Quantitative Macroeconomics

Problem Set 6
November 15, 2018

Mridula Duggal

A Initial values and exogenous process.

A.1 Initial guess.

We have to come up with initial guess for different value functions of the type $v(k, \epsilon; \bar{k}, z_s)$ for $\epsilon = \{0, 1\}$ and $z = \{good, bad\}$ assuming that:

- Agents expect that the individual and aggregate states $(\bar{k}, \epsilon, z)$ will not change in the future.
- Agents' policy is such that $g(k, \epsilon; \bar{k}, z) =$ k

The value function boils down to the following expression:

$$v(k, \epsilon; \bar{k}, z_s) = \max_{\{c,k'\}} \frac{c^{1-\gamma} - 1}{1-\gamma} + \beta \mathbb{E}\big[v(k', \epsilon'; \bar{k}', z_s')\big] \tag{1}$$

where

$$c = w(\bar{k}, z_s)\epsilon + (1 + r(\bar{k}, z_s) - \delta)k - g(\bar{k}, z_s) \tag{2}$$

The technology is given by a Cobb-Douglas aggregate production function:

$$y = z\bar{k}^{\alpha} l^{(1-\alpha)} \tag{3}$$

Then, from (3) we can recover factor prices:

$$w(\bar{k}, z_s) = (1-\alpha)z_s\big(\frac{\bar{k}}{l}\big)^{\alpha} \tag{4}$$

$$r(\bar{k}, z_s) = \alpha z_s\big(\frac{l}{\bar{k}}\big)^{1-\alpha} \tag{5}$$

Now, plug (4) and (5) in (2) and (2) in (1):

$$v(k, \epsilon; \bar{k}, z_s) = \max_{\{k'\}} \frac{\Big[\big((1-\alpha)z_s\big(\frac{\bar{k}}{l}\big)^{\alpha}\big)\epsilon + \big(1 + \alpha z_s\big(\frac{l}{\bar{k}}\big)^{1-\alpha} - \delta\big)k - g(\bar{k}, z_s)\Big]^{1-\gamma} - 1}{1-\gamma} + \beta \mathbb{E}\big[v(k', \epsilon'; \bar{k}', z_s')\big] \tag{6}$$

Given our assumptions:

- $g(.) = k$.
- $\mathbb{E}\big[v(k', \epsilon'; \bar{k}', z_s')\big] = v(k, \epsilon; \bar{k}, z_s)$

(6) reduces to:

$$v(k, \epsilon; \bar{k}, z_s) = \frac{\Big[\big((1-\alpha)z_s\big(\frac{\bar{k}}{l}\big)^{\alpha}\big)\epsilon + \big(\alpha z_s\big(\frac{l}{\bar{k}}\big)^{1-\alpha} - \delta\big)k\Big]^{1-\gamma} - 1}{(1-\gamma)(1-\beta)} \tag{7}$$

From (7) we can derive expressions for the value functions depending on both the idiosyncratic and aggregate shock:

$$v(k,1;\bar{k},z_g)=\frac{\left[\left((1-\alpha)z_g\left(\frac{\bar{k}}{1-u_g}\right)^{\alpha}\right)+\left(\alpha z_g\left(\frac{1-u_g}{\bar{k}}\right)^{1-\alpha}-\delta\right)k\right]^{1-\gamma}-1}{(1-\gamma)(1-\beta)} \tag{8}$$

$$v(k,0;\bar{k},z_g)=\frac{\left[\left(\alpha z_g\left(\frac{1-u_b}{\bar{k}}\right)^{1-\alpha}-\delta\right)k\right]^{1-\gamma}-1}{(1-\gamma)(1-\beta)} \tag{9}$$

$$v(k,1;\bar{k},z_b)=\frac{\left[\left((1-\alpha)z_b\left(\frac{\bar{k}}{1-u_g}\right)^{\alpha}\right)+\left(\alpha z_b\left(\frac{1-u_g}{\bar{k}}\right)^{1-\alpha}-\delta\right)k\right]^{1-\gamma}-1}{(1-\gamma)(1-\beta)} \tag{10}$$

$$v(k,0;\bar{k},z_b)=\frac{\left[\left(\alpha z_b\left(\frac{1-u_b}{\bar{k}}\right)^{1-\alpha}-\delta\right)k\right]^{1-\gamma}-1}{(1-\gamma)(1-\beta)} \tag{11}$$

