STA457H1 Assignment #2

Team Information

Course: STA457H1S LEC0101

Introduction to Economic Factors

Team Leader: Ziyang Lin (Student ID: 1003246551)

Team Member: Yingxin Cui (Student ID: 1003161437)

Due Date: 2019/4/12

General Procedure

By reading Section 1 to Section 3 of the assignment handout, we obtained basic understanding of economic variables and their uses (signals and factor mimicking protfolios). In this assignment, we will utilize the knowledge of multivariate time series models to analyze economic variables data and construct factor mimicking portfolios and discuss their performance.

Some key equations to consider in this assignment includes:

Multifactor model: where R_{it} denotes the return on asset i in period t ($1 \leq i \leq N$), $f_{j,t}$ the *realization* (i.e. the signal or unanticipated shock discussed below) of the j-th factor in period t ($1 \leq j \leq N$), and ϵ_t the error term, and $t = 1, ..., T$ where T is the number of time series observations. (*Equation 1.1*)

$$R_{it} = \alpha_i + \beta_{i1} f_{1t} + \beta_{i2} f_{2t} + ... + \beta_{iK} f_{Kt} + \epsilon_{it}$$

Discounted Cash Flow Model: where p_{it} is the present value of asset i in DCF model, $\rho_{t,s}$ is the discount rate at time t for expected cash flows at time $t + s$, and $E(c_{i,t+s})$ the expected cash flow. (*Equation 1.2*)

$$p_{it} = \sum_{s=1}^{\infty} \frac{E[c_{i,t+s}]}{(1 + \rho_{t,s})^s}$$

Signal: where x_t is any economic variable in period t, E_{t-1} an expectation operator that uses information up to the end of period $t - 1$, and u_t the corresponding signal. (*Equation 1.3*)

$$u_t = x_t - E_{t-1} \cdot x_t$$

VAR approach: let $x_{k,t}$ where $k = 1, ..., K$ and $t = 1, ..., T$ denote the k-th economic state variable in period t, and $\boldsymbol{z}_t = (x_{1,t}, ..., x_{K,t})^T$. The VAR approach assumes $\boldsymbol{z}_t$ follows a first-order VAR given by the below. (*Equation 1.4*)

$$\boldsymbol{z}_t = \boldsymbol{A}\boldsymbol{z}_{t-1} + \boldsymbol{u}_t$$

Fama-MacBeth Method: in the second stage, where $R_t = (R_{1t}, ..., R_{Nt})^T$ is the access return for period t, and $\hat{X}$ is the matrix formed by $\hat{X} = [1_N, \hat{\boldsymbol{\beta}}]$ where $\hat{\boldsymbol{\beta}}$ given as in below. (*Equation 1.5*)

$$\hat{\boldsymbol{\Gamma}}_{\boldsymbol{t}} = (\hat{\boldsymbol{X}}' \hat{\boldsymbol{X}})^{-1} \hat{\boldsymbol{X}}' \boldsymbol{R}_{\boldsymbol{t}}, \hat{\boldsymbol{X}} = \begin{bmatrix} 1 & \hat{\beta}_{11} & ... & \hat{\beta}_{1N} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \hat{\beta}_{N1} & ... & \hat{\beta}_{NN} \end{bmatrix}$$

We will first download the economic variable data from the internet, and retrieve them in R to obtain our desired economic factors. Then we will estimate the signals using VAR approach, and then construct the economic factor mimicking portfolios. We will also construct a factor momentum portfolio as what we did for the 30 Dow-Jones constituents by re-using the method in Assignment 1.

Data Retrieval

In this part, we will download the data from several online sources, and then construct the following 5 macroeconomic variables as our economic factors for further analysis and portfolio constructions.

1. `OIL`: The change rate on the crude oil price (monthly-term)
2. `TERM`: The difference between long-term government bond yield and the 1-year constant maturity rate (monthly-term)
3. `DEF`: Moody's Seasoned Baa Corporate Bond Yield relative to Yield on 10-year Treasury Constant Maturity (monthly-term)
4. `R_m`: Excess market return from the Fama-French dataset
5. `DIF`: Diffusion index for FRB - Philadelphia District

We will first utilize the `quantmod` library in R to access the online data and for analysis, and construct a new environment to store the data.

```
library(quantmod)
```

```
## Loading required package: xts
```

```
## Loading required package: zoo
```

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

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

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

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

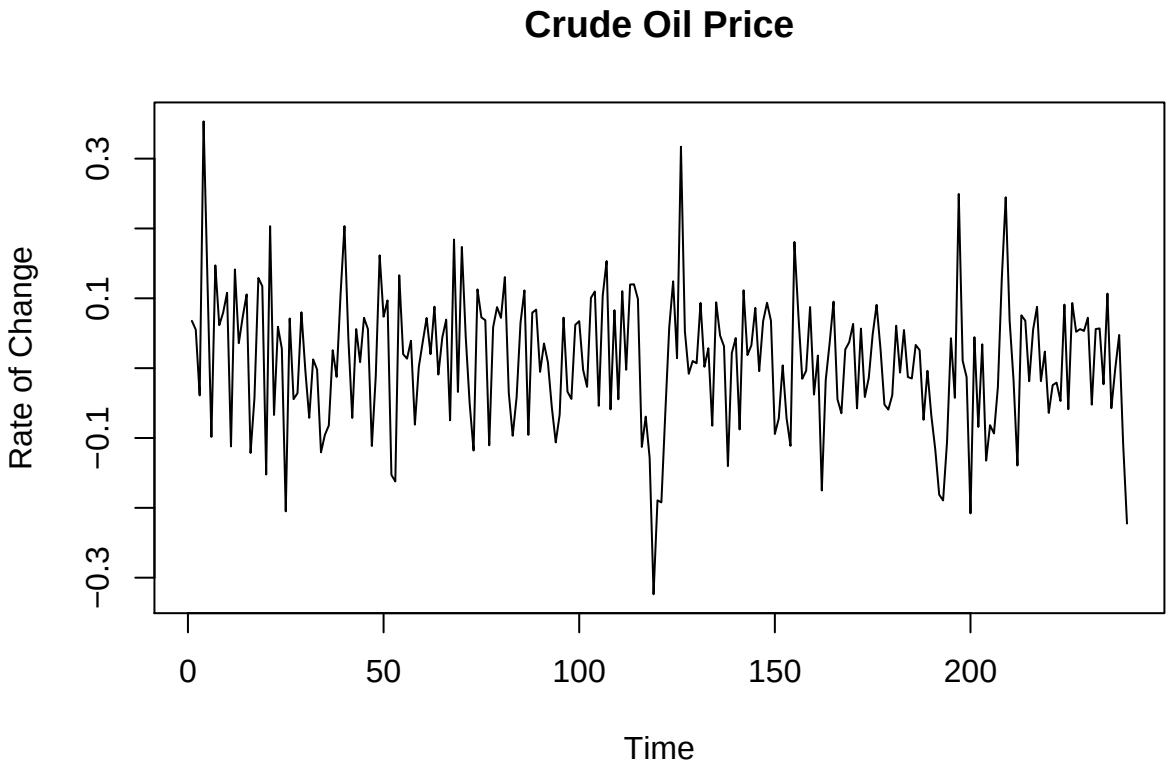
```
data <- new.env()
```

Note that for all data we download and retrieve below, we will use a 20-year time range from 1998/11 to 2018/10 (end of each month) for simplicity, so that our multivariate time series model can be constructed with the correct corresponding time for each economic factor.

