Mu,Sigma,gamma) {
  u1 <- t(X)%*%(Mu)
  u2 <- t(X)%*%Sigma%*%X
  total <- u1 - (gamma/2)*u2
  return(c(total))
}

G <- function(X,Mu,Sigma,gamma) {
  total <-  Mu - gamma*Sigma%*%X
  return(total)
}

U2 <- function(X,Sigma) {
  u2 <- t(X)%*%Sigma%*%X
  return(c(u2))
}

G2 <- function(X,Sigma) {
  g2 <- Sigma%*%X
  total <- g2
  return(total)
}

# add constraints
BC_f <- function(d) {
  # sum to one constraint
  A <- matrix(1,1,d)
  A <- rbind(A,-A)
  B <- c(0.99999,-1.00001)

  # short-sales constraints
  A2 <- diag(rep(1,d))
  B2 <- rep(0,d)
  A2 <- rbind(A,A2)
  B2 <- c(B,B2)

  # stack altogether in a list
  BC1 <- list(A,B)
  BC2 <- list(A2,B2)
  list(BC1,BC2)
}
```

A couple of comments are in order. First, note that `U` denotes the objective function that captures the mean-variance preference, whereas `gamma` denotes the risk aversion level. To result in faster convergence, we define the gradient function, denoted by `G`. Hence, for a given mean vector and

[1]For further details about numerical portfolio optimization see this link relevant, for instance.

covariance matrix, we can train the algorithm to seek the optimal point using gradient descent, e.g., Newton's method. The `U2` and `G2` functions follow suit, whereas the main difference is that the objective is to minimize the portfolio variance rather than optimize a mean-variance trade-off.

Second, the `BC_f` function is written to impose constraints on the optimization problem. By design, the base function `constrOptim` takes constraints in a linear form, such that it can be written as

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{b} \end{aligned}$$

In this regard, the `BC_f` function returns the relevant $\mathbf{A}$ and $\mathbf{b}$ matrices to reflect two constraints:

$$\mathbf{1}^\top \mathbf{x} = 1 \text{ and } x_i \geq 0, \forall i = 1, ..., d,$$

which correspond to the budget and the no short-sales constraints. For instance, the naive portfolio (i.e., the equally weighted) confirms both:

```
d <- 10
A <- BC_f(d)[[2]][[1]]
b <- BC_f(d)[[2]][[2]]
x <- rep(1/d,d)
all(A%*%x >= b)

#> [1] TRUE
```

Therefore, we can easily include the `A` and `b` matrices into the optimization function to ensure long-only (no negative) portfolio positions while satisfying the budget that all available capital is allocated among the underlying assets.

Before we proceed to the main function covering the decision rules, it is worth noting that any decision rule can be viewed as a mapping function of the data into portfolio weights, i.e., $f : \mathcal{D} \to \mathbf{w}$, where $\mathcal{D}$ denotes the data along with other inputs, f is the mapping function (decision rule), and $\mathbf{w}$ is the corresponding portfolio weights. For instance, in the case of the mean-variance portfolio, the data is mapped into the mean vector and covariance matrix using a sample size of T, which are eventually used as the primary inputs to determine the optimal portfolio weights for given a risk aversion level. Hence, to implement different decision rules, we need to write a general function that takes a data subset `R_sub` with different inputs and returns the portfolio weights. Based on the outcome, we can evaluate different decision rules over time, corresponding to a backtesting procedure. In the end, such a function maps the data and other preferences into future allocations.

We define such a mapping function as follows:

```
DR_function <- function(R_sub,gamma,sample_size,TC) {

  # takes data  object R_sub - see below
  R_sub2 <- tail(R_sub,sample_size)
  R_sub2 <- R_sub2[,-1]
  d <- ncol(R_sub2)
  Mu <- apply(R_sub2,2,mean)
  Sigma <- var(R_sub2)*(d-1)/d # for MLE estimate divide by d
  Sig_inv <- solve(Sigma)
  e <- rep(1,d)

  # first portfolio is the global minimum variance portfolio. KWZ denote it by $\hat{w}_{g,t}$
  GMV <- Sig_inv%*%e/sum(Sig_inv)
  B_mat <- Sig_inv%*%( diag(d) - e%*%t(GMV)  )
  # the second is the plug-in portfolio. KWZ denote it by $\hat{w}_{z,t}$
  MV <- GMV + (1/gamma)*B_mat%*%Mu