A.2 Transition matrix

We have two idiosyncratic shocks $\epsilon=\{0,1\}$ and two aggregate shocks $z=\{bad, good\}$. Thus, the economy can be shocked by 4 differents types of shocks $\{0b,1b,0g,1g\}$. As a result, we need to build a 4x4 transition matrix. For doing so, we have used a system of 16 equations, coming from rows properties (equal to 1), the information about the duration of good and bad times, the duration of unemployment and so on. See the Python code for the details. In the end, we got the following transition matrix. Note that first row/column is for 0b, second for 1b, third 0g and fourth 1g. Lastly, notice that the higher probabilities are in the diagonal, meaning that there is a high degree of persistence:

$$\Pi=\begin{bmatrix}0.6950 & 0.1800 & 0.0809 & 0.0441\\ 0.0200 & 0.8550 & 0.0049 & 0.1200\\ 0.1241 & 0.0009 & 0.7550 & 0.1200\\ 0.0078 & 0.1171 & 0.0050 & 0.8700\end{bmatrix}$$

B Workers problem and simulation

B.1 Solution of the model by VFI.

We have tried to solve it with Python but we got flat policy functions. It seems that the budget constraint is not working properly, but we have not figured it out why yet. As an alternative, we have built a Matlab code (based on the sample code provided by the professor). By doing so, we have got the following policy functions.

Figure 1: Assets policy function for a good productivity shock.

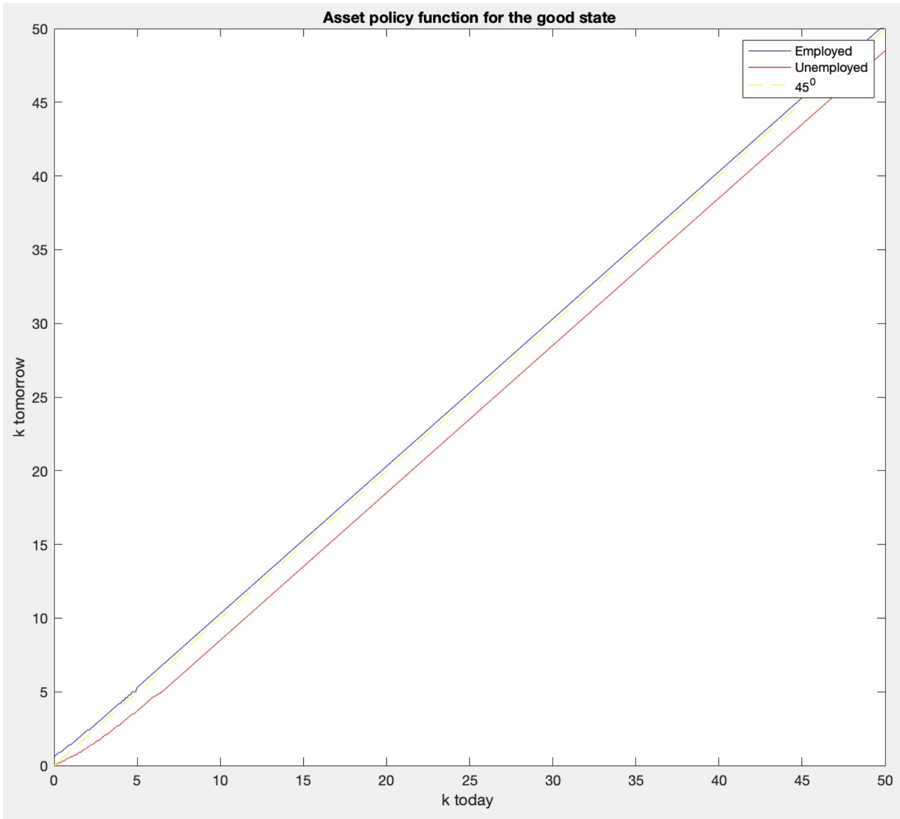
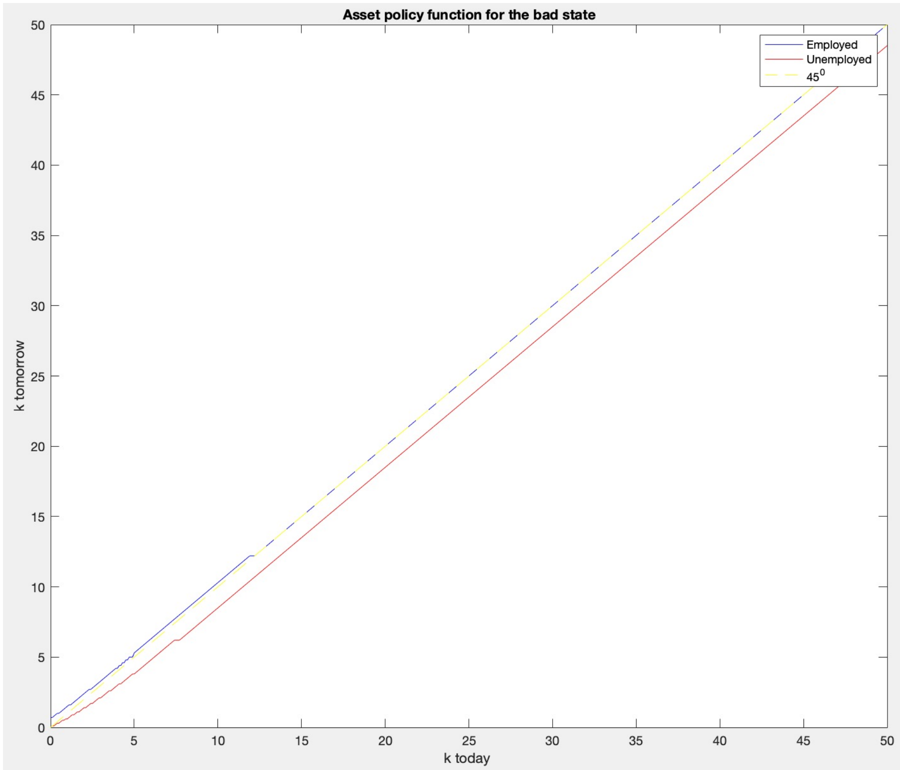


