Appendix

Appendix pre-fix: user guide for matlab program

(1). Firstly, produce optimal portfolio weights using Markwitz Mean_Variance optimization ; Then, use the optimal weights to get given portfolio values X*

>> N=10,T=11 >> x1=optiml_portfolio_mean_varian(N,T) >> plotPortfDemoStandardModel(x1) >> portfolio_given(N,T)

(2). Implement restoring process in EUT, choosing N=3,T=4

>> portfolio_given( ) >> beta(3,4) >> alpha(3,4) >> utility(1,3,4)

>> utility(2,3,4) >> utility(3,3,4)

(3). IID TEST: (All test function outputs the logical value, if equal to 1, that means, the iid hypothesis is accepted;; otherwise , it is rejected)

(1). // `turning_point_test.m` >> N=3,T=4,alpha=0.05 >> turning_point_test(N,T,alpha)

(2). difference_sign_test.m >> difference_sign_test(N,T,alpha) (3). // rank_test.m >> rank_test(N,T,alpha) Result: All three testing method can test the given optimal choice of portfolio weights follows IID.

(4) Implement restoring process in MDT, to restore derivative function of general deviation measure.

Step1. riskI_envelope.m;; [optimization problem for asset i] Step2. mean_returnOfobservation.m;; [obtain mean return based on observation over T time periods]

1. CVaR_asset_I.m [CVaR number for asset i;;]

2. Covar_asset_I.m [covar number for asset I;; it is not covariance, it is the average value evaluation based on risk envelop for each asset i ;;]

3. Mix_CVaR_coeff.m [for worst_case Mixed CVaR]

CVaR_optimality_condition.m 4. risk_profile Test: >> risk_profile(N,T)

The expected should look like a semi-circle in 1.3.

(5) Implement restoring process in MDT, to restore uniquely the general deviation measure.

Evaluate I=1, for 1/T, the deviation measure >> I=1 >> deviation_measure_unique(I,N,T)

Appendix I 1.// turning_point_test.m

```
function [ LV ] =
turning_point_test( N,T,alpha ) %TURNING_POINT_TEST Summary
of this function goes here % the output parameter is a
logical value;;

P = [0.0162355710000000,-0.00246812000000000,-
0.0404016810000000,- 0.0419082760000000,-
0.0737995490000000,-0.0340909090000000,-
0.0419082760000000,1645.88427300000,0.0354098360000000,-
0.184210526000000;

0.0257420960000000,-
0.00515463900000000,0.0194694570000000,0.0707015130000000,-
0.00939457200000000,0.0745098040000000,0.0707015130000000,0
.171171171000000 ,0.0232742240000000,-0.0645161290000000;
```

```
0.0265704650000000,0.0130569950000000,0.00381952700000000,-
0.00539568300000000,-0.0386371620000000,-
0.0364963500000000,-
0.00539568300000000,0.107692308000000,0.0479653410000000,0.
0689655170000000;

0.0131094440000000,-
0.0180032730000000,0.0623067780000000,0.0211831570000000,0.
0372670810000000 ,0.121212121000000,0.0211831570000000,0.00
694444400000000,0.022442049000000 0,0.322580645000000;

-
0.0114467550000000,0.0729166670000000,0.0315648090000000,0.
0131545660000000 ,-
0.126805213000000,0.0405405410000000,0.0131545660000000,-
0.999508966000000,-0.0127075810000000,0.121951220000000;

0.0142033320000000,0.106796117000000,-0.0108506940000000,-
0.00549313400000000,-
0.0338846310000000,0.0324675320000000,-
0.00549313400000000,1515.85393300000,0.0783969580000000,-
0.0652173910000000; -0.00970231200000000,-
0.0417543860000000,-0.0217200530000000,-
0.0271152400000000,0.0237995820000000,-0.0283018870000000,-
0.0271152400000000,-0.999390741000000,0.0187169400000000,-
0.0465116280000000;

-0.0126682260000000,0.0252654710000000,-
0.0461986990000000,-
0.0237419350000000,0.154159869000000,0.0226537220000000,-
0.0237419350000000,-0.0516717330000000,0.0171748100000000,-
0.0731707320000000;

-
0.00121538500000000,0.0446428570000000,0.00681871600000000,
0.00211472400000 000,-
0.0155477030000000,0.107594937000000,0.00211472400000000,-
0.0160256410000000,-0.0210732980000000,-0.0263157890000000;

-0.00787110500000000,0.0769230770000000,-
0.0366651100000000,- 0.0308625690000000,-
0.113424264000000,-0.00571428600000000,-
0.0308625690000000,-0.0358306190000000,-
0.0299505280000000,- 0.0540540540000000;
```

```
0.00523210700000000,0.178571429000000,0.0729696970000000,0.
0204137180000000 ,0.0700404860000000,0.0890804600000000,0.0
204137180000000,488.864864900000,
0.00110268800000000,0;]; P=P'; % P: (N-by-T)

r=P'; lamda = optiml_portfolio_mean_varian(N,T);

lamda=lamda';
ux=lamda*r(1,:)';
for t=2:4
  ux=[ux,lamda*P(:,t)];
end

%%%%%

d1=sign(diff(ux)); d2=diff(d1);
num_turn_point=numel(find(abs(d2))==2);

%%%

 crite=norminv(1-alpha/2);
 drift=(2/3)*(T-2);
 variance=(16*T-29)/90;
 std=sqrt(variance);
if abs(num_turn_point-drift)/std > crite LV=false(ones);

else

    LV=true(ones);
end end
```

