**INSTITUTO TECNOLÓGICO DE ESTUDIOS SUPERIORES DE MONTERREY**

**CAMPUS ESTADO DE MÉXICO**



**Applied Computer Science**

**Masters in Nanotechnology**

Manuel Valenzuela Rendón

**Linear, mixed, and quadratic programming**

Antonio Osamu Katagiri Tanaka (A01212611)

Bruno González Soria (A01169284)

**Due date: April 11, 2019, 15:59PM**

**MATLAB Script and Implemented Functions**

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% \* AUTHOR(S) :

% \* Bruno González Soria (A01169284)

% \* Antonio Osamu Katagiri Tanaka (A01212611)

% \*

% \* FILENAME :

% \* HW01.m

% \*

% \* DESCRIPTION :

% \* Computación Aplicada (Ene 19 Gpo 1)

% \* Homework on Linear, mixed, and quadratic programming

% \*

% \* NOTES :

% \*

% \*

% \* START DATE :

% \* 11 Apr 2019

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

warning('off')

clc;

clear all;

close all;

%% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% Problem 1:

% Solve the Following transportation problem:

s = [37.6; 40.4; 44.5]';

d = [20 30 30 40]';

C = [41 27 28 24; 40 29 100 23; 37 30 27 21];

% transportation costs tableau

% | destination | supply

% | 1 2 3 4 |

% ------------+-----------------+--------

% source 1 | 41 27 28 24 | 37.6

% 2 | 40 29 100 23 | 40.4

% 3 | 37 30 27 21 | 44.5

% ------------+-----------------+--------

% demand | 20 30 30 40 |

% Assume that only integer units can be transported.

% Assume that only multiples of 2 units can be transported.

% Upload to Blackboard a pdf file with a MATLAB script that solves both

% cases.

% The pdf should also include the solution matrix A for each case.

disp("Problem 1: Matrix of assignation and cost.");

[assig, total\_cost] = EvenTransportation(s,d,C);

% Print the calculations

disp(assig);

disp(total\_cost);

disp(" ");

%% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% Problem 2:

% Using function Investopedia, download the adjusted closing prices for the

% following DJI stocks from January 1, 2018 through January 1, 2019.

%

% KO Coca-Cola DIS Disney

% PG Procter & Gamble MCD McDonald’s

% PFE Pfizer WMT WalMart

% MRK Merck V Visa

% VZ Verizon

%

% Obtain the daily returns for these securities. Plot their expected return

% versus variance.

% Using command quadprog, obtain the optimal portfolios for

% k = 1; 2; 2:5; 4; 7; 9; 11; 20; 50; 1000

% For each value of k, report optimal weights, expected return and variance.

% Plot this data on the previous graph.

% Upload to Blackboard a pdf file that contains a MATLAB script, any MATLAB

% functions that you implemented, and the required plots and results.

disp("Problem 2: Percentage of investment given a risk-aversion.");

load('Tiingo\_data.mat')

PortfolioOptimization(res,data,1);

PortfolioOptimization(res,data,2);

PortfolioOptimization(res,data,2:5);

PortfolioOptimization(res,data,4);

PortfolioOptimization(res,data,7);

PortfolioOptimization(res,data,9);

PortfolioOptimization(res,data,11);

PortfolioOptimization(res,data,20);

PortfolioOptimization(res,data,50);

% WARNING: Your PC may freeze with a big k such as 1000. Uncomment at your

% own risk ...

PortfolioOptimization(res,data,1000);

%% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% Problem 1 FUNCTION DEFINITION

function [assig, total\_cost] = EvenTransportation(s,d,C)

% Only multiples of 2 units can be transported, so let's assume each unit

% contains 2 objects. Let's simulate this by divition by 2.

s = s/2;

d = d/2;

% Only integer units can be transported, so round toward negative infinity.

s = floor(s);

% The following is based on:

% http://web.tecnico.ulisboa.pt/mcasquilho/compute/\_linpro/TaylorB\_module\_b.pdf

% f = [C(1,:) C(2,:) C(3,:)]';

n = length(s);

m = length(d);

f = reshape(C',n\*m,1);

A = zeros(n,n\*m);

for i=1:n

A(i,1+(i-1)\*4:i\*4) = 1;

end

%

b = s;

Aeq = zeros(n,n\*m);

%

for j=1:m

Aeq(j,j:m:n\*m) = 1;

end

%

beq = d;

LB = zeros(n\*m,1);

UB = Inf(n\*m,1);

%

x = linprog(f,A,b,Aeq,beq,LB,UB);

%

assig = reshape(x,m,n)';

total\_cost = sum(sum(assig.\*C));

% Up to this point calculations have been made by units of 2 objects, let's

% multiply the result by 2 to get the transportation assignments and const

% by product

assig = assig\*2;

total\_cost = total\_cost\*2;

end

%% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% Problem 2 FUNCTION DEFINITION

function [out] = PortfolioOptimization(res,data,k)