Figure 2: Assets policy function for a bad productivity shock.



When the economy is in good times, the employed agent is always accumulating assets (saving) and the unemployed guy is depleting assets. This agrees with the common sense: you save when employed and dis-save (or borrow) when unemployed. When bad times come, at some point the employed guy stops saving and just keep balance her budget. On the contrary, if a guy is shocked enough times with the unemployment, she will ends up having no assets (poverty trap), and will be borrowed constraint.

B.2 Simulation for 1000 agents and 2000 periods.

See the Matlab code.

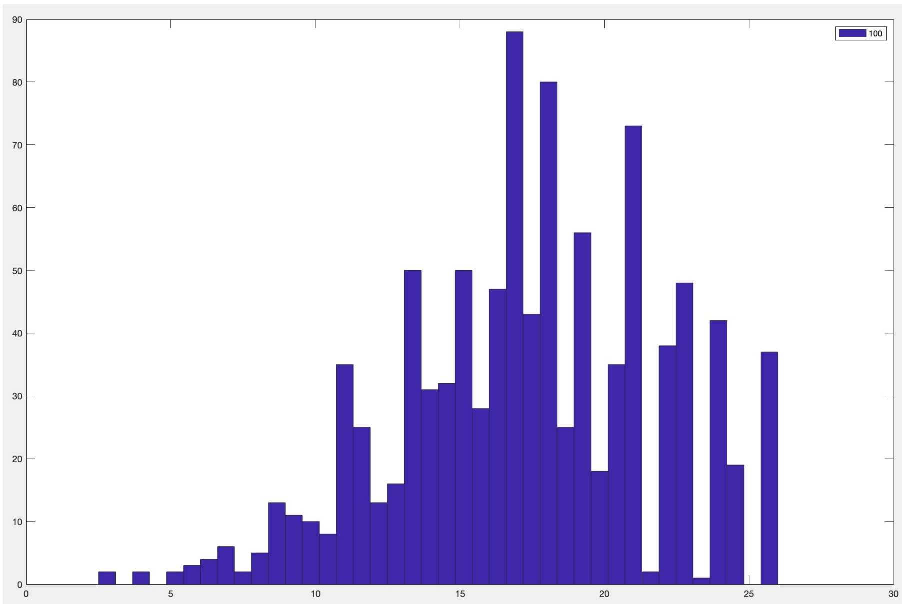
C Solution of the model

C.1 Parameter values and distribution.

