Application of Matlab for Finance

September 22, 201

Today's Class

- Portfolio Optimization
-
- Optimization with constraints
- Construct the efficient frontier

Harvard Management Company Case Study

- Today you are the manager of Harvard Management Company, who manages and invests the endowment funds for the Harvard University.
-
- File `HMC data.xls` reports relevant info of 12 asset classes: mean, standard deviation and variance-covariance matrix.

- You'd like to visually explore the data to understand the relationship between risk (standard deviation) and return among these assets.

Read Data & Scatter Plot


```
[num, txt] = xlsread('HMC data.xls');
% read mean, std deviation and covariance matrix
avg_ret = num(:,1);
s
c
```



```
figure
scatter(stdev, avg_ret, 40,'filled'); % 4
    with filled color
title('Scatter Plot');
xlabel('Risk (%)');
ylabel('Return (%)');
xlim([0 25]); % x-axis limits
ylim([0 10])
```

Scatter Plot

The Investment Decision

- The scatter plot confirms the traditional finance mantra
  - Higher risk, higher return.
-
- i
- Your objective is to find the optimal investment $w = [\omega_1, \omega_2, \ldots, \omega_{12}]'$ so that your portfoli... k level) is the **minimum**.
- Meanwhile, sum of your weights equals to 1 as you invest **all** of your capital without borrowing.

Compute Portfolio Variance

- $w = [\omega_1, \omega_2, ..., \omega_{12}]'$ is a column vector (12x1) representing weights of capital allocated to different assets.

- Σ is the variance-covariance matrix with size (12x12)
-

(1)

- That is, portfolio σ_p^2 is a function of the weig
- **The optimization problem is to find the opti combination** w^* **that minimizes the fun th a** **constraint** **that sum of the portfolio weights**

- We term such optimal portfolio as the **Global Minimum Variance Portfolio (GMVP).**

Global Minimum Variance Portfolio (GMVP)

- Formulate such portfolio optimization problem as follows:

$$\min \quad \sigma_p^2 = w'\Sigma w$$

$$\mathbf{1}' * w = [1\ 1, ..., 1] * \begin{bmatrix} \omega_2 \\ \vdots \\ \omega_{12} \end{bmatrix} = \omega_1 +$$

- We want to find the optimal w^* and the corresponding minimized $\sigma_p^{2*} = w^{*'}\Sigma w^*$

Optimization: `fmincon`

- `x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options)`

$$A \cdot x \leq b$$

$$(\ ) \quad Aeq \quad x = beq$$

-
- `x0`: the starting point value of input x
- `A,b, Aeq, beq, lb, ub`: elements to d

- `nonlcon`: the indicator for nonlinear constra
- If no specific value used, use `[]` to ski
- `options`: define for iterations, limits, increments for the optimization process as well as screen display.

Optimisation: `fmincon`

- `x1 = fmincon(@myfun(x),x0,...)` finds the optimal value `x1` that minimizes the result of function `myfun`, with starting point at

- t

- `[x1, fval1] = fmincon(@(x)myfun(x,y` as above, but with two outputs

  - `x1` is the optimal value of `x` that minimizes the f
  - `fval1` is the minimized value of `m`
  - `fval1 = myfun(x1,y)`

Optimisation: `fmincon`

- `x1 = fmincon(@myfun(x),x0,A,b)` specifies the 1st type of constraint **A*x b**
-

- f
  constraint `Aeq*x = beq` only.
- `x1 = fmincon(@myfun(x),x0,A,b,Aeq`
  types of constraints with additional lower and u , so that the solution is always in the range

Step 1: Define the Objective Function

- Going back to our global minimum variance portfolio question

$$\min \quad \sigma_p^2 = w'\Sigma w$$
$$s.t. \quad \mathbf{1}'w = 1$$

- olio atrix


```
function [pvar] = compute_pvar(w, cov_

% w is a column vector
pvar = w' * cov_mat * w;

end
```

Step 2: Define the Constraint

$$\min \quad \sigma_p^2 = w'\Sigma w$$
$$s.t. \quad \mathbf{1}'w = 1$$

-

$$\mathbf{1}'w =$$

$$\begin{cases} A \cdot x \leq b \\ Aeq \cdot x \\ lb \leq x \leq ub \end{cases}$$

- `Aeq` $= \mathbf{1}' =$ `[1,1,..,1]`
- `beq = 1`

Step 3: Optimize

```
w1 = fmincon(@(w)compute_pvar(w, cov_mat), w0, [], ...
[], Aeq, beq,..)
```



```
% constraint: sum of weights is 1
N = length(avg_ret);
A
b
w
% s
o

% option 1: return optimal weight vector w1
w1 = fmincon(@(w)compute_pvar(w, cov_ma
    Aeq, beq, [], [], [], options);
% option 2: return both the optiomal weight w1
%           and the minimized portfolio variance pvar1
[w1,pvar1] = fmincon(@(w)compute_pvar(w, cov_mat), w0, ...
    [], [], Aeq, beq, [], [], [], options);
disp(['The variance of GMVP is ', num2str(pvar1)]);
```