Construction of `OIL` Factor:

We download the data from St. Louis Fed website, and then apply the specified time range as below to obtain the rate of change in price from 1998/12 to 2018/11. Part of the data is displayed as below.

```
date.start <- "1998-11-28"
date.end <- "2018-11-30"
getSymbols.FRED("DCOILWTICO", env=data)
oilPrice <- na.omit(data$DCOILWTICO)
oilPrice <- apply.monthly(oilPrice, last)
oilPrice <- oilPrice[paste(date.start, date.end, sep="/")] # Apply the given time range
n <- length(oilPrice)
OIL <- NULL
# Calculate the monthly price change of crude oil.
for (i in 1:(n-1)) {
  OIL <- c(OIL, (oilPrice[[i+1]]-oilPrice[[i]])/oilPrice[[i]])
}
```

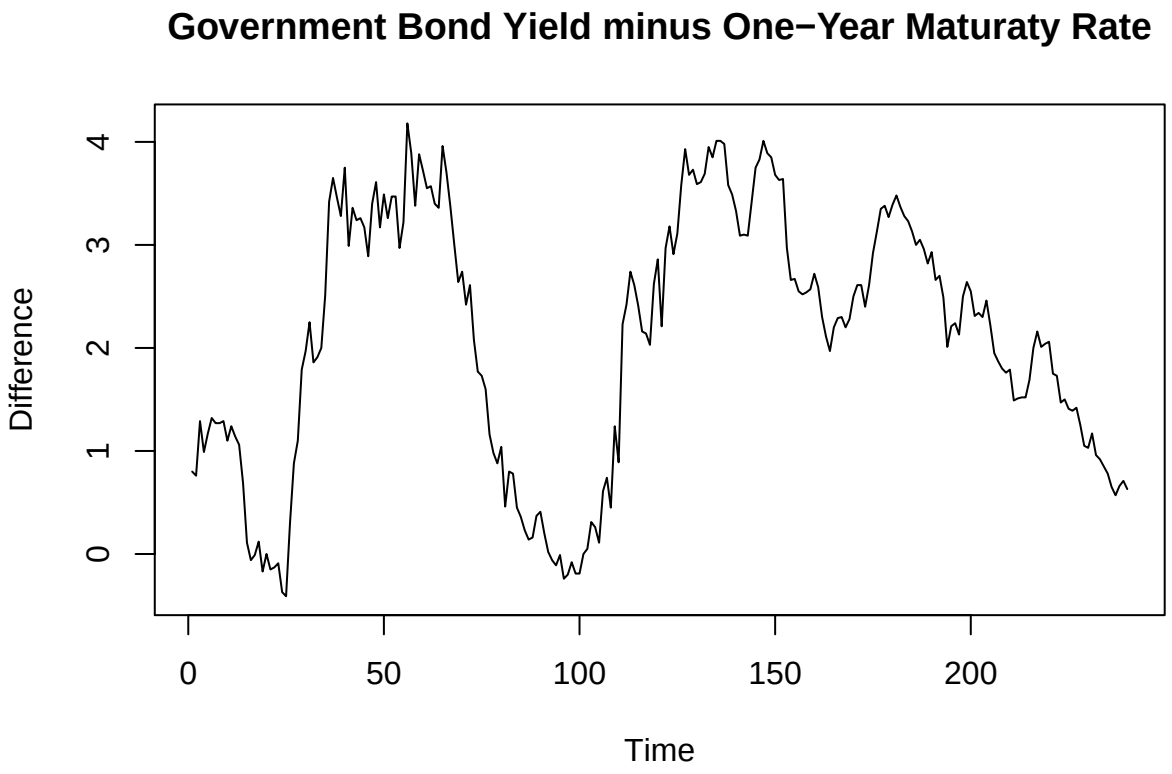


Construction of `TERM` Factor:

As we did with `OIL`, we download the data and then convert it to monthly data with the specified range. Note that the government bond yield is recorded at the start of the month, so for simplicity purpose, we calculate the difference from end of a month in `OneYear` and start of the next month of that month in `GovBond`.

For example, the first entry in `TERM` is `GovBond` at 1998/12/01 minus `OneYear` at 1998/11/30. In this case, we ensure the difference is of the closest to actual values.

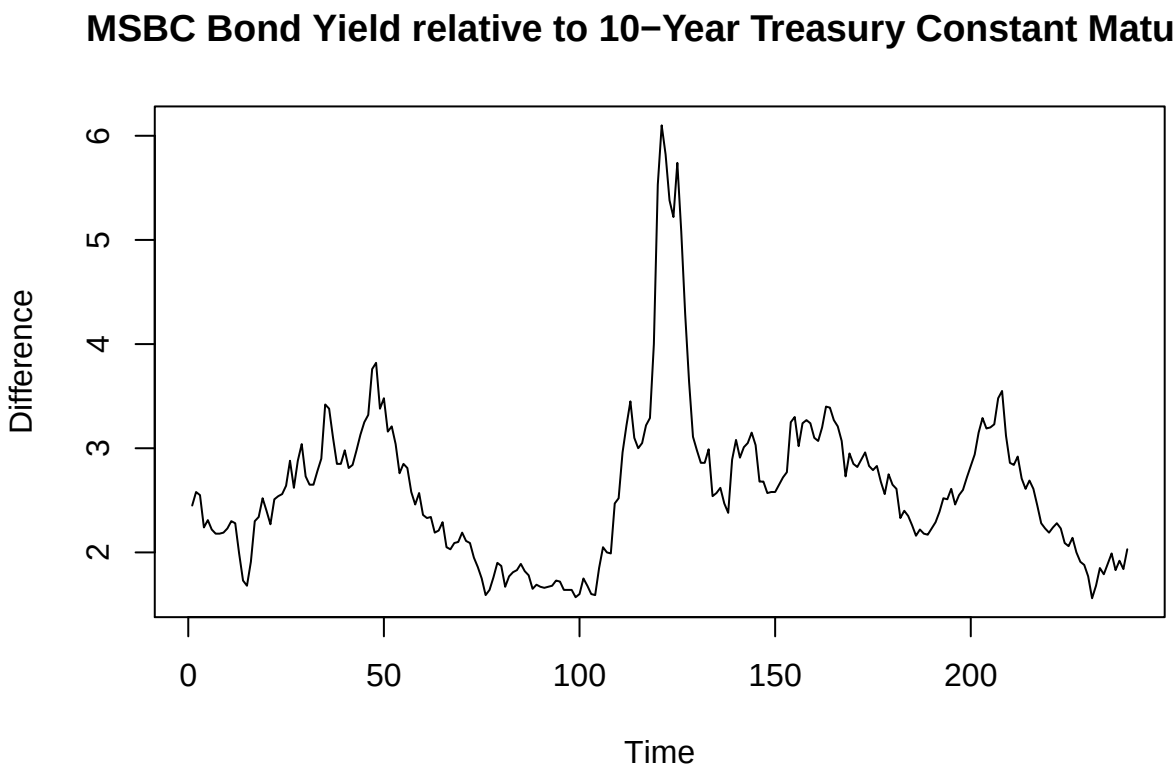
```
# Download daily data for 1-Year constant maturity rate, then retreive as monthly.
getSymbols.FRED("DGS1",env=data)
OneYear <- na.omit(data$DGS1)
OneYear <- apply.monthly(OneYear, last)
OneYear <- OneYear[paste(date.start, "2018-10-31", sep="/")]
# Download daily data for the long-term government bond yield, then retrieve as monthly.
getSymbols.FRED("IRLTCT01USM156N",env=data)
GovBond <- na.omit(data$IRLTCT01USM156N)
GovBond <- apply.monthly(GovBond, last)
GovBond <- GovBond[paste(date.start, "2018-11-1", sep="/")]
n <- length(GovBond)
# Calculate the difference between GovBond and OneYear.
TERM <- NULL
for (i in 1:n) {
  TERM <- c(TERM, GovBond[[i]]-OneYear[[i]])
}
```