2.// difference_sign_test.m

```
function [ LV ] =
difference_sign_test( N,T,alpha ) %DIFFERENCE_SIGN_TEST
Summary of this function goes here % Detailed explanation
goes here P = [0.013109444000000 -0.011446755000000
0.014203332000000 - 0.006573504000000;

-0.006901994000000 -0.016561345000000 0.021431869000000
0.004131142000000;

0.012556248000000 0.051666835000000 0.026383835000000 -
0.005190110000000]; r=P'; lamda =
optiml_portfolio_mean_varian(N,T);

lamda=lamda';
ux=lamda*r(1,:)';
for t=2:4
  ux=[ux,lamda*P(:,t)];
end

%%%

d=diff(ux);
num_diff_positv=numel(find(sign(d)==1)); %%% crite=norminv(1
-alpha/2); drift=(T-
1)/2; variance=(T+1)/12; std=sqrt(variance); if
abs(num_diff_positv-drift)/std > crite

  LV=false(ones);
else

  LV=true(ones);
end end
```

3.// rank_test.m

```
function [ LV ] = rank_test( N,T,alpha )
%RANK_TEST Summary of this function goes here % Detailed
explanation goes here P = [0.013109444000000 -
0.011446755000000 0.014203332000000 - 0.006573504000000;

-0.006901994000000 -0.016561345000000 0.021431869000000
0.004131142000000;

0.012556248000000 0.051666835000000 0.026383835000000 -
0.005190110000000]; r=P'; lamda =
optiml_portfolio_mean_varian(N,T);

lamda=lamda';
ux=lamda*r(1,:)';
for t=2:4
  ux=[ux,lamda*P(:,t)];
end

%%%
```

```
num_pair_positv=0;
for j=T:-1:2
   dj=ux(j)-ux(1:j-1);
num_pair_positv=num_pair_positv+numel(find(dj>0));

end

%%%

crite=norminv(1-alpha/2);
drift=T*(T-1)/4;
variance=T*(T-1)*(2*T+5)/8;
std=sqrt(variance);
if abs(num_pair_positv-drift)/std > crite LV=false(ones);

else

   LV=true(ones);
end end
```

Appendix II 1.//beta.m

```
function [ XX ] = beta( N,T ) %BETA Summary of this function
goes here P = [0.013109444000000 -0.011446755000000
0.014203332000000 - 0.006573504000000;

-0.006901994000000 -0.016561345000000 0.021431869000000
0.004131142000000;

0.012556248000000 0.051666835000000 0.026383835000000 -
0.005190110000000]; lamda =
optiml_portfolio_mean_varian(N,T); lamda=lamda';

ux=lamda*P(:,1);
for t=2:T
  ux=[ux,lamda*P(:,t)];
end

ux=sort(ux);
 f=zeros(1,T+1);
 f(1,T+1)=1;
 bet=zeros(N,T+1);
 X=zeros(N,T+1);
 XX=zeros(N,T);
 B=zeros(N,T);
 C=zeros(N,T);
%the solution of linear programming
% the solution of beta
for i=1:N
    for t=1:T
         B(i,t)=  P(i,t)-ux(t);
end end

for i=1:N
       uB=B(i,:);
       C(i,:)=uB(T:-1:1);
end

e1=ones(N,1); e2=ones(T-1,1); ub1=spdiags([-1*e2],1,T-1,T);
ub2=spdiags([e2,-1*e2],0:1,T-1,T); z=zeros(T-
1,1); ub1=[ub1,z]; ub2=[ub2,z];

%%%%%%%%

 bet=[C,-1*e1];
 for i=1:N
    A=[bet(i,:);ub1;ub2];
    b=zeros(size(A,1),1);
    Aeq=zeros(1,T+1);
    Aeq(1,1)=1;
    beq=ones(size(Aeq,1),1);
    X(i,:)=linprog(f,A,b,Aeq,beq);
    tmx=X(i,:);
    XX(i,:)=tmx(T:-1:1);
end end
```

