Project 3 – Topics in Data Science ECE 592

Name: Alhiet Orbegoso
ID: 200322491

For the development of the project it has been used MATLAB.

4. Toy Example

For the solution of this problem, it has been used 3. One function calculates the profit during all the periods, the second function calculates the optimal proportion investment using a gradient decent algorithm. The last function calculates the gradient of the profit function.

The functions are detailed below:

Function "profit"

The function returns the profit for a given proportion investment vector and assets value in logarithm scale.

Inputs:

- **h**: It is the investment proportion. It is a vector in which element represents the proportion invested in each stock. The sum of its elements must be equal to 1.
- **Assets_value**: It is the matrix representing the value of all stocks for all periods. The rows are the value for each period and the columns are the stocks.
- **Assets**: It a vector representing the assets that will be used for the calculation of the final profit. The length of is vector must be equal to the length of the investment proportion vector.

Output: It is the profit ratio is logarithm scale. To get normal scale the value should be the powered to the natural exponent.

Function "grad_profit"

Calculates the gradient of a given proportion vector, Assets_value matrix and Assets vector.

The following formula has been used as gradient:

$$\nabla P = \left[\sum_{n=1}^{N} \frac{X_n^1 - X_n^D}{\sum_{d=1}^{D} X_n^d h^d}, \sum_{n=1}^{N} \frac{X_n^2 - X_n^D}{\sum_{d=1}^{D} X_n^d h^d}, \sum_{n=1}^{N} \frac{X_n^3 - X_n^D}{\sum_{d=1}^{D} X_n^d h^d}, \dots\right]$$

The iteration is given by:

$$h_{k+1}^d = h_k^d + \alpha . \nabla P$$

Where: α is the learning rate.

The input values are the same as in the ***profit*** function.

Output: It is the gradient of the logarithm of the profit function.

Function “opt_profit”

Calculates the optimal investment proportion vector for a given Assets_value matrix and Assets vector. For calculating the optimal value, it executes a gradient decent routine. If the error tolerance between two consecutive iterations is below a predefined threshold, algorithm will stop.

Inputs:

- **Assets_value**: It is the matrix representing the value of all stocks for all periods. The rows are the value for each period and the columns are the stocks.
- **Assets**: It a vector representing the assets that will be used for the calculation of the final profit. The length of is vector must be equal to the length of the investment proportion vector.
- **L-r:** It is the learning rate factor of the gradient decent algorithm.
- **Tol:** It is the tolerance error.

Outputs:

- **Opt_P**: It is the optimal profit value for the given Assets_value matrix and Assets vector in logarithm scale. It represents the ratio profit.
- **Opt_h**: It is the optimal investment proportion vector.
- **Error**: It is the error of the two final consecutive investment proportion vectors.
- **Num_it**: Provide the number of iterations excuted by the algorithm.

a. Calculate profit for h=[0,1] and h=[1,0]

In order to calculate the profit, it has been used the ***profit*** function. The Assets_value matrix was created using MATLAB functions. For this case the Assets argument is set to [1,2]. Results are presented below:

Vector: h	Profit (Log)	Profit
[0,1]	0	1
[1,0]	0.6931	2

Remark: It has been considered 102 periods for this task.

b. Calculate profit for h=[0.5,0.5]

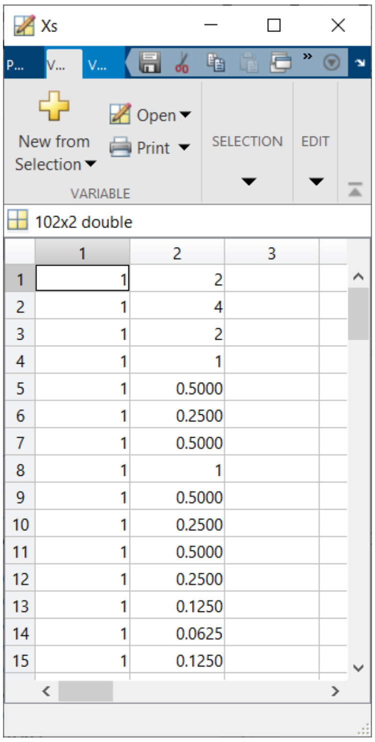
For this calculation it is used the profit function as in section a.