Construction of DEF Factor:

The same procedure as we did with OIL, and no other calculations needed.

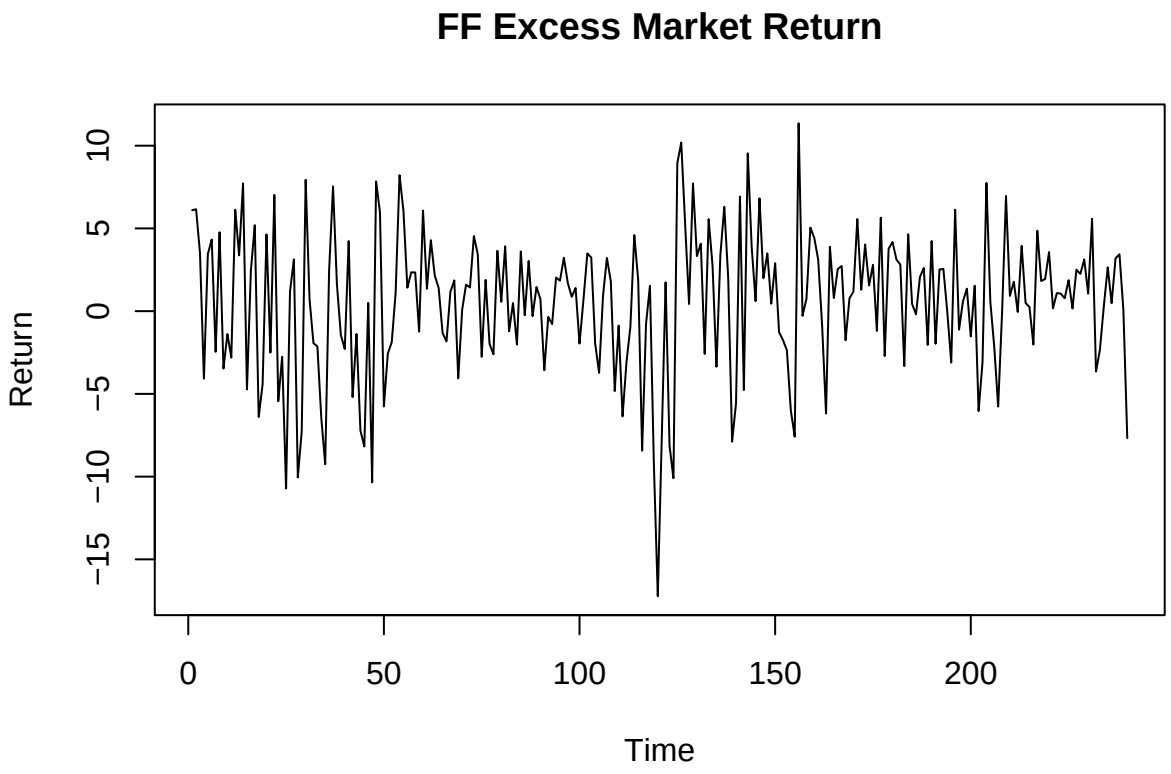
```
# Download daily data for the MSBC Bond Yield relative to
#      10-Year Treasury Constant Maturity, and retrieve as monthly.
getSymbols.FRED("BAA10Y",env=data)
DEF <- na.omit(data$BAA10Y)
DEF <- apply.monthly(DEF, last)
DEF <- DEF[paste(date.start, "2018-11-1", sep="/")]
```



Construction of `R_m` Factor:

We download the below 3 `.csv` files from the Fama-French data library, for reading simplicity, we delete all irrelevant data from the files, and only consider the data for monthly average equal weighted returns. And for `R_mt` factor, we will only use the excess market return recorded in the `FF3` dataset in our specified time range.

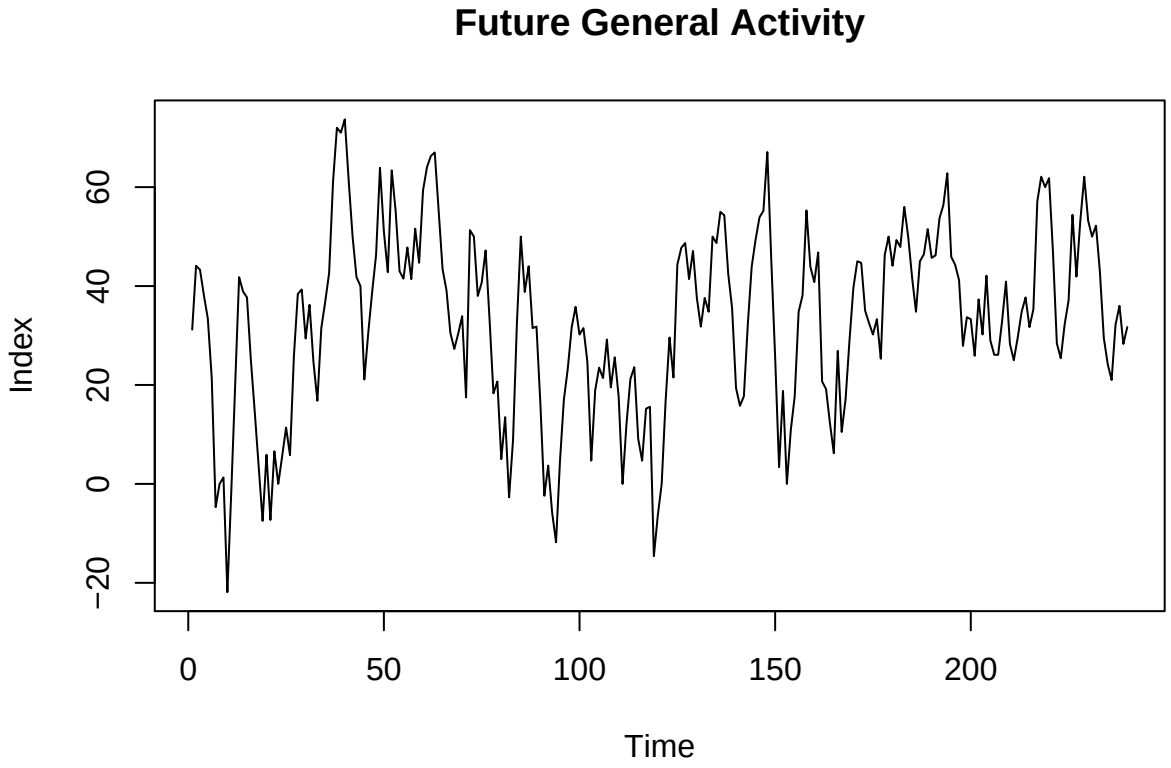
```
FF25 <- read.csv("25_Portfolios_5x5.csv",skip=15)
FF30 <- read.csv("30_Industry_Portfolios.csv",skip=11)
# Only retreive the data from the below file.
FF3 <- read.csv("F-F_Research_Data_Factors.csv",skip=11)
R_m <- FF3[[2]][861:1100] # Excess market return in the specified time range.
```