2.//alpha.m

```
function [ alph ] = alpha( N,T)
%ALPHA Summary of this function goes here
ux=portfolio_given(N,T);
alph=zeros(N,T);
sum=zeros(N,T-1);
xx=beta(N,T);
for i=1:N
  for s=1:(T-1)
     for j=s:(T-1)
```

```
sum(i,s)=(xx(i,j+1)-xx(i,j))*ux(j)+sum(i,s);

end end

end

alph_T=zeros(N,1);
sum=[sum,alph_T];
alph=sum;
end
```

3.//utility.m

```
function [ output_args ] = utility( I,N,T )
%UTILITY Summary of this function goes here
% I denotes the asset I
ux=portfolio_given(N,T);
alph=zeros(N,T);
bet=zeros(N,T);
alph=alpha(N,T);
bet=beta(N,T);
%%%%%

y1={0,0,0,0};
z1=zeros(N,T);
for i=2:3
    y1=[y1;{0,0,0,0}];
end for i=1:N

for t=1:T y1{i,t}=@(x)bet(i,t)*x+alph(i,t);
z1(i,t)=y1{i,t}(ux(t));

end end

uxx=[-0.03,ux];
for i=1:N
   zz1(i)=y1{i,1}(-0.03);
end

zz=[zz1',z1];
uxi=uxx(1):0.00025:uxx(T+1);
uyi = interp1(uxx,zz(I,:),uxi);
plot(uxx,zz(I,:),'O',uxi,uyi);
end
```

Appendix III 1.// optiml_portfolio_mean_varian.m

```
function [ x1 ] =
optiml_portfolio_mean_varian( N,T ) %OPTIML_PORTFOLIO_MEAN_
VARIAN Summary of this function goes here %N=10,T=11,P:(T-
by-N) P = [0.0162355710000000,-0.00246812000000000,-
0.0404016810000000,- 0.0419082760000000,-
0.0737995490000000,-0.0340909090000000,-
0.0419082760000000,1645.88427300000,0.0354098360000000,-
0.184210526000000;

0.0257420960000000,-
0.00515463900000000,0.0194694570000000,0.0707015130000000,-
0.00939457200000000,0.0745098040000000,0.0707015130000000,0
.171171171000000 ,0.0232742240000000,-0.0645161290000000;

0.0265704650000000,0.0130569950000000,0.00381952700000000,-
0.00539568300000000,-0.0386371620000000,-
0.0364963500000000,-
0.00539568300000000,0.107692308000000,0.0479653410000000,0.
0689655170000000;

0.0131094440000000,-
0.0180032730000000,0.0623067780000000,0.0211831570000000,0.
0372670810000000 ,0.121212121000000,0.0211831570000000,0.00
694444400000000,0.0224420490000000 0,0.322580645000000;

-
0.0114467550000000,0.0729166670000000,0.0315648090000000,0.
0131545660000000 ,-
0.126805213000000,0.0405405410000000,0.0131545660000000,-
0.999508966000000,-0.0127075810000000,0.121951220000000;

0.0142033320000000,0.106796117000000,-0.0108506940000000,-
0.00549313400000000,-
0.0338846310000000,0.0324675320000000,-
0.00549313400000000,1515.85393300000,0.0783969580000000,-
0.0652173910000000; -0.00970231200000000,-
0.0417543860000000,-0.0217200530000000,-
0.0271152400000000,0.0237995820000000,-0.0283018870000000,-
0.0271152400000000,-0.999390741000000,0.0187169400000000,-
0.0465116280000000;
-0.012668226000000,0.025265471000000,-
0.046198699000000,-
0.023741935000000,0.154159869000000,0.022653722000000,-

0.023741935000000,-0.051671733000000,0.017174810000000,-
0.073170732000000;

-
0.001215385000000,0.044642857000000,0.006818716000000,
0.0021147240000 000,-
0.015547703000000,0.107594937000000,0.002114724000000,-
0.016025641000000,-0.021073298000000,-0.026315789000000;

-0.007871105000000,0.076923077000000,-
0.036665110000000,- 0.030862569000000,-
0.113424264000000,-0.005714286000000,-
0.030862569000000,-0.035830619000000,-
0.029950528000000,- 0.054054054000000;

0.005232107000000,0.178571429000000,0.072969697000000,0.
020413718000000 ,0.070040486000000,0.089080460000000,0.0
20413718000000,488.864864900000,
0.001102688000000,0;]; P=P'; %%P: (N-by-T)

r=P'; mean_return=mean_returnOfobservation(N); %asset
I %alternatively, use command mean(r), get the same
result...

%%desired return;; % risk_free fixed saving account
interest rate %%choose fixed saving rate 2.03% for one
year, we need weekly rate;; rfm=0.0203; % interest rate for
one year rfm=(rfm+1)^(7/365)-1 ; % weekly risk_free interst
rate chosen

%%% stdDev_return=std(r,1,1); %%% set flag=1 (the second
input parameter)
correlation=corrcoef(r); a1=stdDev_return'*stdDev_return; co
variance=correlation.*a1; nAssets=numel(mean_return);

Aeq=ones(1,nAssets); beq=1; Aineq=-mean_return; bineq=-
rfm; lb=zeros(nAssets,1); ub=ones(nAssets,1); c=zeros(nAsset
s,1); options=optimset('quadprog'); %%%default options for
the solver quadprog;;
```

```
options = optimset(options,'Algorithm', 'interior-point-
convex'); options =
optimset(options,'Display','iter','TolFun',1e-
10); %%%additional option setting tic
[x1,fval1]=quadprog(covariance,c,Aineq,bineq,Aeq,beq,lb,ub,
[],options); toc end
```