Vector: h	Profit (Log)	Profit
[0.5,0.5]	6.2946	541.6483

Remark: It has been considered 102 periods for this task.

c. Calculate profit for h=[0.5,0.5] and stock2 ratio being 0.5 or 2 with 0.5 probability

The Xn ratio should be 0.5 or 2 with 0.5 probability, then in ***Assets_value*** matrix it is calculated the first element being 0.5 or 2 with 0.5 probability then next elements are the product of the previous one by 2 or 0.5. Doing this way will ensure that the ratio is 0.5 or 2 for each period. Then is used the ***profit*** function to calculate profit.



	1	2	3
1	1	2	
2	1	4	
3	1	2	
4	1	1	
5	1	0.5000	
6	1	0.2500	
7	1	0.5000	
8	1	1	
9	1	0.5000	
10	1	0.2500	
11	1	0.5000	
12	1	0.2500	
13	1	0.1250	
14	1	0.0625	
15	1	0.1250	

Random Assets_value Matrix

Vector: h	Profit (Log)	Profit
[0.5,0.5]	9.0672	8666.4

Remark: It has been considered 102 periods for this task.

d. Task: Is h=[0.5,0.5] optimal?

In order to answer this question, it is used the function ***opt_profit***. The learning rate and error tolerance values are set to 0.01 and 0.001 respectively. The deterministic and stochastic cases are evaluated:

Deterministic case:

Opt. Profit (Log)	Opt. Profit	Opt. h
6.2996	544.3363	[0.4851,0.5149]

Stochastic case: Given that Assets_value matrix changes randomly, it will be presented the optimal values for 7 iterations and is also presented the number of ratios 2 (double profit) in the Stock2:

It	Opt. Profit (Log)	Opt. h	2s in Stock 2
1	8.6168	[0.3959,0.6041]	54
2	4.339	[0.5742,0.4258]	48
3	11.2925	[0.3069,0.6931]	57
4	2.2815	[0.693,0.307]	44
5	7.0323	[0.4554,0.5446]	52
6	10.3606	[0.3365,0.6635]	56
7	14.3308	[0.2179,0.7821]	60

Comments

- From the last section d). It is observed that the optimal investing proportion vector near [0.5,0.5] for the case of the deterministic stock value. But for the stochastic stock value, are totally different values.

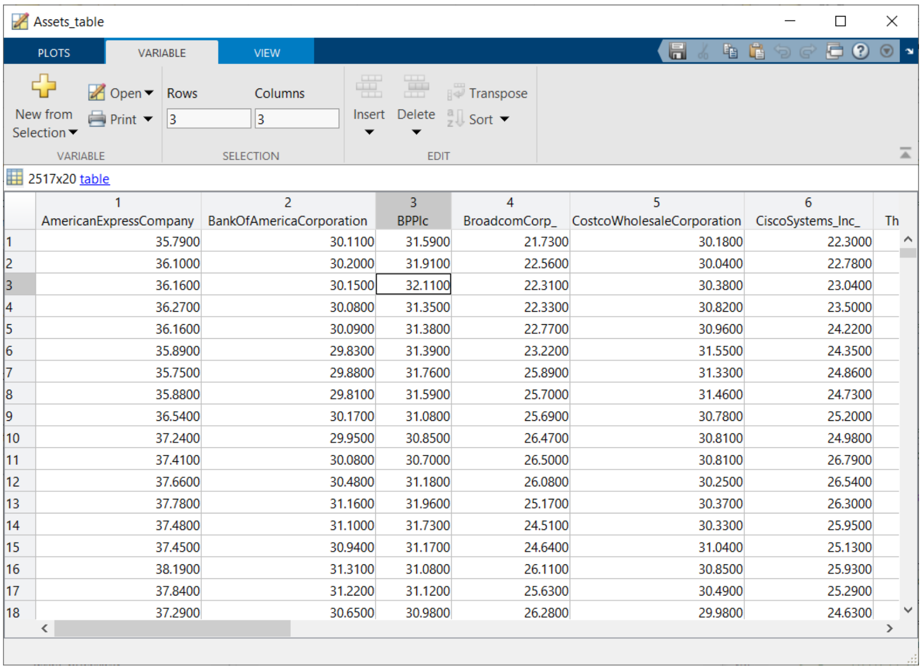
- The optimal profit is proportional to the number of ratios 2 (double profit) in stock 2. Indeed, more double profits is translated to more final profit at the end of all periods. In addition, it also observed that optimal investing vector tends to prioritize the stock 2 when the number of ratios 2 are more than 50, but if the number is below 50, then it prioritizes the stock 1.