Construction of DIF Factor:

The same procedure as we did with OIL, and no other calculations needed.

```
# Download daily data for Future General Activity, and retrieve as monthly.
getSymbols.FRED("GAFDFNA066MNFRBPHI",env=data)
DIF_temp <- na.omit(data$GAFDFNA066MNFRBPHI)
DIF_temp <- apply.monthly(DIF_temp, last)
DIF_temp <- DIF_temp[paste(date.start, "2018-11-1", sep="/")]
DIF <- NULL
for (i in 1:240) {
  DIF <- c(DIF, DIF_temp[[i]])
}
```


Future General Activity
Index
60
40
20
0
−20
0
50
100
150
200
Time

Estimation of Unanticipated Shocks

Consider the below equation (*Equation 2.1*) for $VAR(p)$-process with the above 5 economic factors. We have the excess market return `R_m` as well as the 4 predictors.

$$\begin{bmatrix} R_{M,t} \\ TERM_t \\ DEF_t \\ OIL_t \\ DIF_t \end{bmatrix} = \sum_{i=1}^{p} \boldsymbol{A}_i \begin{bmatrix} R_{M,t-i} \\ TERM_{t-i} \\ DEF_{t-i} \\ OIL_{t-i} \\ DIF_{t-i} \end{bmatrix} + \boldsymbol{u}_t$$

We will first examine the order selection criteria to choose the optimal order p for this VAR approach. Note that to do this, we will need to include the `vars` library in R. Consider the below code and outputs.

```
library(vars)
```

```
## Loading required package: MASS
## Loading required package: strucchange
## Loading required package: sandwich
## Loading required package: urca
## Loading required package: lmtest
```

```
y_t = cbind(R_m, TERM, DEF, OIL, DIF)
vars::VARselect(y_t, lag.max=8, type="none") # Model selection
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      2      1      1      2
##
## $criteria
##                 1           2           3           4           5
## AIC(n) -3.4235315 -3.42500962 -3.37891014 -3.34414975 -3.22858943
## HQ(n)  -3.2737434 -3.12543342 -2.92954585 -2.74499735 -2.47964893
## SC(n)  -3.0521158 -2.68217829 -2.26466315 -1.85848709 -1.37151111
## FPE(n)  0.0325982  0.03255767  0.03411538  0.03536599  0.03978007
##                 6           7           8
## AIC(n) -3.22063708 -3.20050527 -3.17265321
## HQ(n)  -2.32190848 -2.15198857 -1.97434841
## SC(n)  -0.99214309 -0.60059561 -0.20132789
## FPE(n)  0.04022138  0.04121713  0.04262742
```

Based on the model selection criteria, we will follow the optimal output for $AIC(n)$, that is, to use the $VAR(2)$ model (vector autoregression with lag 2). Now, we estimate the coefficient matrix for vector autoregression model with order 1. Note that we set the `type` in the below function call to be `"none"` because there is no deterministic term in *Equation 2.1*.

```
model <- vars::VAR(y_t, p=2, type="none") # Model estimation
summary(model$varresult$R_m)
```

```
##
## Call:
## lm(formula = y ~ -1 + ., data = datamat)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.9671  -2.3378   0.3682   2.6221   9.5056
##
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## R_m.l1    -0.01226    0.07605  -0.161 0.872048
## TERM.l1    1.80125    0.98303   1.832 0.068206 .
## BAA10Y.l1 -1.52884    1.59918  -0.956 0.340078
## OIL.l1    11.64064    2.97697   3.910 0.000122 ***
## DIF.l1     0.06854    0.02626   2.609 0.009670 **
## R_m.l2    -0.16250    0.06961  -2.334 0.020451 *
## TERM.l2   -1.65102    0.97441  -1.694 0.091557 .
## BAA10Y.l2  0.99590    1.61233   0.618 0.537406
## OIL.l2    -1.74089    2.98403  -0.583 0.560199
## DIF.l2    -0.02154    0.02559  -0.842 0.400906
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.05 on 228 degrees of freedom
## Multiple R-squared:  0.1608, Adjusted R-squared:  0.124
## F-statistic: 4.368 on 10 and 228 DF,  p-value: 1.302e-05
```

To examine the model adequacy, we apply the Portmanteau Test. Since our lag is not sufficiently large, therefore we will set the type of the test to `"PT.adjusted"`. Consider the below output, we see that the p-value is small, and therefore this model seems good.

```
serial.test(model, lags.pt=16, type="PT.adjusted") # Portmanteau Test
```

```
##
##  Portmanteau Test (adjusted)
##
## data:  Residuals of VAR object model
## Chi-squared = 453.99, df = 350, p-value = 0.0001462
```

Now, from `model` we extract the residuals for the market and the 4 predictors as innovations, and then orthogonalize them according to the method introduced by Petkova 2006 and Campbell 1996. Consider the below code and output to complete this task.

```
library(heplots)
```

```
## Loading required package: car
```

```
## Loading required package: carData
```

```
library(ppls)
```

```
## Loading required package: splines
```

```
# Extract the residual terms as innovations
market_innovations <- residuals(model)[,1]
TERM_innovations <- residuals(model)[,2]
DEF_innovations <- residuals(model)[,3]
OIL_innovations <- residuals(model)[,4]
```

```r
DIF_innovations <- residuals(model)[,5]
# 2-Stage orthogonalization: first orthogonalize each predictor accordingly, then normalize them.
temp <- matrix(c(market_innovations, TERM_innovations, DEF_innovations,
                 OIL_innovations, DIF_innovations), ncol=5, nrow=238)
TERM_orth <- normalize.vector(gsorth(temp, order=1:2)[,2])
DEF_orth <- normalize.vector(gsorth(temp, order=1:3)[,3])
OIL_orth <- normalize.vector(gsorth(temp, order=1:4)[,4])
DIF_orth <- normalize.vector(gsorth(temp, order=1:5)[,5])
innovations <- cbind(TERM_orth, DEF_orth, OIL_orth, DIF_orth)
head(innovations)
```

```
##          TERM_orth     DEF_orth    OIL_orth    DIF_orth
## row1  0.114645236 -0.004148316 -0.06742496  0.01005018
## row2 -0.075870149  0.120428001  0.20531129  0.02084545
## row3  0.044954734 -0.059879851  0.07406661 -0.02190805
## row4 -0.005897465 -0.016886507 -0.09137535 -0.05927391
## row5 -0.022755669  0.034699262  0.09367834 -0.15220813
## row6  0.009155456 -0.022265871  0.04798661 -0.02316349
```