2.// plotPortfDemoStandardModel.m

```
function plotPortfDemoStandardModel(x1) %
plotPortfDemoStandardModel Helper function for portfolio
optimization demo

figure; bar1 = bar(x1,'FaceColor','b','EdgeColor','b');
set(bar1,'BarWidth',0.2); %set(gca,'xlim',[1
length(x1)]) %set(gca,'ylim',[0 0.3])

%%%objective function has no linear term
%set(gca,'xTick',[1 75 150 225]);
title('Mean-Variance-Standard model - 10-asset problem')
xlabel('Assets') ylabel('Fraction of investment') grid on

end
```

3.// portfolio_given.m

```
function [ ux ] = portfolio_given(N,T) %PORTFOLIO_GIVEN
Summary of this function goes here %%N=10,T=11,P:(T-by-N) P
= [0.016235571000000,-0.002468120000000,-
0.040401681000000,- 0.041908276000000,-
0.073799549000000,-0.034090909000000,-
0.041908276000000,1645.884273000000,0.035409836000000,-
0.184210526000000;

0.025742096000000,-
0.005154639000000,0.019469457000000,0.070701513000000,-
0.009394572000000,0.074509804000000,0.070701513000000,0
.171171171000000 ,0.023274224000000,-0.064516129000000;

0.026570465000000,0.013056995000000,0.003819527000000,-
0.005395683000000,-0.038637162000000,-
0.036496350000000,-
```

```
0.00539568300000000,0.107692308000000,0.0479653410000000,0.
0689655170000000;

0.0131094440000000,-
0.0180032730000000,0.0623067780000000,0.0211831570000000,0.
0372670810000000 ,0.121212121000000,0.0211831570000000,0.00
694444400000000,0.02244204900000 0,0.322580645000000;

-
0.0114467550000000,0.0729166670000000,0.0315648090000000,0.
0131545660000000 ,-
0.126805213000000,0.0405405410000000,0.0131545660000000,-
0.999508966000000,-0.0127075810000000,0.121951220000000;

0.0142033320000000,0.106796117000000,-0.0108506940000000,-
0.00549313400000000,-
0.0338846310000000,0.0324675320000000,-
0.00549313400000000,1515.85393300000,0.0783969580000000,-
0.0652173910000000; -0.00970231200000000,-
0.0417543860000000,-0.0217200530000000,-
0.0271152400000000,0.0237995820000000,-0.0283018870000000,-
0.0271152400000000,-0.999390741000000,0.0187169400000000,-
0.0465116280000000;

-0.0126682260000000,0.0252654710000000,-
0.0461986990000000,-
0.0237419350000000,0.154159869000000,0.0226537220000000,-
0.0237419350000000,-0.0516717330000000,0.0171748100000000,-
0.0731707320000000;

-
0.00121538500000000,0.0446428570000000,0.00681871600000000,
0.0021147240000 000,-
0.0155477030000000,0.107594937000000,0.00211472400000000,-
0.0160256410000000,-0.0210732980000000,-0.0263157890000000;

-0.00787110500000000,0.0769230770000000,-
0.0366651100000000,- 0.0308625690000000,-
0.113424264000000,-0.00571428600000000,-
0.0308625690000000,-0.0358306190000000,-
0.0299505280000000,- 0.0540540540000000;

0.00523210700000000,0.178571429000000,0.0729696970000000,0.
0204137180000000 ,0.0700404860000000,0.0890804600000000,0.0
```

```
204137180000000,488.864864900000,
0.00110268800000000,0;]; P=P'; % P: (N-by-T)

%%lamda =[0.410000000000000 0.240000000000000
0.350000000000000]; lamda =
optiml_portfolio_mean_varian(N,T); %given optimal portfolio
weights lamda=lamda'; ux=lamda*P(:,1); for t=2:T

  ux=[ux,lamda*P(:,t)];
end

ux=sort(ux); % rank portfolio observations due to the
timing of draws are inconsequential end
```