- In the calculation of the optimal values for the deterministic stock 2, it was expected to have an investment vector of [0.5,0.5]. The value presented is different because the number of periods is 101 and not 102. This is translated in one less negative profit of stock2 (0.5), reason why the vector slightly prioritizes Stock2 over Stock1. Given that the Xn is the division between two consecutive stocks, the final Xn vector has 1 less element than the stocks vector which is 102 for this case.

5. Data Retrieval and Implementation

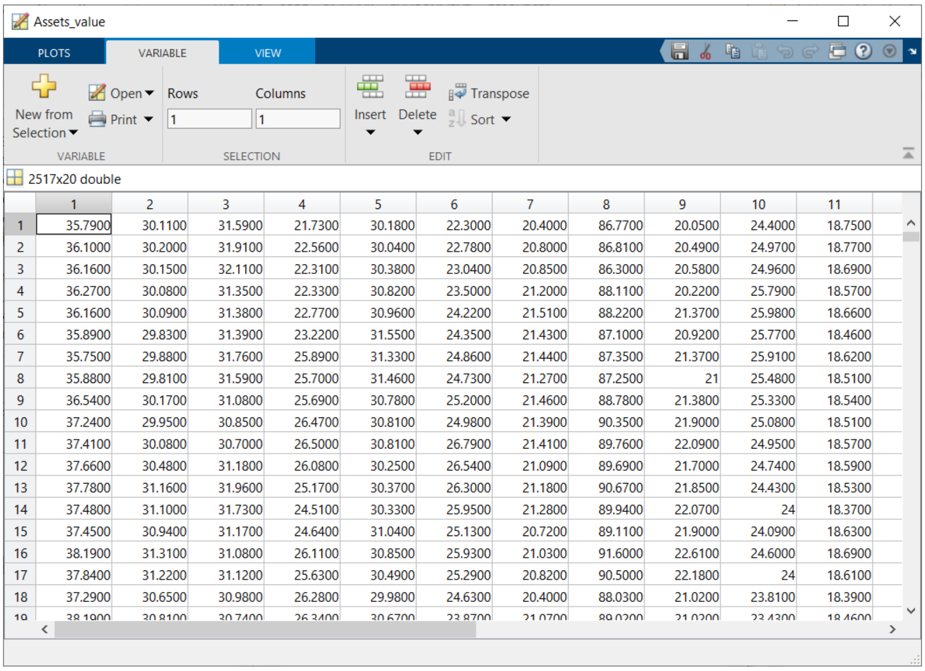
a. Downloading data from link in PDF

The data was downloaded loaded as table, then stock values saved matrix.



	1 AmericanExpressCompany	2 BankOfAmericaCorporation	3 BPPlc	4 BroadcomCorp_	5 CostcoWholesaleCorporation	6 CiscoSystems_Inc_	Th
1	35.7900	30.1100	31.5900	21.7300	30.1800	22.3000	
2	36.1000	30.2000	31.9100	22.5600	30.0400	22.7800	
3	36.1600	30.1500	32.1100	22.3100	30.3800	23.0400	
4	36.2700	30.0800	31.3500	22.3300	30.8200	23.5000	
5	36.1600	30.0900	31.3800	22.7700	30.9600	24.2200	
6	35.8900	29.8300	31.3900	23.2200	31.5500	24.3500	
7	35.7500	29.8800	31.7600	25.8900	31.3300	24.8600	
8	35.8800	29.8100	31.5900	25.7000	31.4600	24.7300	
9	36.5400	30.1700	31.0800	25.6900	30.7800	25.2000	
10	37.2400	29.9500	30.8500	26.4700	30.8100	24.9800	
11	37.4100	30.0800	30.7000	26.5000	30.8100	26.7900	
12	37.6600	30.4800	31.1800	26.0800	30.2500	26.5400	
13	37.7800	31.1600	31.9600	25.1700	30.3700	26.3000	
14	37.4800	31.1000	31.7300	24.5100	30.3300	25.9500	
15	37.4500	30.9400	31.1700	24.6400	31.0400	25.1300	
16	38.1900	31.3100	31.0800	26.1100	30.8500	25.9300	
17	37.8400	31.2200	31.1200	25.6300	30.4900	25.2900	
18	37.2900	30.6500	30.9800	26.2800	29.9800	24.6300	

