Computational Economics Problem Set 5

Names: Di Lu(5706079),), Zexi Sun(5917910), Penghui Yin(5702358), Shiwei Yu(5913037)

**Problem 2**

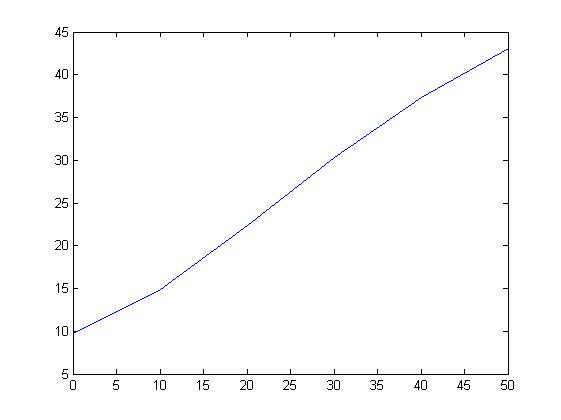
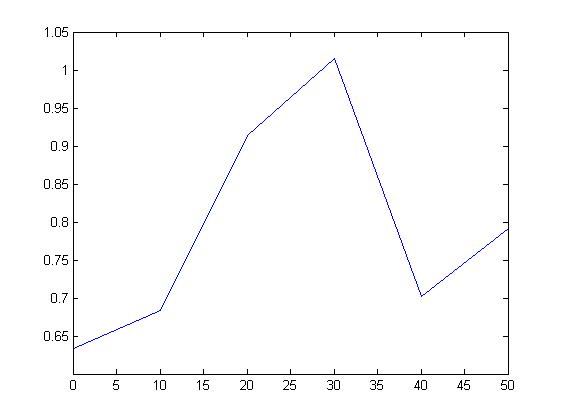
2.1

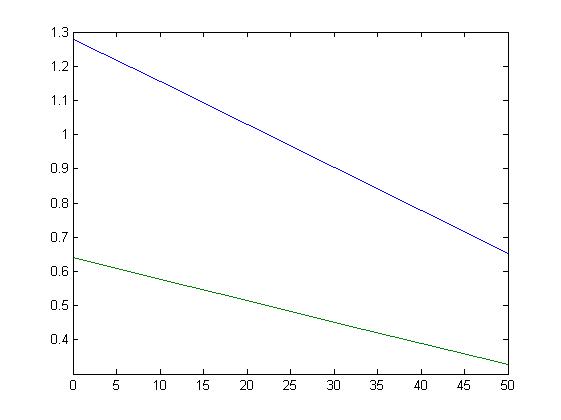
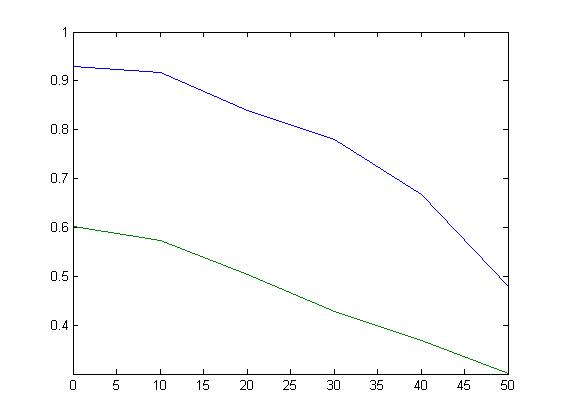
Very straight forward if we plug in the second term of W1 into the first formula and take the first derivative respect to αi , and then we get n equations exactly the same as we are asked to verify.

2.2

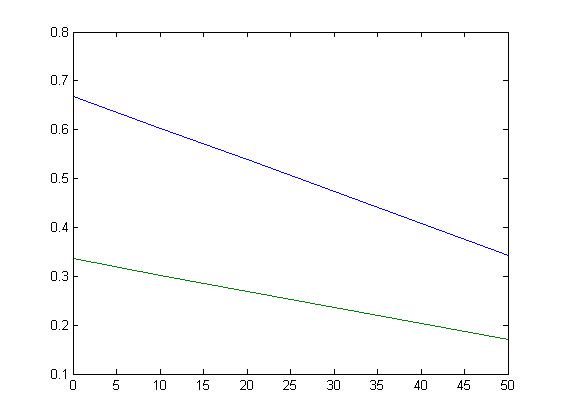
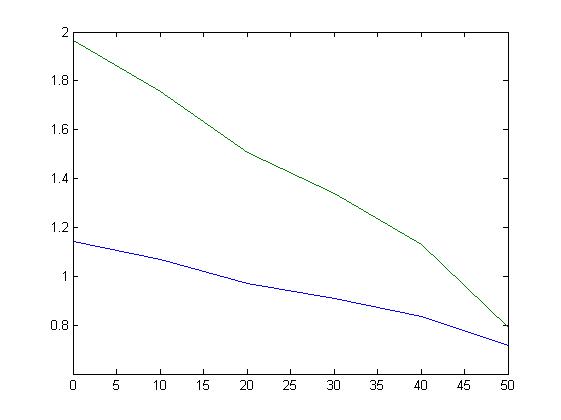
We first think of this problem with an implicit assumption that the sum of shares equal to one. Then this problem becomes a univariate one, and we try different method to solve this problem. We first implement Newton Method manually and get a robust answer in terms of initial guess. Then we also apply the fzero, fsolve, fminunc to this problem but all were found sensitive to the initial guess. So we use the first method to solve the problem with different amount of minimum wealth and get the picture in the up-left. We find it really strange, so head to another method afterwards which is fixed point iteration. As we thought there was no constraint on the share, any number might be a solution. Actually the fixed point iteration’s answer is also stable within the initial guess range from 3 to 13. Thus we also plot a solution of this method in the up-right.