4.// Covar_asset_I.m

```
function [ cova ] = Covar_asset_I( I,J,N,T ) %COVAR_ASSET_I
Summary of this function goes here % Detailed explanation
goes here P = [0.0162355710000000,-0.00246812000000000,-
0.0404016810000000,- 0.0419082760000000,-
0.0737995490000000,-0.0340909090000000,-
0.0419082760000000,1645.88427300000,0.0354098360000000,-
0.184210526000000;

0.0257420960000000,-
0.00515463900000000,0.0194694570000000,0.0707015130000000,-
0.00939457200000000,0.0745098040000000,0.0707015130000000,0
.171171171000000 ,0.0232742240000000,-0.0645161290000000;

0.0265704650000000,0.0130569950000000,0.00381952700000000,-
0.00539568300000000,-0.0386371620000000,-
0.0364963500000000,-
0.00539568300000000,0.107692308000000,0.0479653410000000,0.
0689655170000000;

0.0131094440000000,-
0.0180032730000000,0.0623067780000000,0.0211831570000000,0.
0372670810000000 ,0.121212121000000,0.0211831570000000,0.00
694444400000000,0.0224420490000000 0,0.322580645000000;

-
0.0114467550000000,0.0729166670000000,0.0315648090000000,0.
0131545660000000 ,-
0.126805213000000,0.040540541000000,0.013154566000000,-
0.999508966000000,-0.012707581000000,0.121951220000000;

0.014203332000000,0.106796117000000,-0.010850694000000,-
0.005493134000000,-
0.033884631000000,0.032467532000000,-
0.005493134000000,1515.853933000000,0.078396958000000,-
0.065217391000000; -0.009702312000000,-
0.041754386000000,-0.021720053000000,-
0.027115240000000,0.023799582000000,-0.028301887000000,-
0.027115240000000,-0.999390741000000,0.018716940000000,-
0.046511628000000;

-0.012668226000000,0.025265471000000,-
0.046198699000000,-
0.023741935000000,0.154159869000000,0.022653722000000,-
0.023741935000000,-0.051671733000000,0.017174810000000,-
0.073170732000000;

-
0.001215385000000,0.044642857000000,0.006818716000000,
0.002114724000000 000,-
0.015547703000000,0.107594937000000,0.002114724000000,-
0.016025641000000,-0.021073298000000,-0.026315789000000;

-0.007871105000000,0.076923077000000,-
0.036665110000000,- 0.030862569000000,-
0.113424264000000,-0.005714286000000,-
0.030862569000000,-0.035830619000000,-
0.029950528000000,- 0.054054054000000;

0.005232107000000,0.178571429000000,0.072969697000000,0.
020413718000000 ,0.070040486000000,0.089080460000000,0.0
20413718000000,488.864864900000,
0.001102688000000,0;]; P=P'; %%P: N-by-T

r=P';
Qi=riskI_envelope( I,N,T );
rj=r(:,J);
cova=mean(rj.*Qi);
end
```

5.// CVaR_asset_I.m

```
function [ CVaR_I ] = CVaR_asset_I( I,N,T ) %UNTITLED
Summary of this function goes here % Detailed explanation
goes here risk_envlopI=riskI_envelope( I,N,T );

%%% givePortfoval=portfolio_given(N,T); % an given vector
over T periods %%% CVaR_I=(givePortfoval*(1-
risk_envlopI))/T;

end
```