Csv file as table in MATLAB



	1	2	3	4	5	6	7	8	9	10	11
1	35.7900	30.1100	31.5900	21.7300	30.1800	22.3000	20.4000	86.7700	20.0500	24.4000	18.7500
2	36.1000	30.2000	31.9100	22.5600	30.0400	22.7800	20.8000	86.8100	20.4900	24.9700	18.7700
3	36.1600	30.1500	32.1100	22.3100	30.3800	23.0400	20.8500	86.3000	20.5800	24.9600	18.6900
4	36.2700	30.0800	31.3500	22.3300	30.8200	23.5000	21.2000	88.1100	20.2200	25.7900	18.5700
5	36.1600	30.0900	31.3800	22.7700	30.9600	24.2200	21.5100	88.2200	21.3700	25.9800	18.6600
6	35.8900	29.8300	31.3900	23.2200	31.5500	24.3500	21.4300	87.1000	20.9200	25.7700	18.4600
7	35.7500	29.8800	31.7600	25.8900	31.3300	24.8600	21.4400	87.3500	21.3700	25.9100	18.6200
8	35.8800	29.8100	31.5900	25.7000	31.4600	24.7300	21.2700	87.2500	21	25.4800	18.5100
9	36.5400	30.1700	31.0800	25.6900	30.7800	25.2000	21.4600	88.7800	21.3800	25.3300	18.5400
10	37.2400	29.9500	30.8500	26.4700	30.8100	24.9800	21.3900	90.3500	21.9000	25.0800	18.5100
11	37.4100	30.0800	30.7000	26.5000	30.8100	26.7900	21.4100	89.7600	22.0900	24.9500	18.5700
12	37.6600	30.4800	31.1800	26.0800	30.2500	26.5400	21.0900	89.6900	21.7000	24.7400	18.5900
13	37.7800	31.1600	31.9600	25.1700	30.3700	26.3000	21.1800	90.6700	21.8500	24.4300	18.5300
14	37.4800	31.1000	31.7300	24.5100	30.3300	25.9500	21.2800	89.9400	22.0700	24	18.3700
15	37.4500	30.9400	31.1700	24.6400	31.0400	25.1300	20.7200	89.1100	21.9000	24.0900	18.6300
16	38.1900	31.3100	31.0800	26.1100	30.8500	25.9300	21.0300	91.6000	22.6100	24.6000	18.6900
17	37.8400	31.2200	31.1200	25.6300	30.4900	25.2900	20.8200	90.5000	22.1800	24	18.6100
18	37.2900	30.6500	30.9800	26.2800	29.9800	24.6300	20.4000	88.0300	21.0200	23.8100	18.3900
19	38.1900	30.8100	30.7400	26.3400	30.6700	23.8700	21.0700	89.0200	21.0200	23.4300	18.4600

Stock values in Matrix

b. Compute optimal for D = 2

It is presented the optimal for 2 pairs of companies:

Companies American Express and BPPlc:

- Assets: [1,3]
- Learning rate: 0.01
- Error Tolerance: 0.0001

Opt. Profit (Log)	Opt. Profit	Opt. h
0.9301	2.5348	[0.8832,0.1168]

The investing proportion vector prioritizes American Express over BPPlc

Companies Costco and Starbucks:

- Assets: [5,16]
- Learning rate: 0.01
- Error Tolerance: 0.0001

Opt. Profit (Log)	Opt. Profit	Opt. h
1.64	5.1552	[0.2343,0.7657]

The investing proportion vector prioritizes Starbucks over Costco

c. Compute optimal for D = 3

It is presented the optimal for 2 groups of companies:

Companies American Express, Costco and Tiffany Co:

- Assets: [1,5,18]
- Learning rate: 0.01
- Error Tolerance: 0.0001

Opt. Profit (Log)	Opt. Profit	Opt. h
1.3746	3.9535	[0.1213,0.8293,0.0493]

The investing proportion vector prioritizes Costco over the others.

Companies BPPlc, Broadcom and Microsoft:

- Assets: [3,4,13]
- Learning rate: 0.01
- Error Tolerance: 0.0001

Opt. Profit (Log)	Opt. Profit	Opt. h
0.6920	1.9977	[0.1932,0.2682,0.5387]

The investing proportion vector prioritizes Microsoft over the others.

d. Implement any convex optimization algorithm.

The project has been solved using Gradient Descent approach. This task is already solved.

e. Confirm results using brute force for D=2 and D=3.