  # the unbiased estimate
  MV_UB <- GMV + (sample_size - d - 1)/(gamma*sample_size)*B_mat%*%Mu

  # Naive Portfolio
  Naive <- rep(1/d,d)

  BC <- BC_f(d)[[2]] # 2 for no short-sales and 1 for yes
  A <- BC[[1]]
  B <- BC[[2]]
  # initial guess is based on naive portfolio
  X0 <- Naive
  X_opt <- constrOptim(X0,
                       function(x) -U(x,Mu,Sigma,gamma),
                       grad = function(x) -G(x,Mu,Sigma,gamma),
                       ui = A,ci = B)
  X1 <- X_opt$par
  MV_NS <- X1/sum(X1)

  X_opt <- constrOptim(X0,function(x) U2(x,Sigma),
                       grad = function(x) G2(x,Sigma),
                       ui = A,ci = B)
  X1 <- X_opt$par
  GMV_NS <- X1/sum(X1)

  # Volatility Timing: the first strategy proposed by Kirby and Ostdiek (2012)
  KO_VT <- (1/diag(Sigma))^4 # footnote in Table 2 states that eta = 4
  KO_VT <- KO_VT/sum(KO_VT)

  # Reward-to-Risk Timing: The second strategy proposed by Kirby and Ostdiek (2012)
  Mu_plus <- Mu
  Mu_plus[Mu_plus < 0] <- 0
  KO_RT <- (Mu_plus/diag(Sigma))^4 # footnote in Table 2 states that eta = 4
  KO_RT <- KO_RT/sum(KO_RT)

  # add KWZ decision rule
  N <- d
  T <- sample_size
  # the proposed way to estimate
  psi2hat <- t(MV) %*% Mu;
  psi2hat <- ((T-N-1)*psi2hat-(N-1))/T +
    ((2*(psi2hat)^((N-1)/2)*((1+psi2hat)^(-(T-2)/2)))/(T*Ibeta(psi2hat/(1+psi2hat),(N-1)/2,(T-N+1)/2)));
  psi2hat <- as.numeric(psi2hat)

  kappaE <- (((T-N)*(T-N-3))/(T*(T-2)))*(psi2hat/(psi2hat+(N-1)/T));
  KWZ <-  GMV*(1-kappaE) + MV*(kappaE)

  # Ledoit and Wolf (2004)
  m_i <- list(type = "diag")
  Sigma <- covEstimation(as.matrix(R_sub2),control = m_i)
  Sig_inv <- solve(Sigma)
  e <- rep(1,d)

  # update the GMV based on the cov estimate
  GMV_LW_2004 <- Sig_inv%*%e/sum(Sig_inv)
  B_mat <- Sig_inv%*%( diag(d) - e%*%t(GMV_LW_2004))
  MV_LW_2004 <- GMV_LW_2004 + (1/gamma)*B_mat%*%Mu

  # do the same for the KWZ decision rule
  KWZ_LW_2004 <-  GMV_LW_2004*(1-kappaE) + MV_LW_2004*(kappaE)

  # organize in a similar order as in KWZ
  list(KWZ = KWZ, MV = MV, MV_UB = MV_UB,
       KWZ_LW_2004 = KWZ_LW_2004, MV_LW_2004 = MV_LW_2004,
       MV_NS = MV_NS,
       GMV = GMV,GMV_NS = GMV_NS,
       Naive = Naive,
       KO_VT = KO_VT,KO_RT = KO_RT)
}
```

For brevity, we skip the discussion of the above decision rules. However, note that the function returns eleven decision rules in a list. As a summary, Table 1 provides the mathematical definition of each rule from `DR_function` in line with the notation used by KWZ :

Table 1: This table reports the decision rules along with their definitions. The first column corresponds to the name of the decision rule used in the `DR_function` function, whereas the second and third columns denote the definition and the mathematical representation used in the original paper by Kan et al. (2022).

R Notation	Definition	Mathematical Definition
KWZ	KWZ Closed-Form	$\hat{w}_{q,t}$
MV	Mean-Variance Plugin	$\hat{w}_{p,t}$
MV_UB	Mean-Variance Unbiased	$\hat{w}_{u,t}$
KWZ_LW_2004	KWZ Closed-Form with Linear Shrinkage	$\hat{w}_{q,t}^{LW2004}$
MV_LW_2004	Mean-Variance with Linear Shrinkage	$\hat{w}_{p,t}^{LW2004}$
MV_NS	Mean-Variance Plugin without Short-Sales	$\hat{w}_{p,t}^{NS}$
GMV	Global Minimum Variance Plugin	$\hat{w}_{g,t}$
GMV_NS	Global Minimum Variance Plugin without Short-Sales	$\hat{w}_{g,t}^{NS}$
Naive	Equally Weighted Portfolio	$1/N$
KO_VT	Volatility Timing by Kirby and Ostdiek (2012)	KO_{VT}
KO_RT	Reward-to-Risk Timing by Kirby and Ostdiek (2012)	KO_{RT}

We note that as long as the function `DR_function` organizes the portfolio weights in a single list, the backtesting procedure in the following section can be efficiently utilized to accommodate other decision rules and estimation methods.

Backtesting

We can run the `DR_function` on a rolling basis to perform backtesting. Nonetheless, note that three main inputs are needed to reduplicate the main results from KWZ:

1. The risk aversion level denoted by γ (`gamma`).
2. Sample size denoted by T (`sample_size`).
3. Transactions cost (`TC`).

Similar to the baseline analysis from KWZ (see, e.g., Tables 1, 2, and 3), we set

