Financial Backtesting Tutorial

Shiwen Shen

March 31, 2016

1 Introduction

This is a short tutorial leading you to the world of financial backtesting using R. Generally speaking, you need two major components to evalute the performance of your potential trading strategy, say a valid strategy and a historical time series data, which is hopefully long enough to display some insights. In this tutorial, I would like to use a common strategy, Global Minimum Variance Portfolio (GMPV), as an example to illustrate how to use R to conduct backtesting with different baseline settings.

2 Load Working Space

I assume you have already installed R in your personal computer. (If not, please visit `https://www.r-project.org/` to find the version works for your Windows, Mac, or UNIX platform.) Open the R program, and the first thing you see is the R Console, in which codes are excuted. Press Ctrl+N to open a new R Editor. You can write anything you want in the editor, however, only valid R code can be recognized by R and feed you with proper results. Now copy and paste the following R code into your editor, and select the code you just pasted and press F5.

```
load(url("http://people.stat.sc.edu/sshen/events/backtesting/backtesting_pack.RData"))
```

"F5" is the button we use to tell R to run the selected code written in the editor. The above code is to let R load the working space that I created for backtesting.

Now, let's see what is included in the working space by running the following R code by running **ls()**:

```
ls()
```

and you will see the following feedback in the console.

```
[1] "backtest" "GMVP"     "PerfInd"  "SHARPE"   "testdata"
```

R informs you that there are 5 objects in this working space. Objects can be anything including data, function, etc. In this specific example, the first four objects **backtest**, **GMVP**, **PerfInd**, and **SHARPE** are functions, and **testdata** is the historical time series data we are going to use to test the GMVP strategy.

3 GMVP Strategy

The GMVP strategy provides us with the portfolio weights minimizing the historical portfolio variance (risk). Let's copy and paste the following R code into the editor, and pree F5 to run it.

```
args(GMVP)
```

The R command **args** displays the argument names and corresponding default values of a function.

```
function (data, lower = 0, upper = 1)
```

args(GMVP) shows the arguments for **GMVP** function are data, lower bound of the portfolio weights, and upper bound of the portfolio weights. I set the default

values of lower bounds to be 0 and the ones of upper bounds to be 1 to mimic the idea of "long only". You can assign any lower and upper bound as you want in using the GMVP function, however, if you forget to assign something, R will use zeros and ones. Now, let's play with the GMVP function by running the following code,

```
GMVP(data=testdata, lower=0, upper=1)
```

The output of the GMVP function is shown in the following shaded box. Note that the dataset we are using is pre-stored in the working space, named as "testdata".

```
$name
 [1] "Brazil"       "China"        "India"        "Indonesia"
 [5] "Korea"        "Malaysia"     "Mexico"       "Poland"
 [9] "Russia"       "South.Africa" "Taiwan"       "Thailand"
[13] "Turkey"

$gmvp
 [1] 0.0000 0.0272 0.0350 0.0000 0.0000 0.5040 0.2373 0.0000
 [9] 0.0000 0.1660 0.0305 0.0000 0.0000

$comb
      name           gmvp
 [1,] "Brazil"       "0"
 [2,] "China"        "0.0272"
 [3,] "India"        "0.035"
 [4,] "Indonesia"    "0"
 [5,] "Korea"        "0"
 [6,] "Malaysia"     "0.504"
 [7,] "Mexico"       "0.2373"
 [8,] "Poland"       "0"
 [9,] "Russia"       "0"
[10,] "South.Africa" "0.166"
```

```
[11,] "Taiwan"       "0.0305"
[12,] "Thailand"     "0"
[13,] "Turkey"       "0"
```

There are three objects in the GMVP function output, **name**, **gmvp**, and **comb**. **Name** shows what assets are included in the portfolio, **gmvp** is the estimated weight for each asset, and **comb** is a combined display of names and weights. As you can see, this is the results for "long only", e.g. **lower=0, upper=1**. You can achieve different GMVP weights by using different lower and upper bound settings, e.g.

```
GMVP(data=testdata, lower=-1, upper=1)
GMVP(data=testdata, lower=-0.5, upper=0.5)
GMVP(data=testdata, lower=0, upper=0.5)
```

4 Backtesting with GMVP

"backtest" is the function we use to conduct backtesting. Let's check what arguments are needed here:

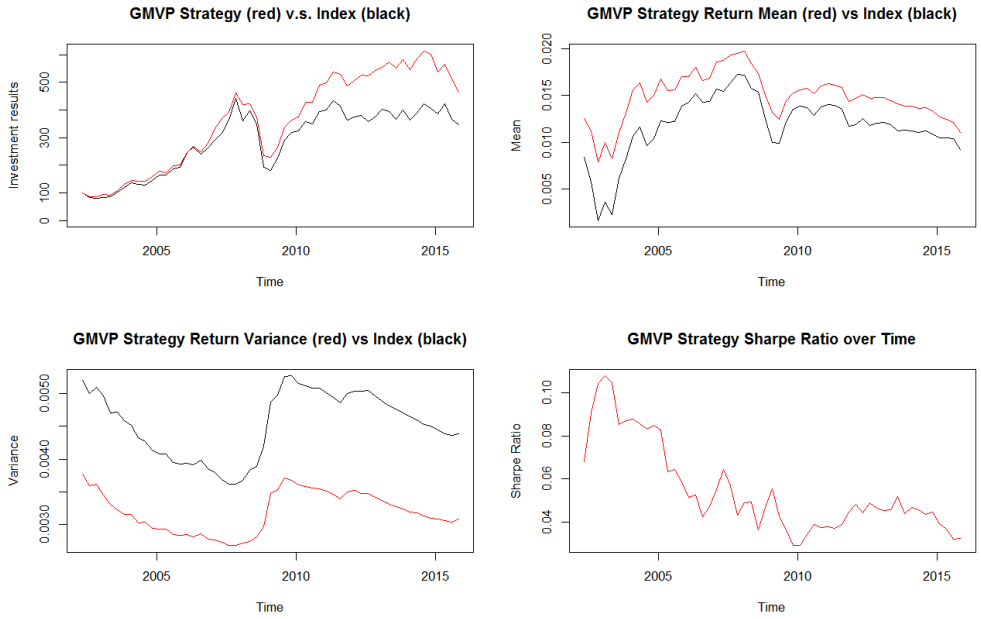
```
args(backtest)

function (data, initial.time, initial.money = 100, rebalance = 1,
    method = GMVP, lower = 0, upper = 1, rf = 0)
```

There are eight arguments in "backtest" function and six of them have corresponding default values, in which **initial.time** is the starting time point of backtesting; **initial.money** is the money you have at the initial.time; **rebalance** the time gap between rebalancing the portfolio; **method** is the trading strategy you use with default to be GMVP; **lower** and **upper** are weight bounds, and **rf** is the risk free rate.

Let's run the backtest function to see its output

```
result <- backtest(data=testdata, initial.time=41, initial.money=100,
          rebalance=3, method=GMVP, lower=0, upper=1, rf=0)
result
```

R automatically displays the difference between the GMVP strategy (red solid line) with benchmark index (black solid line). It shows that GMVP started to beat the index after 2010.



If you go back to the R Console, you can see a detailed output including portfolio weights in each rebalancing time point. Moreover, you can play with different baseline settings to compare the backtesting results, e.g.

```
## remove long only assumption
result <- backtest(data=testdata, initial.time=41, initial.money=100,
          rebalance=3, method=GMVP, lower=-10, upper=10, rf=0)
```

```
## remove long only assumption and rebalance every month
result <- backtest(data=testdata, initial.time=41, initial.money=100,
         rebalance=1, method=GMVP, lower=-10, upper=10, rf=0)

## long only with upper bound=0.5, and rebalance every month
result <- backtest(data=testdata, initial.time=41, initial.money=100,
         rebalance=1, method=GMVP, lower=0, upper=0.5, rf=0)
```

5 More Infomation

So far I have composed strategy functions for GMVP, maximizing sharpe ratio (SHARPE), and maximizing Performance Index (PerfInd). In order to use different strategies, you just need to replace **method=GMVP** argument in the backtest function with **method=SHARPE** or **method=PerfInd**. However, you are encouraged to build your own trading strategies and backtest them, which perhaps may lead to your first one million.

Note: maximizing Performance Index method is proposed by Dr. Michael Stutzer. You can find the his paper in the Reference Papers section in the webpage.