For this task a loop over all possible combinations is used. A conditional statement will verify which of all profits calculated is the greatest, and its value and investment are saved. Each loop increment has been set to 0.01.

The error is defined:

$$e_P = abs(P_G - P_B)$$

$$e_h = sum(abs(\bar{h}_G - \bar{h}_B))$$

where:

- $e_P: Profit\ error$
- $P_G: Optimal\ profit\ (gradient)$
- $P_B: Optimal\ profit\ (Brute\ Force)$
- $e_h: Investment\ proportion\ vector\ error$
- $h_G: Optimal\ Investment\ proportion\ vector\ (gradient)$
- $P_B: Optimal\ Investment\ proportion\ vector\ (Brute\ Force)$

The results for all portfolio profiles are shown below, all profits are in logarithm scale:

Assets	e_P	P_G	P_B	e_h	h_G	h_B
[1,3]	0	0.9301	0.9301	0.0137	[0.8832,0.1168]	[0.89,0.11]
[5,16]	0.0001	1.64	1.6401	0.0286	[0.2343,0.7657]	[0.22,0.78]
[1,5,18]	0.0001	1.3746	1.3747	0.0227	[0.1213,0.8293,0.0493]	[0.11,0.84,0.05]
[3.4.13]	0.0001	0.6920	0.6921	0.0263	[0.1932,0.2682,0.5387]	[0.18,0.27,0.55]

Comments

- It is verified that the gradient descent performs well in optimizing the portfolio profit. Compared with brute forces results, error near to zero.

- During testing it was observed that for some stock pairs, the optimal value was located in the boundary of the profit curve (h=[1,0] or h=[0,1]). All these group of stocks (D=2 and D=3) were discarded, due to solutions are trivial and would be not possible to show performance measurements.

- In gradient descent the learning rate should be carefully set, large learning set did not converge and consequently, the algorithm will never reach the optimal profit.