Construction of Economic Factor Mimicking Portfolios

Now, we will proceed to utilize the signals we obtained above, *Equation 1.1*, and *Equation 1.5* to apply the FM method to the multifactor model to construct a factor mimicking portfolios from the FF industry portfolios dataset. We will first extract the time series from the dataset.

```
# Extract time series for each industry.
food <- FF30[,2]
beer <- FF30[,3]
smoke <- FF30[,4]
games <- FF30[,5]
books <- FF30[,6]
hshld <- FF30[,7]
clths <- FF30[,8]
hlth <- FF30[,9]
chems <- FF30[,10]
txtls <- FF30[,11]
cnstr <- FF30[,12]
steel <- FF30[,13]
fabpr <- FF30[,14]
elceq <- FF30[,15]
autos <- FF30[,16]
carry <- FF30[,17]
mines <- FF30[,18]
coal <- FF30[,19]
oil <- FF30[,20]
util <- FF30[,21]
telcm <- FF30[,22]
servs <- FF30[,23]
buseq <- FF30[,24]
paper <- FF30[,25]
trans <- FF30[,26]
whlsl <- FF30[,27]
rtail <- FF30[,28]
meals <- FF30[,29]
fin <- FF30[,30]
other <- FF30[,31]
```

In the first stage of the FM approach, we will use the $VAR(1)$ approach to regress excess return on each of the 30 industry portfolios R_{it} to the excess market return and all 4 innovations of predictors obtained. Note that for simplicity sake, we will only show the through all time (i.e. 1998/11 to 2018/10). But to apply the rolling window of 60-month of different time range, we can modify the index of each predictors in `Ri_t`. For example, a rolling window from 1998/12 to 2003/12 can be as following:

- `R1_t <- cbind(food[3:62], market_innovations[2:61], TERM_orth[2:61], DEF_orth[2:61], OIL_orth[2:61], DIF_orth[2:61])`

```
# VAR(1) approach for each of the 30 industry portfolios
R1_t <- cbind(food[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
food_mod <- vars::VAR(R1_t, p=1, type="const")$varresult[1]
R2_t <- cbind(beer[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
beer_mod <- vars::VAR(R2_t, p=1, type="const")$varresult[1]
R3_t <- cbind(smoke[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
smoke_mod <- vars::VAR(R3_t, p=1, type="const")$varresult[1]
```

```
R4_t <- cbind(games[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
games_mod <- vars::VAR(R4_t, p=1, type="const")$varresult[1]
R5_t <- cbind(books[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
books_mod <- vars::VAR(R5_t, p=1, type="const")$varresult[1]
R6_t <- cbind(hshld[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
hshld_mod <- vars::VAR(R6_t, p=1, type="const")$varresult[1]
R7_t <- cbind(clths[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
clths_mod <- vars::VAR(R7_t, p=1, type="const")$varresult[1]
R8_t <- cbind(hlth[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
hlth_mod <- vars::VAR(R8_t, p=1, type="const")$varresult[1]
R9_t <- cbind(chems[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
chems_mod <- vars::VAR(R9_t, p=1, type="const")$varresult[1]
R10_t <- cbind(txtls[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
txtls_mod <- vars::VAR(R10_t, p=1, type="const")$varresult[1]
R11_t <- cbind(cnstr[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
cnstr_mod <- vars::VAR(R11_t, p=1, type="const")$varresult[1]
R12_t <- cbind(steel[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
steel_mod <- vars::VAR(R12_t, p=1, type="const")$varresult[1]
R13_t <- cbind(fabpr[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
fabpr_mod <- vars::VAR(R13_t, p=1, type="const")$varresult[1]
R14_t <- cbind(elceq[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
elceq_mod <- vars::VAR(R14_t, p=1, type="const")$varresult[1]
R15_t <- cbind(autos[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
autos_mod <- vars::VAR(R15_t, p=1, type="const")$varresult[1]
R16_t <- cbind(carry[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
carry_mod <- vars::VAR(R16_t, p=1, type="const")$varresult[1]
R17_t <- cbind(mines[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
mines_mod <- vars::VAR(R17_t, p=1, type="const")$varresult[1]
R18_t <- cbind(coal[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
coal_mod <- vars::VAR(R18_t, p=1, type="const")$varresult[1]
R19_t <- cbind(oil[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
oil_mod <- vars::VAR(R19_t, p=1, type="const")$varresult[1]
R20_t <- cbind(util[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
util_mod <- vars::VAR(R20_t, p=1, type="const")$varresult[1]
R21_t <- cbind(telcm[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
telcm_mod <- vars::VAR(R21_t, p=1, type="const")$varresult[1]
R22_t <- cbind(servs[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
servs_mod <- vars::VAR(R22_t, p=1, type="const")$varresult[1]
R23_t <- cbind(buseq[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
buseq_mod <- vars::VAR(R23_t, p=1, type="const")$varresult[1]
R24_t <- cbind(paper[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
paper_mod <- vars::VAR(R24_t, p=1, type="const")$varresult[1]
R25_t <- cbind(trans[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
trans_mod <- vars::VAR(R25_t, p=1, type="const")$varresult[1]
R26_t <- cbind(whlsl[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
whlsl_mod <- vars::VAR(R26_t, p=1, type="const")$varresult[1]
R27_t <- cbind(rtail[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
rtail_mod <- vars::VAR(R27_t, p=1, type="const")$varresult[1]
R28_t <- cbind(meals[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
meals_mod <- vars::VAR(R28_t, p=1, type="const")$varresult[1]
R29_t <- cbind(fin[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
fin_mod <- vars::VAR(R29_t, p=1, type="const")$varresult[1]
R30_t <- cbind(other[2:239], market_innovations, TERM_orth, DEF_orth, OIL_orth, DIF_orth)
```

```r
other_mod <- vars::VAR(R30_t, p=1, type="const")$varresult[1]
# Combine all models together
models <- c(food_mod, beer_mod, smoke_mod, games_mod, books_mod, hshld_mod,
            clths_mod, hlth_mod, chems_mod, txtls_mod, cnstr_mod, steel_mod,
            fabpr_mod, elceq_mod, autos_mod, carry_mod, mines_mod, coal_mod,
            oil_mod, util_mod, telcm_mod, servs_mod, buseq_mod, paper_mod,
            trans_mod, whlsl_mod, rtail_mod, meals_mod, fin_mod, other_mod)
```