After we were informed of the online solution, we figured out there was no constraint on the sum of 1. We revised our code and get a reasonable result similar to the reference code. However, we find the solutions are not close with the same parameters and initial guess. Furthermore, they are both sensitive to the initial guess. In the down-left, we plot the initial guess of [1,1] in comparison to the sample answer given, which is in the down-right. As the fix-point iteration doesn’t get a stable answer within the reasonable range in the first case, and we are constrained by the time, we didn’t do that again. In order to save space, we just include the last part of code in the Appendix.



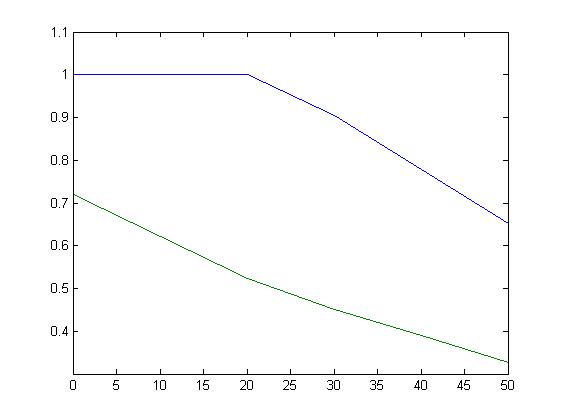
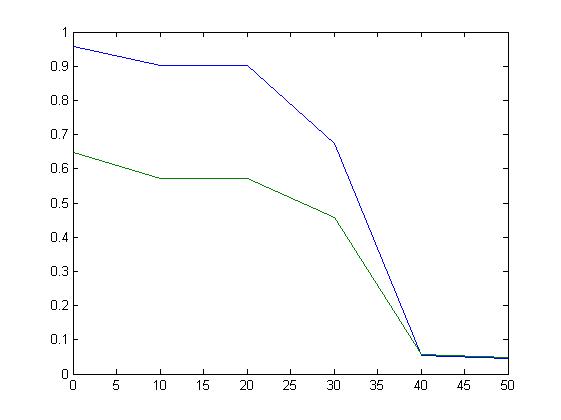


During our discussion, we find something interesting when playing around the parameters. When we change the degree of risk averse, gemma from 1 to 2, we get following pictures, also ours result in the left and sample solution in the right. But when it comes to the constrained problem in next part, our methods get the same result as the sample solution gives, both are similar to the picture in the right, which is the unconstrained problem. Maybe in this way, our result is better.



2.3

In the constrained problem, our method with matlab function fmincon is surely worse than the sample solution given the same initial setting of the problem set, including both parameters and initial guess. Comparison is shown in the following. But as mentioned, we get the similar result if we change gemma from 1 to 2. But in this way, our solution seems strange then. As we see in the previous picture that both shares exceed one in the unconstrained case, there should be a flat line of one when it is the binding case, but there isn’t. Furthermore, the constrained result from sample solution is around 0.6 lower than the unconstrained one for both lines, which also makes us very confusing.



