Minimum Variance Portfolio

Philipp Jung

January 17, 2019

Minimum Variance Portfolio

Data

The data being used is provided by Kenneth French. The well known dataset '10 Industry Portfolios' contains 10 portfolios of different US-companies.

Loading the data

Read 'average value weighted returns (monthly)' from line 12 to line 1121

```
return_matrix = read.csv('10_Industry_Portfolios.CSV', header=TRUE, sep=',', skip = 11, nrows = 1109)
head(return_matrix)
```

```
##        X NoDur Durbl Manuf Enrgy HiTec Telcm Shops  Hlth Utils Other
## 1 192607  1.45 15.55  4.69 -1.18  2.90  0.83  0.11  1.77  7.04  2.16
## 2 192608  3.97  3.68  2.81  3.47  2.66  2.17 -0.71  4.25 -1.69  4.38
## 3 192609  1.14  4.80  1.15 -3.39 -0.38  2.41  0.21  0.69  2.04  0.29
## 4 192610 -1.24 -8.23 -3.63 -0.78 -4.58 -0.11 -2.29 -0.57 -2.63 -2.85
## 5 192611  5.21 -0.19  4.10  0.01  4.71  1.63  6.43  5.42  3.71  2.11
## 6 192612  0.82  9.89  3.74  2.82 -0.02  1.99  0.62  0.11 -0.17  3.40
```

Calculate returns

Returns are generally calculated logarithmically, such that, considering a time-series of prices $p_1, p_2, ..., p_T$, the return at time i+1 is given by

$$r_{i+1} = ln(p_{i+1}/p_i).$$

Just a side-note as we thankfully already work with a return_matrix.

Interpret the problem as a system of linear equations

As written in DeMiguel, we choose the weights according to the portfolio that minimizes the variance of return, e.g.

$$\min_{w_t} w_t^{\perp} \Sigma_t w_t$$

under the restriction that $1_N^{\perp} w_t \stackrel{!}{=} 1$.

$$\Sigma_t w_t \stackrel{!}{=} 1$$

, where Σ_t is the covariance matrix of the excess-returns R_t at time t. If there are N risky assets considered, 1 is thought of as the N-dimensional vector containing 1's.

Table 6.1 Details for 10 industry portfolios

1	NoDur	Consumer NonDurables: Food, Tobacco, Textiles, Apparel, Leather, Toys
2	Durbl	Consumer Durables: Cars, TV's, Furniture, Household Appliances
3	Manuf	Manufacturing: Machinery, Trucks, Planes, Chemicals, Off Furn, Paper, Com Printing
4	Enrgy	Oil, Gas, and Coal Extraction and Products
5	HiTec	Business Equipment: Computers, Software, and Electronic Equipment
6	Telcm	Telephone and Television Transmission
7	Shops	Wholesale, Retail, and Some Services (Laundries, Repair Shops)
8	Hlth	Healthcare, Medical Equipment, and Drugs
9	Utils	Utilities
10	Other	Other: Mines, Constr, BldMt, Trans, Hotels, Bus Serv, Entertainment, Finance

Figure 1: Explanation 10 Industry Portfolios

In the following section, the restriction $\min_{w_t}$ shall be investigated for one fixed time-period T, such that weights are not time-dependent anymore. Considering 3 risky assets and $w_1, w_2, \ldots, w_N$ 3 weights, the minimal variance restraind can be interpreted as follows:

$$\begin{aligned}\min_{w_1,w_2,w_3} \sigma^2_{w_1,w_2,w_3} =& w_1^2\sigma_1^2 + w_2^2\sigma_2^2 + w_3^2\sigma_3^2 + \\ & 2w_1w_2\sigma_{12} + 2w_1w_3\sigma_{13} + 2w_2w_3\sigma_{23}\end{aligned}$$

The Lagrange-function of this problem is

$$\begin{aligned}L(w_1,w_2,w_3,\lambda) =& w_1^2\sigma_1^2 + w_2^2\sigma_2^2 + w_3^2\sigma_3^2 + \\ & 2w_1w_2\sigma_{12} + 2w_1w_3\sigma_{13} + 2w_2w_3\sigma_{23} \\ & + \lambda(w_1 + w_2 + w_3 - 1)\end{aligned}$$

Now, the component-wise deviates can be found and be restructured to form a linear system of equations.

Computing the Covariance Matrix

As we saw in the section above, to minimize the overall variance, we first need to compute the covariance matrix. This can be achieved by applying the cov() function to the rows (t-th row for time t) of return_matrix.

```
nA = dim(return_matrix[,-1])[2] # number of assets
cov_return_matrix = cov(return_matrix[,-1]) # dates are deleted
```

All positive values in the cov_return_matrix, meaning there is at least some linear relation between all the values.