Next, for the second stage of the FM method, we will use the estimates of the β's of each industry to construct the $\hat{\boldsymbol{X}}$ matrix in *Equation 1.5*, and the asset returns vector $\boldsymbol{R}_t = (R_{1t}, ..., R_{Nt})^T$.

```r
# Construction of the X_hat matrix in Equation 1.5
X_hat <- NULL
for (i in 1:30) {
  X_hat <- rbind(X_hat, models[i]$X$coefficients[2:6])
}
X_hat <- cbind(rep(1,30), X_hat)
# Construction of the asset return vector.
R_t <- c(R1_t[238,1], R2_t[238,1], R3_t[238,1], R4_t[238,1], R5_t[238,1], R6_t[238,1],
                   R7_t[238,1], R8_t[238,1], R9_t[238,1], R10_t[238,1], R11_t[238,1], R12_t[238,1],
                   R13_t[238,1], R14_t[238,1], R15_t[238,1], R16_t[238,1], R17_t[238,1], R18_t[238,1],
                   R19_t[238,1], R20_t[238,1], R21_t[238,1], R22_t[238,1], R23_t[238,1], R24_t[238,1],
                   R25_t[238,1], R26_t[238,1], R27_t[238,1], R28_t[238,1], R29_t[238,1], R30_t[238,1])
```

Now, we regress asset returns $\boldsymbol{R}_t$ on $\hat{\boldsymbol{X}}$ to construct the factor mimicking portfolio in the last period (i.e. at 2018/11).

```r
# Utilize Equation 1.5 to construct factor mimicking portfolios.
allocating_weights <- solve(t(X_hat)%*%X_hat)%*%t(X_hat)
portfolios <- ts(allocating_weights*R_t) # Change into a R time series.
portfolios
```

```
## Time Series:
## Start = 1
## End = 6
## Frequency = 1
##           [,1]         [,2]        [,3]         [,4]         [,5]
## 1  0.381077799 -0.541185100 -0.03869219  0.090037578 -0.094084619
## 2 -0.333344336  0.296911188 -0.10757280  0.005757931 -0.002860020
## 3  0.022857090  0.033040613 -0.06669939  0.007366378  0.003837104
## 4 -0.003924456 -0.005701339 -0.09489572 -0.006613893 -0.011235996
## 5  0.015784356  0.010540776  0.02573860  0.024311927  0.015427764
## 6 -0.005161163 -0.003483032  0.01735749 -0.006243508  0.042647377
##            [,6]         [,7]         [,8]         [,9]        [,10]
## 1  0.0098952057 -0.052377468  0.011446088 -0.012860429 -0.043893075
## 2  0.0571052895  0.038105856 -0.181189020 -0.006430265 -0.004794283
## 3 -0.0119836937  0.004071178 -0.006744025 -0.003369651  0.024260035
## 4  0.0013174708 -0.025876253  0.017816147  0.012274571  0.001724678
## 5 -0.0008790722 -0.009808056  0.056469155 -0.031943595 -0.040012455
## 6  0.0078196977 -0.005292931  0.012274875 -0.002574369  0.010329016
##           [,11]        [,12]         [,13]         [,14]        [,15]
## 1 -0.0378758092  0.132651259  0.0002337370 -0.0174921836 -0.126292736
```

```
## 2  0.0669856283 -0.137392090 -0.2317844917 -0.0071713230 -0.007944451
## 3  0.0036035255  0.017653769  0.0112262142 -0.0117475423  0.003480961
## 4 -0.0008559641 -0.003731025 -0.0136840484  0.0060381440 -0.001304375
## 5  0.0045466313 -0.038849592 -0.0005390964 -0.0069597749 -0.026048593
## 6  0.0010319073  0.011265490 -0.0084115485  0.0009902149  0.020299209
##          [,16]         [,17]        [,18]       [,19]        [,20]
## 1  0.159269973 -0.1008650534 -0.066139819 -0.26942612  0.179588410
## 2 -0.050567095  0.0917206506  1.108117488 -0.01452233  0.008341744
## 3 -0.011886444 -0.0059593253 -0.064367723  0.00591480  0.053016940
## 4  0.005712261 -0.0254888104  0.054301801 -0.01290003 -0.002082093
## 5 -0.003735399 -0.0105940812 -0.007399235 -0.03679434 -0.008987995
## 6 -0.013962285 -0.0005869977 -0.025094940  0.02189988  0.013118297
##           [,21]        [,22]        [,23]        [,24]        [,25]
## 1 -0.3641233294  0.053033232  0.042344953  0.082740869  0.015610238
## 2  0.3536569758 -0.071889710 -0.749433755 -0.002543734 -0.001147700
## 3  0.0475185088 -0.005019887  0.003663967 -0.002926861 -0.022363070
## 4 -0.0005372496 -0.001187569 -0.041330440  0.018649197  0.005871274
## 5  0.0188406819  0.058315131  0.019348661 -0.027851184 -0.009056581
## 6  0.0195484584  0.007581134 -0.020594241 -0.007170831 -0.026663986
##           [,26]        [,27]        [,28]        [,29]        [,30]
## 1  0.1281189111 -0.019376661 -0.085350884  0.389300223  0.055910414
## 2 -0.0753755743 -0.021176336  0.804507740  0.014870706  0.003092664
## 3  0.0069068588 -0.014897587 -0.003666322  0.006847279  0.030474871
## 4  0.0007933761  0.016521790  0.010103867  0.020286292 -0.005700026
## 5  0.0058955328  0.004795123  0.015545105  0.002749250  0.004260958
## 6 -0.0146567321  0.015320217 -0.023443602 -0.010796935  0.024673553
```

Up to this stage, the portfolio has been successfully constructed. Now what is remaining is the analysis of the performance of the portfolios, in which we will consider Sharpe ratio, mean, standard deviation, and maximum draw-down. In terms of the optimal re-calibration time, we can modify the above code for portfolios for different re-calibration time (for example, 1 month, 1 year, etc. . .), then we choose the one that has the highest Sharpe ratio value to be the optimal among all re-calibrations.

```
library(PerformanceAnalytics)
```

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

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

```
SharpeRatio.annualized(ts(portfolios), Rf=0.02) # Annulized Sharpe Ratio.
```

```
##                                      [,1]      [,2]      [,3]        [,4]
## Annualized Sharpe Ratio (Rf=2%) -0.1296691 -0.361569 -1.165518 -0.03932998
##                                      [,5]       [,6]      [,7]       [,8]
## Annualized Sharpe Ratio (Rf=2%) -0.6195227 -0.4025683 -0.956697 -0.4559836
##                                     [,9]     [,10]      [,11]     [,12]
## Annualized Sharpe Ratio (Rf=2%) -1.89412 -1.055245 -0.4207972 -0.300908
##                                     [,13]     [,14]      [,15]      [,16]
## Annualized Sharpe Ratio (Rf=2%) -0.6898383 -3.069561 -0.8376526 -0.1085845
##                                     [,17]     [,18]      [,19]     [,20]
## Annualized Sharpe Ratio (Rf=2%) -0.4914807 0.1945312 -0.7077408 0.2592447
##                                     [,21]      [,22]      [,23]
##
```

```
## Annualized Sharpe Ratio (Rf=2%) -0.1381822 -0.2992597 -0.7471206
##                                       [,24]     [,25]     [,26]     [,27]
## Annualized Sharpe Ratio (Rf=2%) -0.2715068 -1.615982 -0.198966 -1.335427
##                                     [,28]     [,29]       [,30]
## Annualized Sharpe Ratio (Rf=2%) 0.1963703 0.2690704 -0.06265005
```

```
mean(portfolios)
```

```
## [1] 0.004149213
```

```
sd(portfolios)
```

```
## [1] 0.1482296
```

```
maxDrawdown(portfolios)
```

```
##                       [,1]      [,2]      [,3]        [,4]      [,5]
## Worst Drawdown 0.3333443 0.5411851 0.2753047 0.006613893 0.1033981
##                        [,6]       [,7]      [,8]       [,9]      [,10]
## Worst Drawdown 0.01198369 0.05237747 0.1867111 0.04458834 0.06277573
##                       [,11]     [,12]     [,13]      [,14]     [,15]
## Worst Drawdown 0.03787581 0.1594154 0.2406452 0.03692631 0.1539783
##                      [,16]     [,17]      [,18]     [,19]      [,20]
## Worst Drawdown 0.0731422 0.1008651 0.06613982 0.3114234 0.01105137
##                      [,21]      [,22]    [,23]     [,24]      [,25]
## Worst Drawdown 0.3641233 0.07764538 0.759306 0.0348223 0.05260099
##                       [,26]      [,27]      [,28]      [,29]       [,30]
## Worst Drawdown 0.07649444 0.05444223 0.08535088 0.01079694 0.005700026
```