```
gamma <- 3
sample_size <- 120
TC <- 0/(100^2) # to reflect basis points
```

The analysis in this article **does not take into consideration transaction costs**. However, with the right calculation of the portfolio gross return and turnover, one can easily impose a penalty on the portfolio return to reflect transaction costs. In the following code, we utilize `DR_function` to write a single main function that takes the data index and the above three inputs to evaluate the portfolio rules in terms of out-of-sample returns.

```
main_run_portfolio_fun <- function(choose_data,gamma,sample_size,TC) {

  # choose the data - in total we have six different sets
  ds <- ds_list[[choose_data]]
  months_tot <- sort(unique(ds$date))

  # store data
  results_all <- data.frame()

  # store portfolio weights - relevant for portfolio turnover and TC
  W_ALL_list <- list()

  # run a loop from T until the end-1 of data
  for(i in sample_size:(length(months_tot) - 1)) {

    # current month
    month_i <- months_tot[i]
    # next month, which is unknown during portfolio construction
    month_i_plus <- months_tot[i+1]

    # for tracking backtesting
    if( month(month_i_plus) == 12) {
      cat("this is month ", as.character(month_i_plus),"\n")
    }

    # define the in-sample data
    R_sub <- ds[ds$date <= month_i,]
    run_DR_fun <- DR_function(R_sub,gamma,sample_size,TC)


    # keep track of weights and stack in matrix
    W_1_list <- lapply(run_DR_fun,t)
    W_1_list <- lapply(W_1_list,function(x) data.frame(month_i_plus,x) )
    W_ALL_list <- c(W_ALL_list,list(W_1_list))

    R_next <- ds[ds$date == month_i_plus,-1]
    R_port <- lapply(W_1_list, function(x)  sum(x[,-1]*R_next)  )
    R_port2 <- data.frame(date = month_i_plus,Reduce(cbind,R_port))
    names(R_port2)[-1] <- names(R_port)
    results_all <- rbind(results_all,R_port2)
  }

  list(port_ret = results_all, weights = W_ALL_list)
}
```

The loop in the above function computes two main objects. The first one contains the out-of-sample portfolio returns `results_all`, whereas the other is a list of all portfolio weights over time denoted by `W_ALL_list`. Given the former, we can easily compute gross returns and other performance measures. For the latter, we can consider performance in terms of net transaction cost and other portfolio characteristics such as turnover or diversification. As mentioned above, the function is generalized to evaluate multiple decision rules simultaneously by utilizing the `lapply` base function.

As an illustration, let us consider the gross returns and replicate some of the results of Tables 1 and 2 from KWZ, which does not consider transaction costs. In particular, Tables 1 and 2 from KWZ report the certainty equivalent returns (CER) and the Sharpe ratio (SR) for different decision rules (rows) and data sets (columns). In our case, we have eleven decision rules and six data sets. Given that the data is public, we expect identical results, especially for the closed-form decision rules, which are less influenced by hidden codes or "black box" models. However, researchers should be aware of one issue when it comes to the Fama-French data. It appears that the same time series of factor returns changes over time due to ex-post adjustments in constructing the factors/portfolios. This issue has been raised by Akey et al. (2022), which the authors refer to as "Noisy Factors." Particularly, the authors find that "factor returns differ substantially depending on when the data were downloaded, and only a small portion of these retroactive changes is explained by revisions in the underlying data." Nonetheless, this issue should result in a slight difference, given that KWZ was conducted post-2018.

To implement, we utilize `main_run_portfolio_fun` across all data sets and compute the time series returns of our decision rules in the following manner:

```
run_experiment <- mclapply(1:length(ds_list),
                           function(data_index) main_run_portfolio_fun(data_index,gamma,sample_size,TC = 0),
                           mc.cores = detectCores() )
```

Note that the above command runs the backtesting experiment across all data sets using three main inputs, specifying the risk aversion, sample size, and transaction cost. Given the portfolio returns from each data experiment, we can easily compute the CER and SR for each portfolio-data specification:

```
sr_fun <- function(x) round(mean(x)/sd(x),4)
cer_fun <- function(x) round(mean(x) - (gamma/2)*var(x),4)

port_ret_list <- lapply(run_experiment,function(x) x$port_ret)
MS_CER <- sapply(port_ret_list, function(x) apply(x[,-1],2,cer_fun))
MS_SR <- sapply(port_ret_list, function(x) apply(x[,-1],2,sr_fun))
```

For a consistent comparison, let us copy the corresponding CER and the SR values from Tables 1 and 2 of KWZ, respectively. With some adjustments, the tables can be copied from the manuscript into R as follows:

```
Table1_KWZ <- "KWZ 0.0098 0.0102 0.0064 0.0060 0.0081 0.0259 0.0022 -0.0068
MV -0.0063 -0.0635 -0.0141 -0.0705 -0.0270 -1.2294 -0.3998 -27.0937
```

```
MV_UB -0.0013 -0.0284 -0.0075 -0.0361 -0.0147 -0.3039 -0.1278 -0.7464
KWZ_LW_2004 0.0120 0.0112 0.0100 0.0083 0.0100 0.0695 0.0042 0.0020
MV_LW_2004 0.0070 -0.0009 0.0048 -0.0219 0.0013 -0.0444 -0.1243 -0.5255
MV_NS 0.0060 0.0050 0.0028 0.0045 0.0048 0.0090 0.0003 0.0000
GMV 0.0051 0.0060 0.0020 0.0050 0.0050 0.0035 0.0025 -0.0056
GMV_NS 0.0035 0.0041 0.0041 0.0049 0.0025 0.0059 0.0042 0.0027
Naive 0.0026 0.0036 0.0022 0.0034 0.0022 0.0013 0.0039 0.0024
KO_VT 0.0036 0.0043 0.0041 0.0045 0.0030 0.0046 0.0049 0.0037
KO_RT 0.0047 0.0045 0.0034 0.0049 0.0046 0.0061 0.0038 0.0019"

Table2_KWZ <- "KWZ 0.2521 0.2479 0.2452 0.1916 0.2238 0.5778 0.1194 0.0234
MV 0.2375 0.1999 0.2387 0.1470 0.1896 0.5614 0.0610 -0.0482
MV_UB 0.2405 0.2069 0.2400 0.1534 0.1940 0.5648 0.0671 -0.0428
KWZ_LW_2004 0.2693 0.2838 0.2456 0.2312 0.2591 0.6925 0.1661 0.1093
MV_LW_2004 0.2545 0.2393 0.2383 0.1792 0.2118 0.6558 0.0728 -0.0138
MV_NS 0.1905 0.1734 0.1299 0.1644 0.1700 0.2495 0.0996 0.0943
GMV 0.1823 0.2085 0.1088 0.1813 0.1843 0.1478 0.1227 0.0369
GMV_NS 0.1450 0.1595 0.1639 0.1816 0.1236 0.2188 0.1717 0.1291
Naive 0.1276 0.1484 0.1224 0.1426 0.1209 0.1034 0.1527 0.1205
KO_VT 0.1466 0.1636 0.1619 0.1704 0.1338 0.1751 0.1902 0.1584
KO_RT 0.1695 0.1651 0.1438 0.1768 0.1665 0.2046 0.1534 0.1078"
```

However, to process in a readable format, we need a function that takes the above content and loads it into a readable matrix:

```
read_table_fun <- function(KWZ_table) {
  x <- scan(textConnection(KWZ_table),what = character(),sep = "\n")
  x <- strsplit(x," ")
  x_names <- sapply(x,function(x) x[1] )
  X <- t(sapply(x,function(x) as.numeric(x[-1]) ))
  colnames(X) <- c("MOM10","BM25",  "IVOL", "OPIN25", "NMV16", "NMV46", "IND49","Stocks")
  # keep relevant data sets
  X <- X[,c("MOM10","BM25","OPIN25","NMV16","NMV46","IND49")]
  rownames(X) <- x_names
  KWZ_table <- X
  return(KWZ_table)
}