With Further Constraints

- Add two more constraints to the portfolio variance optimization:
  - NO short-selling is allowed. That is $w_i \geq 0 \quad \forall i \in [1, N]$.

$$\begin{aligned} \min \quad & \sigma_P^2 = w'\Sigma w \\ s.t. \quad & 1'w = 1 \\ & 0 \leq w_i \leq 0.2 \end{aligned}$$

With Further Constraints

- Same objective function `compute_pvar`
- Define the constraints

$$
\begin{cases}
Aeq \cdot x = beq \\
lb \leq x \leq ub
\end{cases}
$$

- `Aeq`$= \mathbf{1}' =$ `[1,1,..,1]`, and `beq = 1`
- `lb = [0,0,..,0]` and `ub =[0.2,0.2,..,0.2]`

With Further Constraints


```
%% With Further constraints
% weights with ceiling (20%) and no short sell
A
b
l
u

w0 = ones(N,1)*(1/N); % initial weight
w2 = fmincon(@(w)compute_pvar(w, cov_ma
    Aeq, beq, lb, ub, [], options);
[w2,pvar2] = fmincon(@(w)compute_pvar
    [], [], Aeq, beq, lb, ub, [], options);
disp(['The minimised portfolio variance is ', ...
    num2str(pvar2)]);
```

With a Target Return

- Minimize portfolio variance for a given target return 10%

$$\min \quad w'\Sigma w$$

$$\begin{cases} A \cdot x \leq b \\ Aeq \cdot x \\ lb \leq x \leq ub \end{cases}$$

- `Aeq` $= \begin{bmatrix} \mathbf{1}' \\ \mathbf{r}' \end{bmatrix}$, and `beq` $= \begin{bmatrix} 1 \\ 10 \end{bmatrix}$

With a Target Return


```
%% With a target return of 10%
A
b

w
w
    Aeq, beq, [], [], [], options);
[w3,pvar3] = fmincon(@(w)compute_pvar
    [], [], Aeq, beq, [], [], [], options);
disp(['The minimised portfolio variance i
    num2str(pvar3)]);
```

Maximize Sharpe Ratio

- The portfolio Sharpe Ratio is calculated as

$$\text{Sharpe Ratio} = \frac{r_p - r_f}{\sigma_p}$$

$$r_p = w'\mathbf{r} \qquad \sigma_p = \sqrt{w'\Sigma w}$$

- Step 1: Define the function

```
function [p_sharpe] = compute_sharp
    cov_mat)
% w and avg_ret are column vectors
pvar = w' * cov_mat * w;
pret = w' * avg_ret;
% assume risk free rate = 3% annually
% 3 as return in percentage
p_sharpe= (pret - 3) / sqrt(pvar);
end
```

Maximize Sharpe Ratio

- Maximise the Sharpe ratio allows a relative optimization
  - For the same level of risk σ_p, max Sharpe = max return r_p
  - For the same level of return r_p, max Sharpe = min risk σ_p
-
-

$$s.t. \quad \mathbf{1}'w$$

which is equivalent to the following minimizati

$$\begin{aligned} \min \quad & -SR = -f(w) \\ s.t. \quad & \mathbf{1}'w = 1 \end{aligned}$$

Maximize Sharpe Ratio

- Put a negative sign – on the `compute_sharpe` objective function: maximization problem into a minimization one:


```
%% Maximize Sharpe ratio
Aeq = ones(1,N);
```



```
    optimization problem
w4 = fmincon(@(w)-compute_sharpe(
    cov_mat), w0, [], [], Aeq, beq, [], [], [
    options);
[w4,fval4] = fmincon(@(w)-compute
    cov_mat), w0, [], [], Aeq, beq, [], [], [
    options);
% fval4 = -sr1 as the fun = -compute_sharp
sr = - fval4;
disp(['The maximised portfolio Sharpe ratio is ', ...
    num2str(sr)]);
```

Construct Efficient Frontier

- The efficient frontier is the best return vs. risk investment combinations that one can obtain via portfolio diversification on risky assets.

- All assets that lie on the frontier are the most efficient.
-
-

for a range of target returns.

- We **exclude Cash** since it is the risk-free ass

- Loop through all target return levels, find the opti weights that minimize the portfolio risk with res return.
- With the optimal weights, calculate the portfolio variance and plot against the target return.

Construct Efficient Frontier


```
target_ret = 3:0.2:15;
M = length(target_ret);
p
w

% M
f
  Aeq = [ones(1,N-1); avg_ret(1:end-1)'];
  beq = [1; target_ret(i)];
  w_opt = fmincon(@(w) compute_pvar(w, c
      1:end-1)), w0, [], [], Aeq, beq, [], [], [])
  pstdevs(i)  = sqrt(compute_pvar(w_opt
      cov_mat(1:end-1, 1:end-1)));
end
```

Construct Efficient Frontier

- Plot the efficient frontier


```
scatter(stdev(1:end-1), avg_ret
title('Efficient Frontier')
xlabel('Risk (%)');
ylabel('Return (%)')
```

Construct Efficient Frontier

TakeAway

- After today's class, you will be able to finish the Coursework
-
- and standard deviation in the CW are not the same exercises'. Write the codes accordingly.
- Do not forget to assume your risk free rate.