Above equation is of form

$$A \cdot x = b$$

, thus a system of linear differential equations. Since $A = \Sigma$ and $b = 1$ are known, we can solve this system to obtain $x = R_t$ the

```
b = vector(length=nA) + 1
x = solve(cov_return_matrix, b)
```

```r
x = x/sum(x)
x
```

```
##       NoDur       Durbl       Manuf       Enrgy       HiTec       Telcm
##  0.75270484 -0.05959476 -0.13776060  0.21510686 -0.09869360  0.53630598
##       Shops        Hlth       Utils       Other
## -0.05085601  0.07921599  0.08130649 -0.31773519
```

Note that there are *negative weights*. These are interpreted as short sales.

Computing the global minimum-variance portfolio tutorial

I first wanted to do this with datacamp, but then I came across this tutorial (the first hit on google really). So I just want to follow the steps described there to get an understanding for the method.

Generate example-data

As an example dataset, we create artificial returns with mean zero (the default with rnorm) and volatility 5%. (This order of magnitude would be reasonable for monthly equity returns.) These returns are stored in a matrix mData.

```r
n0 = 100  ## number of observations
nA = 10   ## number of assets
artificial_return_matrix = array(rnorm(n0 * nA, sd = 0.05),
               dim = c(n0, nA))
head(artificial_return_matrix)
```

```
##              [,1]         [,2]         [,3]         [,4]        [,5]
## [1,]  0.048543223  0.025549712 -0.036241397  0.001558986 -0.07788848
## [2,]  0.021107401  0.039900960  0.023441691  0.070329707  0.03847394
## [3,]  0.052311613  0.055825823  0.024568102 -0.080399794  0.08952787
## [4,]  0.008082707 -0.031442965 -0.016150986 -0.013751655  0.13106706
## [5,]  0.002605631 -0.004389381  0.006598466  0.057152618  0.02102842
## [6,] -0.051311087  0.023836069 -0.004255117 -0.079445208  0.10059357
##                [,6]          [,7]         [,8]         [,9]        [,10]
## [1,]  0.0046498427  0.0532427082 -0.056030729  0.015003843  0.055309495
## [2,] -0.0947051882  0.0109354067 -0.080786897 -0.067900469  0.054068695
## [3,] -0.0051344683 -0.0008632606 -0.010445604 -0.003025618  0.189710998
## [4,] -0.0002558516 -0.0717658346 -0.046043386  0.065030882  0.006507901
## [5,]  0.0438108583  0.0080630880 -0.003804028 -0.024953157  0.006969991
## [6,] -0.0093112238 -0.0179104950 -0.028455016 -0.197574754 -0.004394137
```

The following section describes different approaches to gaining the weights.

Approach according to Washington State

This approach can be found in this script. The linear system of equations you come up with once you work on the Langrangian can be written as

$$\begin{bmatrix} \sigma^2_{NoDur} & \sigma_{NoDur,Durbl} & \dots & \sigma_{NoDur,Other} & 1 \\ \sigma_{NoDur,Durbl} & \sigma^2_{Durbl} & \dots & \vdots & 1 \\ \vdots & & \ddots & \vdots & \vdots \\ \sigma_{NoDur,Other} & \dots & \dots & \sigma^2_{Other} & 1 \\ 1 & 1 & \dots & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} w_{NoDur} \\ w_{Durbl} \\ \vdots \\ w_{Other} \\ \lambda \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

This is just the same as

$$\begin{bmatrix} 2\Sigma & 1 \\ 1^\intercal & 0 \end{bmatrix} \cdot \begin{bmatrix} w \\ \lambda \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

which is a system of linear equations in matrix notation. ## Solving a system of linear equations To solve the above system, we can use the solve()-Function implemented in R.

```
ATop = cbind(2*cov_return_matrix, rep(1, nA))
ABotVec = c(rep(1, nA), 0)
A = rbind(ATop, ABotVec)
bVec = c(rep(0, nA), 1)
z.m.mat = solve(A) %*% bVec
mVec = z.m.mat[1:nA, 1]
mVec
```

```
##       NoDur       Durbl       Manuf       Enrgy       HiTec       Telcm
##  0.75270484 -0.05959476 -0.13776060  0.21510686 -0.09869360  0.53630598
##       Shops        Hlth       Utils       Other
## -0.05085601  0.07921599  0.08130649 -0.31773519
```

which is exactly the same as if you dropped the whole λ - equation.