Construction of Factor Momentum Portfolio

In this section, we will refine the factor mimicking portfolios constructed above by applying the time series momentum strategy as we exercised in Assignment 1.

From the factor mimicking portfolios, we will first construct the equally weighted portfolio. The construction is according to the equation in Assignment 1 (See *Equation 4.1* below), and we also show the Sharpe ratio, mean, standard deviation, and maximum draw-down as in the following outputs.

$$R_t^{EW} = \frac{1}{N}\sum_{i=1}^{N} RuleReturn(i,t)$$

```
# Construct the Equally-weighted portfolio from the factor mimicking portfolios
EW <- NULL
for (i in 2:6) {
  EW <- cbind(EW, (1/30)*sum(portfolios[i,]))
}
EW <- as.vector(EW); EW
```

```
## [1]  0.0280678182  0.0018036191 -0.0028546139  0.0007703533  0.0017339904
```

```
# Performance analytics
SharpeRatio(ts(EW), Rf=0.02)
```

```
##                                      [,1]
## StdDev Sharpe (Rf=2%, p=95%): -1.1245130
## VaR Sharpe (Rf=2%, p=95%):    -1.8570039
## ES Sharpe (Rf=2%, p=95%):     -0.7253314
```

```
mean(EW)
```

```
## [1] 0.005904233
```

```
sd(EW)
```

```
## [1] 0.012535
```

```
maxDrawdown(EW)
```

```
## [1] 0.002854614
```

Now we move forward to consider the Risk-Parity portfolio, where we also refer back to the equation in Assignment 1 (see *Equation 4.2* below), the key performance analytics statistics are also reported.

$$R_t^{RP} = \sum_{i=1}^{N} \frac{\sigma_i^{-1}}{\sum_{j=1}^{N}\sigma_j^{-1}} \cdot RuleReturn(i,t)$$

```
var_sum <- 0
for (i in 2:6) {
  var_sum <- var_sum+(var(portfolios[i,]))^(-1)
}
# Construct the risk-parity portfolio from the factor mimicking portfolios
RP <- NULL
for (i in 2:6) {
  RP <- cbind(RP, sum((((var(portfolios[i,]))^(-1))/var_sum)*portfolios[i,]))
```

```
}
RP <- as.vector(RP); RP
```

```
## [1]  0.0009783866  0.0097207036 -0.0168229725  0.0045795755  0.0220868034
```

```
# Performance analytics
SharpeRatio(ts(RP), Rf=0.02)
```

```
##                                       [,1]
## StdDev Sharpe (Rf=2%, p=95%): -1.1216939
## VaR Sharpe (Rf=2%, p=95%):    -0.8816050
## ES Sharpe (Rf=2%, p=95%):     -0.7258735
```

```
mean(RP)
```

```
## [1] 0.004108499
```

```
sd(RP)
```

```
## [1] 0.01416741
```

```
maxDrawdown(RP)
```

```
## [1] 0.01682297
```

Now, to construct the TSMOM portfolio with lag $h = 12$, we need to apply $VAR(12)$ approach to each of the 30 industries to fit the new models. This is done by the below code.

```
# VAR(12) approach for each of the 30 industry portfolios
food_mod2 <- vars::VAR(R1_t, p=12, type="const")$varresult[1]
beer_mod2 <- vars::VAR(R2_t, p=12, type="const")$varresult[1]
smoke_mod2 <- vars::VAR(R3_t, p=12, type="const")$varresult[1]
games_mod2 <- vars::VAR(R4_t, p=12, type="const")$varresult[1]
books_mod2 <- vars::VAR(R5_t, p=12, type="const")$varresult[1]
hshld_mod2 <- vars::VAR(R6_t, p=12, type="const")$varresult[1]
clths_mod2 <- vars::VAR(R7_t, p=12, type="const")$varresult[1]
hlth_mod2 <- vars::VAR(R8_t, p=12, type="const")$varresult[1]
chems_mod2 <- vars::VAR(R9_t, p=12, type="const")$varresult[1]
txtls_mod2 <- vars::VAR(R10_t, p=12, type="const")$varresult[1]
cnstr_mod2 <- vars::VAR(R11_t, p=12, type="const")$varresult[1]
steel_mod2 <- vars::VAR(R12_t, p=12, type="const")$varresult[1]
fabpr_mod2 <- vars::VAR(R13_t, p=12, type="const")$varresult[1]
elceq_mod2 <- vars::VAR(R14_t, p=12, type="const")$varresult[1]
autos_mod2 <- vars::VAR(R15_t, p=12, type="const")$varresult[1]
carry_mod2 <- vars::VAR(R16_t, p=12, type="const")$varresult[1]
mines_mod2 <- vars::VAR(R17_t, p=12, type="const")$varresult[1]
coal_mod2 <- vars::VAR(R18_t, p=12, type="const")$varresult[1]
oil_mod2 <- vars::VAR(R19_t, p=12, type="const")$varresult[1]
util_mod2 <- vars::VAR(R20_t, p=12, type="const")$varresult[1]
telcm_mod2 <- vars::VAR(R21_t, p=12, type="const")$varresult[1]
servs_mod2 <- vars::VAR(R22_t, p=12, type="const")$varresult[1]
buseq_mod2 <- vars::VAR(R23_t, p=12, type="const")$varresult[1]
paper_mod2 <- vars::VAR(R24_t, p=12, type="const")$varresult[1]
trans_mod2 <- vars::VAR(R25_t, p=12, type="const")$varresult[1]
whlsl_mod2 <- vars::VAR(R26_t, p=12, type="const")$varresult[1]
rtail_mod2 <- vars::VAR(R27_t, p=12, type="const")$varresult[1]
```

```
meals_mod2 <- vars::VAR(R28_t, p=12, type="const")$varresult[1]
fin_mod2 <- vars::VAR(R29_t, p=12, type="const")$varresult[1]
other_mod2 <- vars::VAR(R30_t, p=12, type="const")$varresult[1]
# Combine all models together
models2 <- c(food_mod2, beer_mod2, smoke_mod2, games_mod2, books_mod2, hshld_mod2,
             clths_mod2, hlth_mod2, chems_mod2, txtls_mod2, cnstr_mod2, steel_mod2,
             fabpr_mod2, elceq_mod2, autos_mod2, carry_mod2, mines_mod2, coal_mod2,
             oil_mod2, util_mod2, telcm_mod2, servs_mod2, buseq_mod2, paper_mod2,
             trans_mod2, whlsl_mod2, rtail_mod2, meals_mod2, fin_mod2, other_mod2)
```