6.// CVaR_optimality_condition.m

```
function [ XV ] =
CVaR_optimality_condition( N,T ) %OPTIMALITY_CONDITION
Summary of this function goes here % Detailed explanation
goes here c=Mix_CVaR_coeff( N,T); %% the matrix used for
calculate independent variables xi C=c; d=zeros(T-
1,1); Aeq=ones(1,T-1); beq=1; lb=zeros(T-1,1); ub=ones(T-
1,1); %%using 'linsolve' _Solve linear system of equations
given in matrix form %%[X,R] = linsolve(A,B); X =
lsqlin(C,d,[],[],Aeq,beq,lb,ub);

K_index=find(X>=0);
XV=[];
for i=1:numel(K_index)
     t=K_index(i);
XV=[XV,X(t)]; %% or use command XV(i)=X(t);;; end

%%%another method using the old definition of 'linsolve' by
constructing %%%symbolic matrices;;;;however, in the new
release version, no definition %%%of solver 'linsolve' for
input arguments of type 'sym';; %%%Therefore, we use 'solve'
iteratively ,see another m_file;;;

end
```

7.// mean_returnOfobservation.m

```
function [ mean_return ] =
mean_returnOfobservation( N,T) %MEAN_RETURNOFOBSERVATION
Summary of this function goes here %%N=10,T=11,P:(T-by-N) %
mean_return vector obtained from observation over T time
```

```
periods P = [0.0162355710000000,-0.00246812000000000,-
0.0404016810000000,- 0.0419082760000000,-
0.0737995490000000,-0.0340909090000000,-
0.0419082760000000,1645.88427300000,0.0354098360000000,-
0.184210526000000;

0.0257420960000000,-
0.00515463900000000,0.0194694570000000,0.0707015130000000,-
0.00939457200000000,0.0745098040000000,0.0707015130000000,0
.171171171000000 ,0.0232742240000000,-0.0645161290000000;

0.0265704650000000,0.0130569950000000,0.00381952700000000,-
0.00539568300000000,-0.0386371620000000,-
0.0364963500000000,-
0.00539568300000000,0.107692308000000,0.0479653410000000,0.
0689655170000000;

0.0131094440000000,-
0.0180032730000000,0.0623067780000000,0.0211831570000000,0.
0372670810000000 ,0.121212121000000,0.0211831570000000,0.00
694444400000000,0.022442049000000 0,0.322580645000000;

-
0.0114467550000000,0.0729166670000000,0.0315648090000000,0.
0131545660000000 ,-
0.126805213000000,0.0405405410000000,0.0131545660000000,-
0.999508966000000,-0.0127075810000000,0.121951220000000;

0.0142033320000000,0.106796117000000,-0.0108506940000000,-
0.00549313400000000,-
0.0338846310000000,0.0324675320000000,-
0.00549313400000000,1515.85393300000,0.0783969580000000,-
0.0652173910000000; -0.00970231200000000,-
0.0417543860000000,-0.0217200530000000,-
0.0271152400000000,0.0237995820000000,-0.0283018870000000,-
0.0271152400000000,-0.999390741000000,0.0187169400000000,-
0.0465116280000000;

-0.0126682260000000,0.0252654710000000,-
0.0461986990000000,-
0.0237419350000000,0.154159869000000,0.0226537220000000,-
0.0237419350000000,-0.0516717330000000,0.0171748100000000,-
0.0731707320000000;
-
0.00121538500000000,0.0446428570000000,0.00681871600000000,
0.00211472400000 000,-
0.0155477030000000,0.107594937000000,0.00211472400000000,-
0.0160256410000000,-0.0210732980000000,-0.0263157890000000;

-0.00787110500000000,0.0769230770000000,-
0.0366651100000000,- 0.0308625690000000,-
0.113424264000000,-0.00571428600000000,-
0.0308625690000000,-0.0358306190000000,-
0.0299505280000000,- 0.0540540540000000;

0.00523210700000000,0.178571429000000,0.0729696970000000,0.
0204137180000000 ,0.0700404860000000,0.0890804600000000,0.0
204137180000000,488.864864900000,
0.00110268800000000,0;]; r=P;

ri1=r(:,1);
mean_return=mean(ri1);
1;;
for i=2:N
ri=r(:,i);

% average of observations over T periods for asset

mean_return=[mean_return,mean(ri)];

end end
```