Table1_KWZ <- read_table_fun(Table1_KWZ)
Table2_KWZ <- read_table_fun(Table2_KWZ)
```

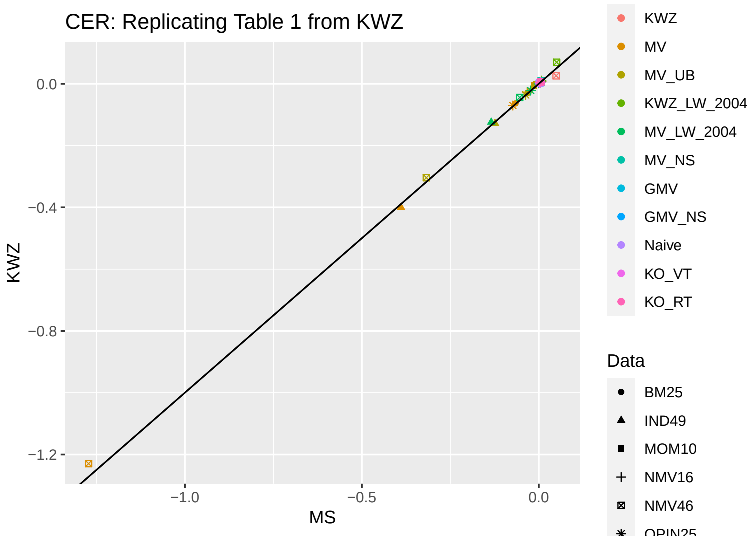
As a summary, we plot our results (denoted by MS) versus those by KWZ using the `ggplot2` library. To do so, we summarize all results in a `data.frame` that is user-friendly to input into `ggplot`:

```
ds_plot <- data.frame()
for (r in 1:nrow(MS_CER)) {
  ds_plot_r <- data.frame(KWZ = Table1_KWZ[r,], MS = MS_CER[r,],
                          Data = names(ds_list), Portfolio = rownames(MS_CER)[r] )
  ds_plot <- rbind(ds_plot,ds_plot_r)
}

# save data
ds_plot1 <- ds_plot
ds_plot1$Metric <- "CER"

ds_plot <- data.frame()
for (r in 1:nrow(MS_SR)) {
  ds_plot_r <- data.frame(KWZ = Table2_KWZ[r,], MS = MS_SR[r,],
                          Data = names(ds_list), Portfolio = rownames(MS_SR)[r] )
  ds_plot <- rbind(ds_plot,ds_plot_r)
}
ds_plot2 <- ds_plot
ds_plot2$Metric <- "SR"

# stack data altogether
ds_plot <- rbind(ds_plot1,ds_plot2)
```

Tables 1 and 2

Since the CER and SR correspond to different performance metrics, we visualize each separately. For the CER, we have:

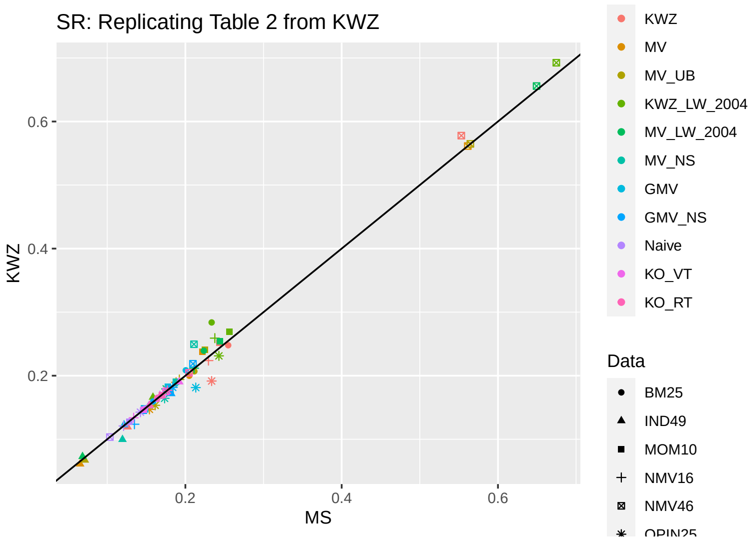
```
p <- ggplot(ds_plot[ds_plot$Metric == "CER",], aes(y = KWZ, x = MS,colour = Portfolio,shape = Data))
p <- p + geom_point()
p <- p + geom_abline(slope = 1, intercept = 0)
p <- p + ggtitle("CER: Replicating Table 1 from KWZ")
p
```



Overall, the results seem consistent, with some slight variations in few cases. The most significant deviation comes from the MV decision rule, which also results in the worst performance. This is most evident in the NVM46 data set.

Let us repeat the same plot for the or the SR:

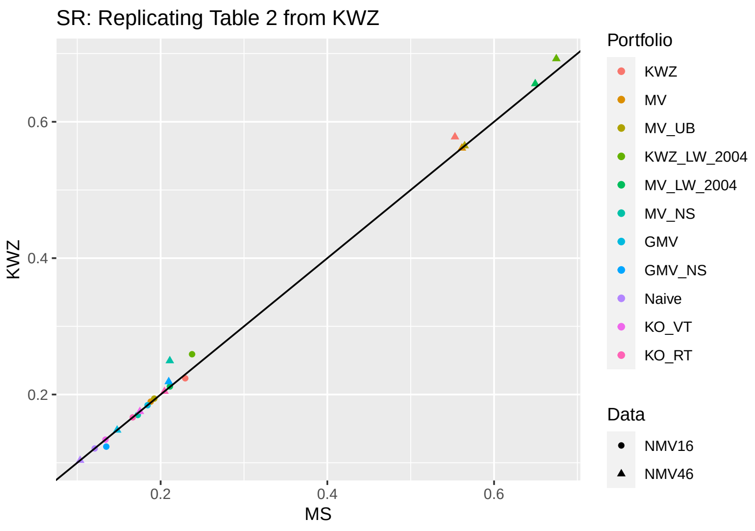
```
p <- ggplot(ds_plot[ds_plot$Metric == "SR",], aes(y = KWZ, x = MS,colour = Portfolio,shape = Data))
p <- p + geom_point()
p <- p + geom_abline(slope = 1, intercept = 0)
p <- p + ggtitle("SR: Replicating Table 2 from KWZ")
p
```



We observe that most decision rules result in SR between 0 and 0.4; whereas in a few cases, the SR is greater than 0.5. This is mainly the case for the NMV46 data. Nonetheless, we observe a good consistency between the original findings by KWZ and ours. There are, however, a few slight deviations. We expect identical results for the MV, GMV, and Naive decision rules since we are using closed-form solutions, i.e., there are no degrees of freedom for an error. For the short sales, there are some degrees of freedom in terms of initial weights and how the optimization algorithm used in R differs from the one used by KWZ, which could be built using different software. The same applies to shrinkage-based techniques.

Additionally, such deviations could be attributed to the fact that the Fama-French data is subjected to changes over time, as pointed out by Akey et al. (2022). To investigate this issue, let us focus on the NMV16 and NMV46 data sets:

```
ds_plot_NVM <- ds_plot[ds_plot$Data %in% c("NMV16","NMV46") & ds_plot$Metric == "SR",]
p <- ggplot(ds_plot_NVM, aes(y = KWZ, x = MS,colour = Portfolio,shape = Data))
p <- p + geom_point()
p <- p + geom_abline(slope = 1, intercept = 0)
p <- p + ggtitle("SR: Replicating Table 2 from KWZ")
p
```



As conjectured above, the closed-form solution portfolios are consistent with KWZ. However, we note a slight difference in the decision rules with no short sales, which requires numerical optimization. A similar observation follows when we compare the shrinkage-based techniques.

Table 3: Turnover

Finally, let us evaluate the portfolio turnover of each decision rule. To do so, we follow the same metric used by KWZ - see Equation (60) from the paper. The following nested loop computes the turnover across different data sources and decision rules:

```
MS_TO <- c()
for (data_i in 1:length(ds_list)) {
  TO_vec <- c()
  for (decision_rule in rownames(MS_SR)) {
    W_list_data <- run_experiment[[data_i]]$weights
    W_list_data_DR <- lapply(W_list_data, function(x) data.frame(x[names(x) %in% decision_rule]) )
    W_mat <- Reduce(rbind,W_list_data_DR)
    R_mat <- ds_list[[data_i]]
    W_mat <- W_mat[-nrow(W_mat),]
    R_mat <- R_mat[R_mat$date %in% W_mat[,1],]
    W_mat_adj <- W_mat[,-1]*(1 + R_mat[,-1])
    W_mat_adj <- t(apply(W_mat_adj,1,function(x) x/sum(x))) # adjust positions to equal one
    TO <- W_mat[-1,-1] - W_mat_adj[-nrow(W_mat_adj),]
    TO <- apply(TO,1,function(x) sum(abs(x)) )
    TO_mean <- mean(TO)
    TO_vec <- c(TO_vec,TO_mean)
  }
  MS_TO <- cbind(MS_TO,TO_vec)
}
```

```
rownames(MS_TO) <- rownames(MS_SR)
colnames(MS_TO) <- names(ds_list)
```

The above code takes into consideration that positions do change over time. For instance, one may conjecture that the naive portfolio has a zero turnover since its weights are constant over time; however, this is not the case. Its positive turnover comes mainly from the fact that the portfolio is re-balanced each month to maintain equal weights after considering the gross returns and, hence, changes in the portfolio position. To better understand how it is computed, consider the following example:

```
d <- 5
W_1 <- rep(1/d,d)
W_1_adj <- W_1*(1+rnorm(d,0,0.06))
W_2 <- W_1
rbind(W_1_adj,W_2)
```

```
#>              [,1]      [,2]      [,3]      [,4]      [,5]
#> W_1_adj 0.1964149 0.1757547 0.1948244 0.1978983 0.2125213
#> W_2     0.2000000 0.2000000 0.2000000 0.2000000 0.2000000
```

As the returns are realized, the initial positions change as well. In order to bring back to equal weights, the portfolio manager needs to re-balance, resulting in a turnover of

```
sum(abs(W_2 - W_1_adj))
```

```
#> [1] 0.04762892
```

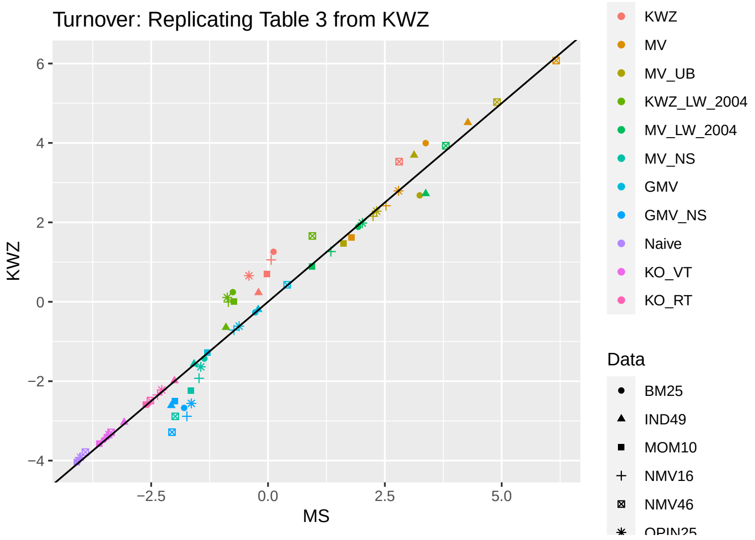
The operations of the above simple example are conducted for each month-rule-data. As a final summary, it is common to report the average over time for each rule-data combination.

For a consistent comparison, let us collect the corresponding data from KWZ (Table 3) as we did for Tables 1 and 2:

```
Table3_KWZ <- "KWZ 2.0157 3.5212 2.9934 1.9283 2.8730 34.0907 1.2583 3.8888
MV 5.0765 54.3620 7.1199 16.3314 11.2294 434.8788 91.2610 1275.5124
MV_UB 4.3229 14.5806 6.5314 9.7989 8.6225 152.8220 40.1044 224.0102
KWZ_LW_2004 1.0061 1.2745 1.0879 1.1136 0.9959 5.2506 0.5241 0.6093
MV_LW_2004 2.4207 6.6031 2.6552 7.2740 3.5400 50.8673 15.2265 212.4453
MV_NS 0.1066 0.2393 0.1808 0.1945 0.1456 0.0558 0.2092 0.2648
GMV 0.2770 0.7665 0.2640 0.5413 0.4911 1.5337 0.8227 3.7827
GMV_NS 0.0817 0.0691 0.0088 0.0774 0.0559 0.0375 0.0733 0.1440
Naive 0.0176 0.0182 0.0172 0.0199 0.0197 0.0227 0.0341 0.0648
KO_VT 0.0281 0.0309 0.0188 0.0353 0.0347 0.0373 0.0481 0.0706
KO_RT 0.0746 0.0767 0.0886 0.1086 0.0959 0.0832 0.1369 0.1591"

Table3_KWZ <- read_table_fun(Table3_KWZ)
```

Similar to the summary of Tables 1 and 2, the following code organizes the turnover results in a user-friendly data frame that is efficiently loaded into ggplot:

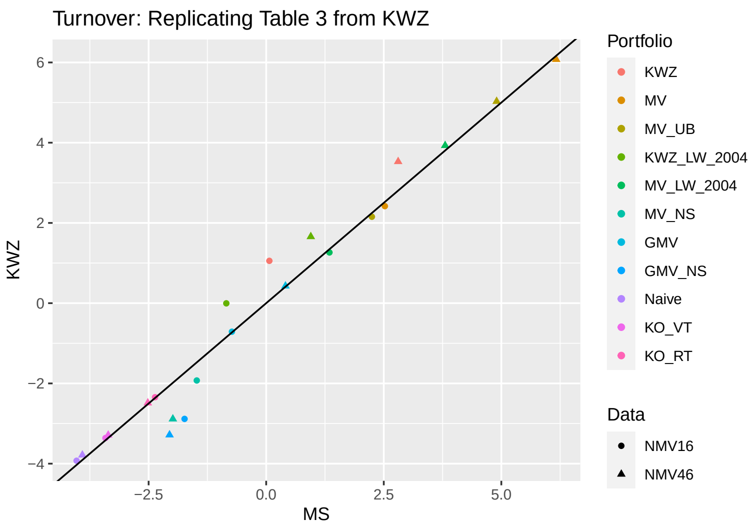
```
ds_plot_TO <- data.frame()
for (r in 1:nrow(MS_TO)) {
  ds_plot_r <- data.frame(KWZ = Table3_KWZ[r,], MS = MS_TO[r,],
                          Data = names(ds_list), Portfolio = rownames(MS_TO)[r] )
  ds_plot_TO <- rbind(ds_plot_TO,ds_plot_r)
}