For the procedure to get convergence in β's see the Matlab code. Basically, we estimate the parametrized expectations parameters by using the time series for K and z got in the previous exercise; then we update the *beta* in the function H; simulate the model again; get time series for K and z; estimate β; keep doing that until the new β were almost equal to the previous one (convergence). The goodness of the estimation is high (R^2 higher than 0.9 for both states).

Now, let's take a look at the equilibrium asset distribution. Figure 3 plot it. The average capital, which in this setup is equal to the aggregate capital is 18.28. The standard deviation is 4.75, and the skewness is 0.01. It means that the inequality is quite high and that there are more people owning a high level of assets than poors.

Figure 3: Equilibrium asset distribution.



Now, we have run the experiment of comparing the evolution of the stationary distribution after 7 consecutive periods of being in a bad state (graph 4) and 7 consecutive periods of being in a good one (graph 5). For bad periods, the sd is 5.93 and the skewness is 1.45. For good periods, the sd is 5.09 and the skewness is 0.09. Then, for bad times the inequality is higher than for good, and also there are more people having more assets.

Figure 4: Equilibrium asset distribution after 7 bad periods.

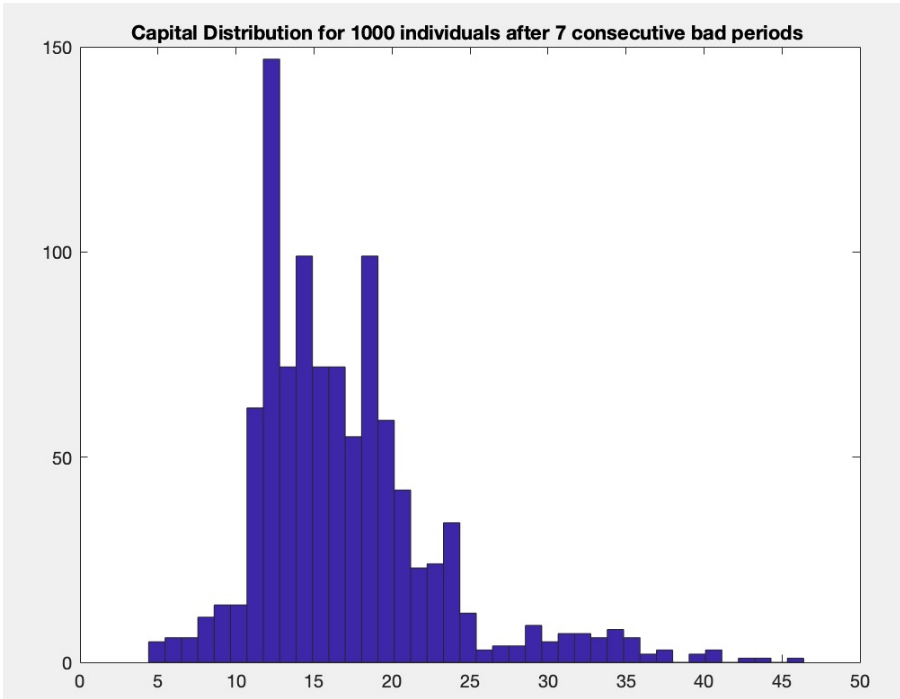
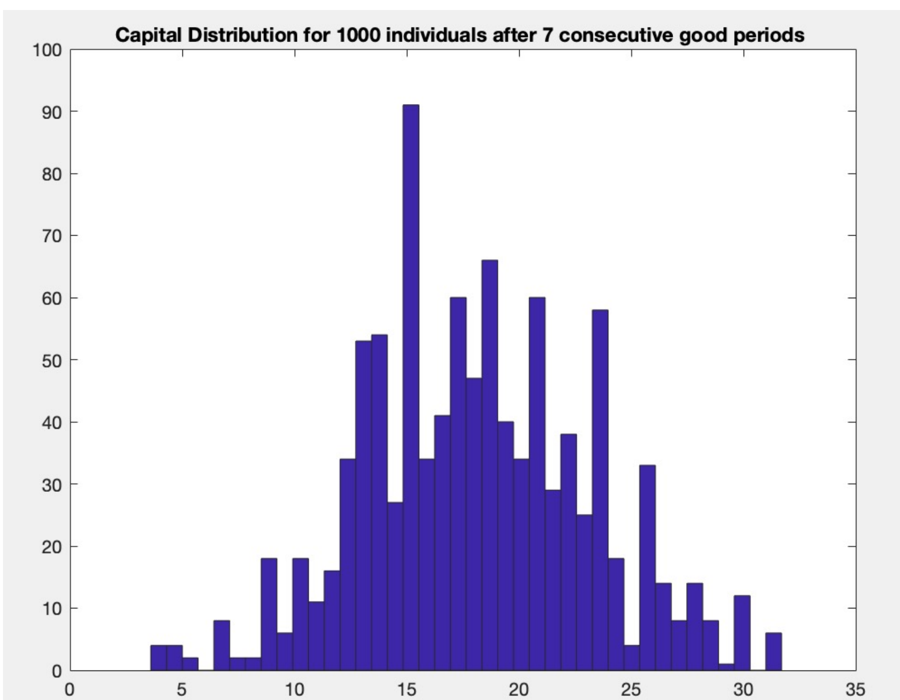