% Let's convert the closing prices into returns

% CLOSING PRICES

CP\_KO = res.KO;

CP\_PG = res.PG;

CP\_PFE = res.PFE;

CP\_MRK = res.MRK;

CP\_VZ = res.VZ;

CP\_DIS = res.DIS;

CP\_MCD = res.MCD;

CP\_WMT = res.WMT;

CP\_V = res.V;

% RETURNS

R\_KO = (CP\_KO(2:end)-CP\_KO(1:end-1))./CP\_KO(1:end-1);

R\_PG = (CP\_PG(2:end)-CP\_PG(1:end-1))./CP\_PG(1:end-1);

R\_PFE = (CP\_PFE(2:end)-CP\_PFE(1:end-1))./CP\_PFE(1:end-1);

R\_MRK = (CP\_MRK(2:end)-CP\_MRK(1:end-1))./CP\_MRK(1:end-1);

R\_VZ = (CP\_VZ(2:end)-CP\_VZ(1:end-1))./CP\_VZ(1:end-1);

R\_DIS = (CP\_DIS(2:end)-CP\_DIS(1:end-1))./CP\_DIS(1:end-1);

R\_MCD = (CP\_MCD(2:end)-CP\_MCD(1:end-1))./CP\_MCD(1:end-1);

R\_WMT = (CP\_WMT(2:end)-CP\_WMT(1:end-1))./CP\_WMT(1:end-1);

R\_V = (CP\_V(2:end)-CP\_V(1:end-1))./CP\_V(1:end-1);

Returns = [R\_KO R\_PG R\_PFE R\_MRK R\_VZ R\_DIS R\_MCD R\_WMT R\_V];

fields = transpose(fieldnames(res)); % asset names

nAssets = length(fields); % number of assets

% Normalized Asset Prices

assetP = data./data(1, :); %%NormalizedPrice

figure(1);

plot(assetP);

xlabel('Day count');

ylabel('Normalized Price');

title('Normalized Asset Prices');

for i = 1:nAssets

text(length(assetP(:,i)),assetP(end,i),strcat( " ", fields(i)) );

end

% Expected returns / mean of the returns per security

muR = mean(Returns);

% % Risk-Adjusted Returns

% assetRisk = std(Returns);

% Returns variance per security / risk

C = cov(Returns); % std(R)^2 variances are in the main diagonal ...

sigmaR = diag(C); % extract the main diagonal of C

% Plot their expected return versus variance.

figure(2);

scatter(sigmaR,muR,2);

title("Pareto Front");

xlabel("Risk (Std Dev of Return)");

ylabel("Expected Return");

for i = 1:nAssets

text(sigmaR(i),muR(i),strcat( " ", fields(i)) );

end

% Quadratic Programming Portfolio Optimization

% based on: openExample('optim/PortfolioMIQPExample')

% Load the data for the problem

r = transpose(muR); % returns

Q = C;%sigmaR; % risk

% Set the number of assets

N = length(r);

% Create continuous variables xvars representing the asset allocation

% fraction

xvars = optimvar('xvars',N,1,'LowerBound',0,'UpperBound',1);

% binary variables vvars representing whether or not the associated

% % xvars is zero or strictly positive

% vvars = optimvar('vvars',N,1,'Type','integer','LowerBound',0,'UpperBound',1);

% and zvar representing the variable, a positive scalar.

zvar = optimvar('zvar',1,'LowerBound',0);

% Set the Optimization Problem

qpprob = optimproblem('ObjectiveSense','maximize');

% Set the risk-aversion: lambda = k

% and iterate if k is a vector

for lambda = k(1):k(end)

% Define the objective function

qpprob.Objective = r'\*xvars - lambda\*zvar;

% solving the problem with the current constraints

options = optimoptions(@intlinprog,'Display','off'); % Suppress iterative display

[xLinInt,~,~,~] = solve(qpprob,'options',options);

% stop iterating when the slack variable is within 0.01% of the true quadratic value

thediff = 1e-4;

iter = 1; % iteration counter

assets = xLinInt.xvars;

truequadratic = assets'\*Q\*assets;

zslack = zeros(length(truequadratic),1);

% keep a history of the computed true quadratic and slack variables for plotting.

history = [truequadratic,zslack];

options = optimoptions(options,'LPOptimalityTolerance',1e-10,'RelativeGapTolerance',1e-8,...