8.// Mix_CVaR_coeff.m

```
function [ c ] = Mix_CVaR_coeff( N,T ) %MIX_CVAR_COEFF
Summary of this function goes here % Detailed explanation
goes here %%% mean_return=mean_returnOfobservation( N,T);

%%minimum desired return;; % risk_free fixed saving account
interest rate %%choose fixed saving rate 2.03% for one
year, we need weekly rate;; rfm=0.0203; % interest rate for
one year

rfm=(rfm+1)^(7/365)-1 ; % weekly risk_free interst rate
chosen %%% r0=0; %CVaR_I=CVaR_asset_I( I,N,T );

%%% %%% givePortfoval=portfolio_given(N,T);
```

```
mean_givenPortfoval=mean(givePortfoval); %%% %cova=Covar_ass
et_I( I,N,T ) %%% for j=1:N

for i=1:T-
1 cova=Covar_asset_I(i,j,N,T ); CVaR_I=CVaR_asset_I( i,N,T )
; c(i,j)=(mean_return(j)-r0)*CVaR_I+(mean_givenPortfoval-
r0)*cova;

end end

end
```

9.// riskI_envelope.m

```
function [ xI ] = riskI_envelope( I,N,T ) %RISK_I_ENVELOPE
Summary of this function goes
here %%% mean_return=mean_returnOfobservation(N,I); %%% give
Portfoval=portfolio_given(N,T); % an
mean_givenPortfoval=mean(givePortfoval); %%% nAssets=numel(m
ean_return);

%%%parameters
apha=[];
for i=1:T-1         %% here, T-1=N ,denotes
given vector over T periods
the number of assets
       apha=[apha,i/T];
end

aphI=apha(I);
%%% %%% optimization using linear programming solver
f=givePortfoval; Aeq=ones(1,T); beq=1; lb=zeros(T,1); ub=(1/
aphI)*ones(T,1); [xI,
fval]=linprog(f,[],[],Aeq,beq,lb,ub); End
```

10.// risk_profile.m

```
function [ G,I_lamda ] = risk_profile( N,T ) %RISK_PROFILE
Summary of this function goes here % Detailed explanation
goes here %%desired return;; % risk_free fixed saving
account interest rate %%choose fixed saving rate 2.03% for
one year, we need weekly rate;; rfm=0.0203; % interest rate
```

```
for one year rfm=(rfm+1)^(7/365)-1 ; % weekly risk_free
interst rate
chosen %%% lamda=CVaR_optimality_condition( N,T );

% an given vector over T periods
num_lamda=numel(lamda); h=1/(num_lamda-
1); alpha=0:h:1; %%% %givePortfoval=portfolio_given(N,T); %%
%

g(1)=0;

      us=lamda./alpha;
  for i=2:num_lamda
% an given vector over T periods
%%%

      s1=sum(lamda(1:i));
      ai=alpha(i);
      s2=ai*sum(us(i:num_lamda));
      g(i)=s1+s2-ai;
end

      g(num_lamda)=0;
  xi = 0:.025:1;
  yi = interp1(alpha,g,xi);
  plot(alpha,g,'o',xi,yi);
end
```

11.// CVaR_optimality_condition_2.m

```
%%%!!!

function [ output_args ] =
CVaR_optimality_condition_2( N,T ) %UNTITLED Summary of this
function goes here % Detailed explanation goes
here %%%another method using the old definition of
'linsolve' by constructing %%%symbolic matrices;;;;however,
in the new release version, no definition %%%of solver
'linsolve' for input arguments of type 'sym';;

%%%Therefore, we use 'solve' iteratively %%%to deal with
input arguments of 'sym' type;;; %%%generating a symbolic
vector with n variables xi=[]; for i=1:T-1
```

```
t=sym(['x' int2str(i)],'positive');

xi=[xi;t];

end

S=[];
for i=1:N
       t=sym(sum(c(:,i).*xi));
S=[S;t];

end

%%additional parameters,i.e. additional requirement for
solution

adt=sym(sum((Aeq').*xi));
beq=zeros(N,1);
B=[beq;ones];
beq=1
Bt=[];
for i=1:T-1
    Bt=[Bt;zeros(1,T)]
end

adB=zeros(1,T)
adB(T)=beq;
B=[Bt;adB]
end
```