Figure 5: Equilibrium asset distribution after 7 good periods.



C.2 MATLAB Code - Baseline Kruseel-Smith

```
%% Code for solving a simplified version of K&S (1998)
% Quantitative Macroeconomics - IDEA programme

%% Initial values and parameters

%%%%%%%%%%% Finding the transition matrix for the state %%%%%%%%%%%%%

% The system of equations
A= [ 1   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0 ; ...
     0   1   0   0   0   1   0   0   0   0   0   0   0   0   0   0 ; ...
     0   0   0   0   0   0   0   0   0   0   1   0   0   0   1   0 ; ...
     0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   1 ; ...
     0   0   0   0   0   0   0   0   1   0   0   0   1   0   0   0 ; ...
     0   0   0   0   0   0   0   0   0   1   0   0   0   1   0   0 ; ...
     0   0   1   0   0   0   1   0   0   0   0   0   0   0   0   0 ; ...
     0   0   0   1   0   0   0   1   0   0   0   0   0   0   0   0 ; ...
     1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 ; ...
     0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0 ; ...
     0   0   0   0   0   0   0   0  5.6  0  -1   0   0   0   0   0 ; ...
    -1 0 28/3 0   0   0   0   0   0   0   0   0   0   0   0   0 ; ...
  .02 .48 .05 .45 0 0   0   0   0   0   0   0   0   0   0   0 ; ...
     0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0 ; ...
     0   0   0   0   0   0   0 0 .02 .48 .05 .45 0 0 0   0 ; ...
     0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0 ];


b= [7/8; 7/8; 7/8; 7/8; 1/8; 1/8; 1/8; 1/8; 7/24; 21/40; 0; 0; 0.02;
    0.005; 0.05; 0.02];


 pize = reshape(A^-1*b,4,4);



% transtion matrix aggregate state

piZ = [ 7/8   1/8;...
        1/8   7/8];


%%%%%%%%%%%   Parameters   %%%%%%%%%%%%%%%%%%%%%%%%%%