MATLAB Scripts

Project 3

```
%% 4) Toy Example
%% Parameters
N = 3000; % Number of periods
Assets = [1,2]; % Assest to be evaluated from Assets Matrix
%% --- a) and b) Assets Matrix generation
X1 = ones(1,N);
X2 = repmat([1,2],[1,N/2]);
Xd = [X1',X2'];
% Investment vectors
ha_1 = [1,0]; % Investment vector for section a)
ha_2 = [0,1]; % Investment vector for section a)
hb = [0.5,0.5]; % Investment vector for section b)
%% Profit Calculation
Pa_1 = profit(ha_1,Xd,Assets);
Pa_2 = profit(ha_2,Xd,Assets);
Pb = profit(hb,Xd,Assets);
%% --- c) Assets Matrix generation
X3 = ones(1,N);
X4 = zeros(1,N);
X4(1) = (rand()>0.5)*1.5+0.5;
for i = 1:N-1
    X4(i+1) = X4(i)*((rand()>0.4)*1.5+0.5);
end
Xs = [X3',X4'];
hc = [0.5,0.5]; % Investment vector for section c)
%% Profit Calculation
Pc = profit(hc,Xs,Assets);
%% --- d) Optimization
alpha = 0.001; % Learning Rate gradiend descent
err_T = 0.0001; % Error Tolerance X(k+1)-X(k)
[Popt_d,hopt_d,err_d] = opt_profit(Xd,Assets,alpha,err_T); % Deterministic case
[Popt_s,hopt_s,err_s] = opt_profit(Xs,Assets,alpha,err_T); % Stochastic case
Ratio_2s = sum(((circshift(Xs(:,2),-1)./Xs(:,2))==2)); % Number of double profits

%% 5) Data Retrieval and Implementation
%% --- a) Data Preprocessing
filename = 'asset_prices.csv';
Assets_table = readtable(filename);
Assets_value = table2array(Assets_table(:,:));
%% --- b) Profit Calculation D=2
%% American Express and BPPIc
Assets = [1,3]; % Stocks 1 and 3
[Popt_b1,hopt_b1,err_b1] = opt_profit(Assets_value,Assets,alpha,err_T); % Optimal
%% Costco and Starbucks
Assets = [5,16]; % Stocks 5 and 16
[Popt_b2,hopt_b2,err_b2] = opt_profit(Assets_value,Assets,alpha,err_T); % Optimal
%% --- 2c) Profit Calculation D=3
%% American Express, Cotsco and Tiffany Co
Assets = [1,5,18]; % Stocks 1, 15 and 18
[Popt_c1,hopt_c1,err_c3] = opt_profit(Assets_value,Assets,alpha,err_T); % Optimal
%% BPPIc, Broadcom and Microsoft
Assets = [3,4,13]; % Stocks 3, 4 and 13
[Popt_c2,hopt_c2,err_c2] = opt_profit(Assets_value,Assets,alpha,err_T); % Optimal
%% --- 2e) Brute Force Verification
%% American Express and BPPIc
Assets = [1,3]; % Stocks 1 and 3
Num = 101;
P = zeros(1,Num);
Pmax_b1 = 0;
for i=1:Num
    h = [(i-1),(Num-i)]/(Num-1);
    P(i) = profit(h,Assets_value,Assets);
    if (P(i)>=Pmax_b1)
```

```
        Pmax_b1 = P(i);
        hmax_b1 = h;
    end
end
%% Costco and Starbucks
Assets = [5,16]; % Stocks 5 and 16
Num = 101;
P = zeros(1,Num);
Pmax_b2 = 0;
for i=1:Num
    h = [(i-1),(Num-i)]/(Num-1);
    P(i) = profit(h,Assets_value,Assets);
    if (P(i)>=Pmax_b2)
        Pmax_b2 = P(i);
        hmax_b2 = h;
    end
end
%% American Express, Cotsco and Tiffany Co
Assets = [1,5,18]; % Stocks 1, 15 and 18
Num = 101;
P = zeros(Num,Num);
Pmax_c1 = 0;
for j=1:Num
    T = Num-j+1;
    for i=1:T
        h = [(i-1),(T-i),j-1]/(Num-1);
        P(i,j) = profit(h,Assets_value,Assets);
        if (P(i,j)>=Pmax_c1)
            Pmax_c1 = P(i,j);
            hmax_c1 = h;
        end
    end
end
%% BPPIc, Broadcom and Microsoft
Assets = [3,4,13]; % Stocks 3, 4 and 13
Num = 101;
P = zeros(Num,Num);
Pmax_c2 = 0;
for j=1:Num
    T = Num-j+1;
    for i=1:T
        h = [(i-1),(T-i),j-1]/(Num-1);
        P(i,j) = profit(h,Assets_value,Assets);
        if (P(i,j)>=Pmax_c2)
            Pmax_c2 = P(i,j);
            hmax_c2 = h;
        end
    end
end
```

Function: opt_profit

```
function [opt_P,opt_h,error,num_it] = opt_profit(Assets_value,Assets,L_r,Tol)
    h_i = 0.5*ones(1, length(Assets)-1);
    h2 = [h_i, 1-sum(h_i)]; % Initial point
    num_it = 0; % Number of iterations
    error = 1; % Initial error
    while (error > Tol)
        grad_P = grad_profit(h2,Assets_value,Assets); % Gradient Calculation
        h_j = h_i + L_r*grad_P; % New h_i
        h2 = [h_j,1-sum(h_j)]; % New point
        if (sum(h_j)<0)
            h_j=zeros(1,length(Assets)-1);
            h2 = [h_j,1-sum(h_j)]; % New point
            break;
        end
        if (sum(h_j)>1)
            h_j=ones(1,length(Assets)-1);
```

```
            h2 = [h_j,1-sum(h_j)]; % New point
            break;
        end
        error = sum(abs(h_j-h_i));
        num_it = num_it + 1;
        h_i = h_j;
    end
    opt_P = profit(h2,Assets_value,Assets); % Optimal profit
    opt_h = h2; % Optimal proportion
end
```

Function: profit

```
function P = profit(h,Assets_value,Assets)
    %% Variables
    X = (circshift(Assets_value,-1)./Assets_value); % Growth matrix
    X(end,:) =[]; % Last row is deleted
    P = sum(log(sum(X(:,Assets).*h,2))); % Profit Calculation
end
```

Function: grad_profit

```
function grad_P = grad_profit(h,Assets_value,Assets)
    %% Variables
    X = (circshift(Assets_value,-1)./Assets_value); % Growth matrix
    X(end,:) =[]; % Last row is deleted
    XA = X(:,Assets);
    X_Xn = XA-XA(:,end);
    X_Xn(:,end) = [];
    grad_P = sum(1./sum(XA.*h,2).*X_Xn); % Profit Calculation
end
```