ds_plot_TO$KWZ <- log(ds_plot_TO$KWZ)
ds_plot_TO$MS <- log(ds_plot_TO$MS)

p <- ggplot(ds_plot_TO, aes(y = KWZ, x = MS,colour = Portfolio,shape = Data))
p <- p + geom_point()
p <- p + geom_abline(slope = 1, intercept = 0)
p <- p + ggtitle("Turnover: Replicating Table 3 from KWZ")
p
```



Note that the above code uses a log scale to control for fixed effects associated with data and decision rules. For instance, the MV decision rule exhibits huge unrealistic positions. This evidence is consistent with the previous literature on estimation error in portfolio selection. Overall, we find consistent results with KWZ. The major flags stem from those with short-sale constraints.

To check the sensitivity with respect to the Fama-French data, let us repeat the same exercise as we did with the CER and SR results, where we keep the NMV16 and NMV46 data sets:

```
ds_plot_TO_sub <- ds_plot_TO[ds_plot_TO$Data %in% c("NMV16","NMV46"),]
p <- ggplot(ds_plot_TO_sub, aes(y = KWZ, x = MS,colour = Portfolio,shape = Data))
p <- p + geom_point()
p <- p + geom_abline(slope = 1, intercept = 0)
p <- p + ggtitle("Turnover: Replicating Table 3 from KWZ")
p
```



Interestingly, we still witness slight variations; however, our results seem consistent overall with KWZ. Similar to the SR performance results, we note that these deviations mainly stem from the portfolio rules that depend on numerical optimization or rely on non-base R packages.

Concluding Remarks

This article is designed to evaluate various portfolio rules using different public data sets. As an illustration, we compute 198 performance metrics reported by Kan et al. (2022) (KWZ), which sets an excellent ground to demonstrate our replication initiative. While our investigation covers a subset of the tests/results reported by the authors, our empirical analysis covers standard benchmarks that future researchers should take into consideration in their "pursuit of the perfect portfolio." Overall, the article finds consistent results with the original research, despite potential differences in the software. Such differences may also be attributed to the time lapse since the empirical analysis by KWZ was conducted, which goes to the "noisy factors" argument by Akey et al. (2022). Another critical part that needs to be taken into account is statistical significance. It is common to utilize a bootstrap methodology to determine whether one rule significantly outperforms another, which we leave for future research.

Source Code

The source used in this article can be retrieved via this vignette.

Bibliography

P. Akey, A. Robertson, and M. Simutin. Noisy factors. *Rotman School of Management Working Paper Forthcoming*, 2022. [p]

R. Kan, X. Wang, and G. Zhou. Optimal portfolio choice with estimation risk: No risk-free asset case. *Management Science*, 68(3):2047–2068, 2022. [p]

C. Kirby and B. Ostdiek. It's all in the timing: simple active portfolio strategies that outperform naive diversification. *Journal of Financial and Quantitative Analysis*, 47(2):437–467, 2012. [p]

R. W. Klein and V. S. Bawa. The effect of estimation risk on optimal portfolio choice. *Journal of financial economics*, 3(3):215–231, 1976. [p]

N. Lassance, A. Martin-Utrera, and M. Simaan. The risk of out-of-sample portfolio performance. *Available at SSRN 3855546*, 2022. [p]

H. Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952. [p]

R. Novy-Marx and M. Velikov. A taxonomy of anomalies and their trading costs. *The Review of Financial Studies*, 29(1):104–147, 2016. [p]

M. Simaan, Y. Simaan, and Y. Tang. Estimation error in mean returns and the mean-variance efficient frontier. *International Review of Economics & Finance*, 56:109–124, 2018. [p]

Majeed Simaan
School of Business, Stevens Institute of Technology
1 Castle Point Terrace, Babbio Center, Hoboken, NJ 07030, USA.
msimaan@stevens.edu