**Appendix**

PS5Q2

|  |
| --- |
| clc; close all; clear all;  global gemma  %Parameters setting  r\_f=0.02;ru=0.5;mu=[0.04,0.06]';delta\_sd=[0.1,0.2];w0=100;wmin=0;gemma=2;a0=6;  rng(321);%set the seed  cov\_m=diag(delta\_sd.^2);cov=ru\*delta\_sd(1)\*delta\_sd(2);  cov\_m(2)=cov;cov\_m(3)=cov;  r=mu+cov\_m\*rand(2,1);  if any(eig(cov\_m)<0),  error('variance-covariance matrix must be p.d.');  end;  %Approximate the expectation by Gauss-Hermite integration using m = 7  %nodes by the MF-function qnwnorm.  m=7;n=[m,m];  [x,w] = qnwnorm(n,mu',cov\_m);  %E=w'\*f(x),e.g.w'\*exp(x(:,1)+x(:,2))    syms A  A0=[1,1]';%very sensitive to initial guess  Astar=fsolve(@(A)-w'.\*1/(1-gemma)\*((1+r\_f+(x-r\_f)\*A)\*w0-wmin).^(1-gemma),A0);    syms a y  y=-w'.\*1/(1-gemma)\*((1+r\_f+a\*(x(:,1)-r\_f)+(1-a)\*(x(:,2)-r\_f))\*w0-wmin).^(1-gemma);  df = diff(y,a);  mdf = matlabFunction(df);  astar = fzero(mdf,1);%additional constrain with a(1)+a(2)=1    wm=(0:10:50);astar1=zeros(2,length(wm));A1=A0;  for i=1:1:length(wm)  astar1(:,i) = fsolve(@(A)-w'.\*1/(1-gemma)\*((1+r\_f+(x-r\_f)\*A)\*w0-wm(i)).^(1-gemma),A1);  A1=astar1(:,i);  end  plot(wm,astar1)    % syms y  % a = sym('a', [2 1]);  % y=-w'.\*1/(1-gemma)\*((1+r\_f+(x-r\_f)\*a)\*w0-wmin).^(1-gemma);  % f=matlabFunction(y);  % astar2=fminunc(f,A0);  options = optimoptions('fminunc');  options = optimoptions(options,'Algorithm', 'quasi-newton');  options = optimoptions(options,'MaxFunEvals', 400);  Astar2=fminunc(@(A)-w'.\*1/(1-gemma)\*((1+r\_f+(x-r\_f)\*A)\*w0-wmin).^(1-gemma),A0,options);    wm=(0:10:50);astar2=zeros(2,length(wm));A1=A0;  for i=1:1:length(wm)  astar2(:,i) = fminunc(@(A)-w'.\*1/(1-gemma)\*((1+r\_f+(x-r\_f)\*A)\*w0-wm(i)).^(1-gemma),A1,options);  A1=astar2(:,i);  end  figure  plot(wm,astar2)    options = optimoptions('fmincon');  options = optimoptions(options,'Algorithm', 'sqp');  Astar3=fmincon(@(A)-w'.\*1/(1-gemma)\*((1+r\_f+(x-r\_f)\*A)\*w0-wmin).^(1-gemma),A0,[],[],[],[],[0;0],[1;1],[],options);  Astar3c=fmincon(@(A)-w'.\*1/(1-gemma)\*((1+r\_f+(x-r\_f)\*A)\*w0-wmin).^(1-gemma),A0,[],[],[1,1],[1],[0;0],[1;1],[],options);  wm=(0:10:50);astar3=zeros(2,length(wm));A1=A0;  for i=1:1:length(wm)  astar3(:,i)=fmincon(@(A)-w'.\*1/(1-gemma)\*((1+r\_f+(x-r\_f)\*A)\*w0-wm(i)).^(1-gemma),A1,[],[],[],[],[0;0],[1;1],[],options);  A1=astar3(:,i);  end  figure  plot(wm,astar3)    optset('ncpsolve','type','smooth');  % USAGE  % optset(funcname,optname,optvalue)  % INPUTS  % funcname : name of function  % optname : name of option  % optval : option value  alpha0=ones(length(mu),1);  disp(' ');  disp('constrained problem:');  a1=ones(2,length(wm));  for i=1:1:6,  wmin=10\*(i-1);  alph=ncpsolve(@ncpalphres,zeros(length(mu),1),ones(length(mu),1),alpha0,w0,wmin,r\_f,x,w,gemma);  alpha0=alph;  disp(['pf-shares for minimum wealth level ', num2str(wmin), ' are: ', num2str(alph')]);  a1(:,i)=alph;i=i+1;  end;  figure  plot(wm,a1(1,:),wm,a1(2,:))  %unconstrained problem with sample code  a2=ones(2,length(wm));  for i=1:1:6  wmin=10\*(i-1);  alph=broyden(@alphres,alpha0,w0,wmin,r\_f,x,w,gemma);  alpha0=alph;a2(:,i)=alph;  disp(['pf-shares for minimum wealth level ', num2str(wmin), ' are: ', num2str(alph')]);  end;  figure  plot(wm,a2(1,:),wm,a2(2,:)) |

|  |
| --- |
| function res=alphres(alph,w0,wmin,rf,rnodes,rwghts,gemma)    n=length(alph);  res=zeros(n,1);  for i=1:n,  res(i) = rwghts'\*((w0.\*(1+rf+(rnodes-rf)\*alph)-wmin).^-gemma.\*w0.\*(rnodes(:,i)-rf));  end; |

|  |
| --- |
| function res=alphres(alph,w0,wmin,rf,rnodes,rwghts,gemma)    n=length(alph);  res=zeros(n,1);  for i=1:n,  res(i) = rwghts'\*((w0.\*(1+rf+(rnodes-rf)\*alph)-wmin).^-gemma.\*w0.\*(rnodes(:,i)-rf));  end; |