Appendix IV

1.// deviation_measure_unique.m

```
function [ x,fval ] =
deviation_measure_unique( I,N,T ) %UNTITLED Summary of this
function goes here % evaluate at i/T;;; %%desired return;; %
risk_free fixed saving account interest rate %%choose fixed
saving rate 2.03% for one year, we need weekly rate;;
rfm=0.0203; % interest rate for one year

rfm=(rfm+1)^(7/365)-1 ; % weekly risk_free interst rate
chosen %%% %%%% average of observations over T periods for N
asset; mean_return=mean_returnOfobservation( N,T);
```

```
%%% %%% %%mean_return=[rfm,mean_return]; CVaR_I=CVaR_asset_I
( I,N,T ); N;; %%% lamd_coeff=mean_return; %%% f=[zeros(1,N)
,0,zeros(1,T),1]; A1=[lamd_coeff,-1,(1/I)*ones(1,T),-
CVaR_I]; %%% P = [0.0162355710000000,-0.00246812000000000,-
0.0404016810000000,- 0.0419082760000000,-
0.0737995490000000,-0.0340909090000000,-
0.0419082760000000,1645.88427300000,0.0354098360000000,-
0.184210526000000;

0.0257420960000000,-
0.00515463900000000,0.0194694570000000,0.0707015130000000,-
0.00939457200000000,0.0745098040000000,0.0707015130000000,0
.171171171000000 ,0.0232742240000000,-0.0645161290000000;

0.0265704650000000,0.0130569950000000,0.00381952700000000,-
0.00539568300000000,-0.0386371620000000,-
0.0364963500000000,-
0.00539568300000000,0.107692308000000,0.0479653410000000,0.
0689655170000000;

0.0131094440000000,-
0.0180032730000000,0.0623067780000000,0.0211831570000000,0.
0372670810000000 ,0.121212121000000,0.0211831570000000,0.00
694444400000000,0.022442049000000 0,0.322580645000000;

-
0.0114467550000000,0.0729166670000000,0.0315648090000000,0.
013154566000000 ,-
0.126805213000000,0.0405405410000000,0.013154566000000,-
0.999508966000000,-0.0127075810000000,0.121951220000000;

0.0142033320000000,0.106796117000000,-0.0108506940000000,-
0.00549313400000000,-
0.0338846310000000,0.0324675320000000,-
0.00549313400000000,1515.85393300000,0.0783969580000000,-
0.0652173910000000; -0.00970231200000000,-
0.0417543860000000,-0.0217200530000000,-
0.0271152400000000,0.0237995820000000,-0.0283018870000000,-
0.0271152400000000,-0.999390741000000,0.0187169400000000,-
0.0465116280000000;

-0.0126682260000000,0.0252654710000000,-
0.0461986990000000,-
0.0237419350000000,0.154159869000000,0.0226537220000000,-
0.0237419350000000,-0.0516717330000000,0.0171748100000000,-
0.0731707320000000;

-
0.00121538500000000,0.0446428570000000,0.00681871600000000,
0.0021147240000 000,-
0.0155477030000000,0.107594937000000,0.00211472400000000,-
0.0160256410000000,-0.0210732980000000,-0.0263157890000000;

-0.00787110500000000,0.0769230770000000,-
0.0366651100000000,- 0.0308625690000000,-
0.113424264000000,-0.00571428600000000,-
0.0308625690000000,-0.0358306190000000,-
0.0299505280000000,- 0.0540540540000000;

%%the number of given portfolio weights is
0.00523210700000000,0.178571429000000,0.0729696970000000,0.
0204137180000000 ,0.0700404860000000,0.0890804600000000,0.0
204137180000000,488.864864900000,
0.00110268800000000,0;]; P=P';

r=P';

%%%

Z_coef=(-1)*eye(T);
A2=[r,ones(T,1),Z_coef,zeros(T,1)]; %%% A3=[zeros(T,N),zeros
(T,1),Z_coef,zeros(T,1)]; A4=[-lamd_coeff,0,zeros(1,T),0];
A5=[(-1)*eye(N),zeros(N,1),zeros(N,T),zeros(N,1)];
A=vertcat(A1,A2,A3,A4,A5);
b=vertcat(0,zeros(T,1),zeros(T,1),zeros(N,1),-rfm);
Aeq=[ones(1,N),0,zeros(1,T),0]; beq=ones; %lb=zeros(N+3,1);
%ub=ones(N+3,1); %lb(N+2:N+3)=[]; %%%% %ub(N+1:N+3)=[]; [x,f
val] = linprog(f,A,b,Aeq,beq);

end
```