```

```
betta=0.95;
delta=0.0025;
z=[1.01 0.99];
alfa=0.36;
L=[0.96, 0.9];

%%%%%%%%%%%% Starting values for V %%%%%%%%%%%%%%%%%

v1g = @(k,K) log( alfa*z(1)*(K/L(1))^(alfa-1)*k+ (1-alfa)*z(1)*(K/L(1))^(alfa) -delta*k )/(1-betta);
v1b = @(k,K) log( alfa*z(2)*(K/L(2))^(alfa-1)*k+ (1-alfa)*z(2)*(K/L(2))^(alfa) -delta*k )/(1-betta);
v0g = @(k,K) log( alfa*z(1)*(K/L(1))^(alfa-1)*k -delta*k )/(1-betta);
v0b = @(k,K) log( alfa*z(2)*(K/L(2))^(alfa-1)*k -delta*k )/(1-betta);


%%%%%%%%%%%% Grid for k and K %%%%%%%%%%%%%%%%%%%%%%

k_grid=[0:0.1:5,5.3:0.3:50];
K_grid=[16:0.04:18.5];

% Evaluation of the VF
for j=1:size(K_grid,2)
V1g(:,j)= v1g(k_grid,K_grid(j))';
V1b(:,j)= v1b(k_grid,K_grid(j))';
V0g(:,j)= v0g(k_grid,K_grid(j))';
V0b(:,j)= v0b(k_grid,K_grid(j))';
end



%%
%%%%%% Perceived law of motion %%%%%%%%%%
% initial values

b0g=0;
b1g=1;
b0b=0;
b1b=1;

%%
for iter_b=1:1000
iter_b
% zi is the index for good shock
```

```
H=@(K,zi) exp( (b0g+b1g*log(K))*zi+ (b0b+b1b*log(K))*(1-zi) );

% approximation

Ha= @(K,zi) min(abs(K_grid-H(K,zi)));



%% Solution of the consumer problem


% Consumption for each possible decision

% e=1 employed
% g=1 good times  =2 bad times
c= @(i,I,e,g) max(alfa*z(g)*(K_grid(I)/L(g))^(alfa-1).*k_grid(i)+ ...
                (1-alfa)*z(g)*(K_grid(I)/L(g))^(alfa)*e +(1-delta)*
                    k_grid(i) ...
                - k_grid,0) ;


for iter=1:1000
for i=1:size(k_grid,2)
    for I=1:size(K_grid,2)

        % approximation next period capital

        [dif,Ip]=min(abs(K_grid-H(K_grid(I),1)));
        V0gt(i,I)= max(log(c(i,I,0,1))' + betta * ([pize(1,:)]*([V0g
            (:,Ip),V1g(:,Ip),V0b(:,Ip),V1b(:,Ip)]'))');
        V1gt(i,I)= max(log(c(i,I,1,1))' + betta * ([pize(2,:)]*([V0g
            (:,Ip),V1g(:,Ip),V0b(:,Ip),V1b(:,Ip)]'))');

        [dif,Ip]=min(abs(K_grid-H(K_grid(I),0)));
        V0bt(i,I)= max(log(c(i,I,0,2))' + betta * ([pize(3,:)]*([V0g
            (:,Ip),V1g(:,Ip),V0b(:,Ip),V1b(:,Ip)]'))');
        V1bt(i,I)= max(log(c(i,I,1,2))' + betta * ([pize(4,:)]*([V0g
            (:,Ip),V1g(:,Ip),V0b(:,Ip),V1b(:,Ip)]'))');

    end

end

```

```
dev= max(max(abs( [V0gt-V0g,V1gt-V1g,V0bt-V0b,V1bt-V1b])));

if dev<0.00001
    break
else
    V0g=V0gt;
    V1g=V1gt;
    V0b=V0bt;
    V1b=V1bt;
end

end


% Recover the policy function


for i=1:size(k_grid,2)
    for I=1:size(K_grid,2)

        % approximation next period capital

        [dif,Ip]=min(abs(K_grid-H(K_grid(I),1)));
        [V0gt(i,I),a(i,I,2,1)]= max(log(c(i,I,0,1))' + betta * ([pize(1,:)]*([V0g(:,Ip),V1g(:,Ip),V0b(:,Ip),V1b(:,Ip)]'))');
        [V1gt(i,I),a(i,I,1,1)]= max(log(c(i,I,1,1))' + betta * ([pize(2,:)]*([V0g(:,Ip),V1g(:,Ip),V0b(:,Ip),V1b(:,Ip)]'))');

        [dif,Ip]=min(abs(K_grid-H(K_grid(I),0)));
        [V0bt(i,I),a(i,I,2,2)]= max(log(c(i,I,0,2))' + betta * ([pize(3,:)]*([V0g(:,Ip),V1g(:,Ip),V0b(:,Ip),V1b(:,Ip)]'))');
        [V1bt(i,I),a(i,I,1,2)]= max(log(c(i,I,1,2))' + betta * ([pize(4,:)]*([V0g(:,Ip),V1g(:,Ip),V0b(:,Ip),V1b(:,Ip)]'))');

    end

end

%Store the policy functions.
   assets(:,:,2,1) = a(i,I,2,1); %employed, bad
   assets(:,:,1,1)= a(i,I,1,1); % unemployed, bad
   assets(:,:,2,2) = a(i,I,2,2); % employed, good
   assets(:,:,1,2)= a(i,I,1,2); %unemployed, good