'ConstraintTolerance',1e-9,'IntegerTolerance',1e-6);

% Compute the quadratic and slack values.

while abs((zslack - truequadratic)/truequadratic) > thediff % relative error

% If the quadratic and slack values differ,

% then add another linear constraint and solve again.

constr = 2\*assets'\*Q\*xvars - zvar <= assets'\*Q\*assets;

newname = ['iteration',num2str(iter)];

qpprob.Constraints.(newname) = constr;

% Solve the problem with the new constraints

[xLinInt,~,~,~] = solve(qpprob,'options',options);

assets = (assets+xLinInt.xvars)/2; % Midway from the previous to the current

%assets = xLinInt(xvars); % Use the previous line or this one

truequadratic = xLinInt.xvars'\*Q\*xLinInt.xvars;

zslack = xLinInt.zvar;

history = [history;truequadratic,zslack];

iter = iter + 1;

end

% Convert the porfolio weights into percentages of investment per

% asset

Percentage\_of\_Investment = xLinInt.xvars/sum(xLinInt.xvars);

% Prepare variables to print

for i = 1:nAssets

print(i) = ...

strcat( ...

fields(i), ...

" : ", ...

num2str(round(Percentage\_of\_Investment(i),4)), ...

"%" ...

);

end

% Retun the final calculation

out = transpose(print);

% and print

disp(strcat( ...

"Percentage of Investment with ", ...

"risk-aversion k = ", num2str(lambda)));

disp(out);

% PLOT Efficient Frontier

% Let's do sigmaR.^2 as it (re)calculates the standard deviation

% which is the square-root of variance)

p = Portfolio('AssetMean',muR, 'AssetCovar',sigmaR.^2,'AssetList',fields);

p = setDefaultConstraints(p);

p = setSolver(p, 'quadprog');

hold on

plotFrontier(p)

hold off

% % PLOT Percentage\_of\_Investment

% figure;

% bar(Percentage\_of\_Investment, 0.125);

% grid on;

% xlabel('Asset index');

% ylabel('Proportion of investment');

% title(strcat("Optimal asset allocation with k = ",num2str(lambda)));

% for i = 1:nAssets

% text(i,Percentage\_of\_Investment(i),fields(i));

% end

end

end

**Plots and Results**

**Problem 1:** Matrix of assignation and cost.

Optimal solution found.

**Matrix A:**

0 30 6 0

0 0 0 40

20 0 24 0

**Optimal Cost: 3286**

**Problem 2:** Percentage of investment given a risk-aversion.

Percentage of Investment with risk-aversion **k = 1**

"KO : 0%"

"PG : 0%"

"PFE : 0.3258%"

"MRK : 0.3258%"

"VZ : 0.1509%"

"DIS : 0%"

"MCD : 0%"

"WMT : 0%"

"V : 0.1974%"

Percentage of Investment with risk-aversion **k = 2**

"KO : 0%"

"PG : 0%"

"PFE : 0.4578%"

"MRK : 0.5325%"

"VZ : 0%"

"DIS : 0%"

"MCD : 0%"

"WMT : 0%"

"V : 0.0097%"

Percentage of Investment with risk-aversion **k = 2.5**

"KO : 0%"

"PG : 0%"

"PFE : 0.358%"

"MRK : 0.642%"

"VZ : 0%"

"DIS : 0%"

"MCD : 0%"

"WMT : 0%"

"V : 0%"

Percentage of Investment with risk-aversion **k = 4**

"KO : 0%"

"PG : 0%"

"PFE : 0.0974%"

"MRK : 0.9026%"

"VZ : 0%"

"DIS : 0%"

"MCD : 0%"

"WMT : 0%"

"V : 0%"

Percentage of Investment with risk-aversion **k = 7**

"KO : 0%"

"PG : 0%"

"PFE : 0.04%"

"MRK : 0.96%"

"VZ : 0%"

"DIS : 0%"

"MCD : 0%"

"WMT : 0%"

"V : 0%"

Percentage of Investment with risk-aversion **k = 9**

"KO : 0%"

"PG : 0%"

"PFE : 0.0447%"

"MRK : 0.9553%"

"VZ : 0%"

"DIS : 0%"

"MCD : 0%"

"WMT : 0%"

"V : 0%"

Percentage of Investment with risk-aversion **k = 11**

"KO : 0%"

"PG : 0%"

"PFE : 0.0387%"

"MRK : 0.9613%"

"VZ : 0%"

"DIS : 0%"

"MCD : 0%"

"WMT : 0%"

"V : 0%"

Percentage of Investment with risk-aversion **k = 20**

"KO : 0%"

"PG : 0%"

"PFE : 0.0406%"

"MRK : 0.9594%"

"VZ : 0%"

"DIS : 0%"

"MCD : 0%"

"WMT : 0%"

"V : 0%"

Percentage of Investment with risk-aversion **k = 50**

"KO : 0%"

"PG : 0%"

"PFE : 0.0463%"

"MRK : 0.9537%"

"VZ : 0%"

"DIS : 0%"

"MCD : 0%"

"WMT : 0%"

"V : 0%"

Percentage of Investment with risk-aversion **k = 1000**

"KO : 0%"

"PG : 0%"

"PFE : 0.0463%"

"MRK : 0.9537%"

"VZ : 0%"

"DIS : 0%"

"MCD : 0%"

"WMT : 0%"

"V : 0%"