Then, we refer back to *Equation 1.5* to regress the regress asset returns $\boldsymbol{R}_t$ on $\hat{\boldsymbol{X}}$. The procedure and output is as follows.

```
# Construction of the X_hat matrix in Equation 1.5
X_hat2 <- NULL
for (i in 1:30) {
  X_hat2 <- rbind(X_hat2, models2[i]$X$coefficients[2:6])
}
X_hat2 <- cbind(rep(1,30), X_hat)
R_t2 <- c(R1_t[238,1], R2_t[238,1], R3_t[238,1], R4_t[238,1], R5_t[238,1], R6_t[238,1],
         R7_t[238,1], R8_t[238,1], R9_t[238,1], R10_t[238,1], R11_t[238,1], R12_t[238,1],
         R13_t[238,1], R14_t[238,1], R15_t[238,1], R16_t[238,1], R17_t[238,1], R18_t[238,1],
         R19_t[238,1], R20_t[238,1], R21_t[238,1], R22_t[238,1], R23_t[238,1], R24_t[238,1],
         R25_t[238,1], R26_t[238,1], R27_t[238,1], R28_t[238,1], R29_t[238,1], R30_t[238,1])
# Calculate the portfolios returns from FM method (Equation 1.5)
allocating_weights2 <- solve(t(X_hat)%*%X_hat)%*%t(X_hat)
returns <- ts(allocating_weights*R_t) # Change into a R time series.
returns
```

```
## Time Series:
## Start = 1
## End = 6
## Frequency = 1
##           [,1]         [,2]        [,3]         [,4]         [,5]
## 1  0.381077799 -0.541185100 -0.03869219  0.090037578 -0.094084619
## 2 -0.333344336  0.296911188 -0.10757280  0.005757931 -0.002860020
## 3  0.022857090  0.033040613 -0.06669939  0.007366378  0.003837104
## 4 -0.003924456 -0.005701339 -0.09489572 -0.006613893 -0.011235996
## 5  0.015784356  0.010540776  0.02573860  0.024311927  0.015427764
## 6 -0.005161163 -0.003483032  0.01735749 -0.006243508  0.042647377
##            [,6]         [,7]         [,8]         [,9]        [,10]
## 1  0.0098952057 -0.052377468  0.011446088 -0.012860429 -0.043893075
## 2  0.0571052895  0.038105856 -0.181189020 -0.006430265 -0.004794283
## 3 -0.0119836937  0.004071178 -0.006744025 -0.003369651  0.024260035
## 4  0.0013174708 -0.025876253  0.017816147  0.012274571  0.001724678
## 5 -0.0008790722 -0.009808056  0.056469155 -0.031943595 -0.040012455
## 6  0.0078196977 -0.005292931  0.012274875 -0.002574369  0.010329016
##           [,11]        [,12]         [,13]         [,14]        [,15]
## 1 -0.0378758092  0.132651259  0.0002337370 -0.0174921836 -0.126292736
## 2  0.0669856283 -0.137392090 -0.2317844917 -0.0071713230 -0.007944451
## 3  0.0036035255  0.017653769  0.0112262142 -0.0117475423  0.003480961
## 4 -0.0008559641 -0.003731025 -0.0136840484  0.0060381440 -0.001304375
## 5  0.0045466313 -0.038849592 -0.0005390964 -0.0069597749 -0.026048593
```

```
## 6  0.0010319073  0.011265490 -0.0084115485  0.0009902149  0.020299209
##          [,16]         [,17]        [,18]       [,19]        [,20]
## 1  0.159269973 -0.1008650534 -0.066139819 -0.26942612  0.179588410
## 2 -0.050567095  0.0917206506  1.108117488 -0.01452233  0.008341744
## 3 -0.011886444 -0.0059593253 -0.064367723  0.00591480  0.053016940
## 4  0.005712261 -0.0254888104  0.054301801 -0.01290003 -0.002082093
## 5 -0.003735399 -0.0105940812 -0.007399235 -0.03679434 -0.008987995
## 6 -0.013962285 -0.0005869977 -0.025094940  0.02189988  0.013118297
##           [,21]        [,22]        [,23]        [,24]        [,25]
## 1 -0.3641233294  0.053033232  0.042344953  0.082740869  0.015610238
## 2  0.3536569758 -0.071889710 -0.749433755 -0.002543734 -0.001147700
## 3  0.0475185088 -0.005019887  0.003663967 -0.002926861 -0.022363070
## 4 -0.0005372496 -0.001187569 -0.041330440  0.018649197  0.005871274
## 5  0.0188406819  0.058315131  0.019348661 -0.027851184 -0.009056581
## 6  0.0195484584  0.007581134 -0.020594241 -0.007170831 -0.026663986
##           [,26]        [,27]        [,28]        [,29]        [,30]
## 1  0.1281189111 -0.019376661 -0.085350884  0.389300223  0.055910414
## 2 -0.0753755743 -0.021176336  0.804507740  0.014870706  0.003092664
## 3  0.0069068588 -0.014897587 -0.003666322  0.006847279  0.030474871
## 4  0.0007933761  0.016521790  0.010103867  0.020286292 -0.005700026
## 5  0.0058955328  0.004795123  0.015545105  0.002749250  0.004260958
## 6 -0.0146567321  0.015320217 -0.023443602 -0.010796935  0.024673553
```

Next, we consult the below equation for constructing TSMOM portfolio (*Equation 4.3*) with $h = 12$ for each industry, and then use the following R code to construct the portfolio. Finally, we analyze the performance of this TSMOM portfolio as we did with previous portfolios.

$$R_{t,t+1}^{TSMOM} = \frac{1}{N}\sum_{i=1}^{N} sign(r_{i,t-h:t}) \cdot \frac{40\%}{\sigma_{i,t}} r_{s,t:t+1}$$

```r
# TSMOM construction
TSMOM <- NULL
for (i in 2:6) {
  TSMOM <- cbind(TSMOM, (1/30)*sum(sign(returns[i,])*(0.4/var(returns[i,]))*portfolios[i,]))
}
TSMOM <- as.vector(TSMOM); TSMOM
```

```
## [1]  0.6504847 10.7147992  9.7027391 12.3818661 19.5926746
```

```r
# Performance analytics
SharpeRatio.annualized(ts(TSMOM), Rf=0.02)
```

```
##                                   [,1]
## Annualized Sharpe Ratio (Rf=2%) 1.16598
```

```r
mean(TSMOM)
```

```
## [1] 10.60851
```

```r
sd(TSMOM)
```

```
## [1] 6.779638
```

```r
maxDrawdown(TSMOM)
```

```
## [1] 0
```