```

```
%% Simulation


% A sequence of TFP
% using the index =1 good ,  =2 bad

if iter_b==1

zt(1)=1;

for t=2:2000
     draw=rand;
     zt(t)= 1+(rand>=piZ(zt(t-1),1));
end
% Splitting the sample for good and bad times

% "burning" the first 200 periods
ztb=zt;
ztb(1:200)=0;
% Construct an index for the good times
i_zg=find(zt==1);

% Construct an index for the bad times
i_zb=find(zt==2);

% initial distribution of assets and employment
% =1 employed
N_state(1:960,:,1)=ones(960,1)*[26,1];
% =2 unemployed
N_state(961:1000,:,1)=ones(40,1)*[26,2];

K_sim(1)=find(K_grid == 17); %The index is for the generated capital from the simulated series.

%%%%%%%%%%%%%%%% Simulating the Distributions %%%%%%%%%%%%%%%%%%%%%%%%

for t=2:2000
for n=1:1000

% Evolution of assets
     N_state(n,1,t)=a(N_state(n,1,t-1),K_sim(t-1),N_state(n,2,t-1),zt(t-1));
% Evolution of the employment status
```

```
      N_state(n,2,t)= 2-(rand>=pize(1 + zt(t-1)*2 - N_state(n,2,t-1),zt(t)*2-1)/piZ(zt(t-1),zt(t)));


end

% Storage of the sequence of aggregate capital
[dev2, K_sim(t)]=min(abs(k_grid(round(mean(N_state(:,1,t))))-K_grid));


end

else


for t=2:2000
for n=1:1000

% Evolution of assets
      N_state(n,1,t)=a(N_state(n,1,t-1),K_sim(t-1),N_state(n,2,t-1),zt(t-1));

end

% Storage of the sequence of aggregate capital
[dev2, K_sim(t)]=min(abs(k_grid(round(mean(N_state(:,1,t))))-K_grid));


end

end

% Regression model for the evolution of aggregate capital

% regression for good times (burning the first 20 periods of g times)

Yg=log(K_grid(K_sim(i_zg(20:end)))');
Xg=[ones(size(i_zg(20:end),2),1),log(K_grid(K_sim(i_zg(20:end)-1))')];
Bg=Xg\Yg
b0gp=Bg(1);
b1gp=Bg(2);
```

```
% regression for bad times (burning the first 20 periods of bad times.

Yb=log(K_grid(K_sim(i_zb(20:end)))');
Xb=[ones(size(i_zb(20:end),2),1),log(K_grid(K_sim(i_zb(20:end)-1))')];
Bb=Xb\Yb
b0bp=Bb(1);
b1bp=Bb(2);


dev_b=max(abs([b0g-b0gp b1g-b1gp b0b-b0bp b1b-b1bp]))

pause(1)
if dev_b<=0.01
    break
end

b0g=0.1*b0gp+0.9*b0g;
b1g=0.1*b1gp+0.9*b1g;
b0b=0.1*b0bp+0.9*b0b;
b1b=0.1*b1bp+0.9*b1b;


end

%% Number of Iterations
disp('Number of Iterations')
disp(iter_b)

%% Mean of the Simulated series
mean_s = mean(K_grid(K_sim));
disp('Simulated series Aggregate/Average Capital is')
disp(mean_s)

%% R2 for the regression

mdlg = fitlm(Xg(:,2),Yg);
R2g = mdlg.Rsquared.Ordinary;
stdg= mdlg.RMSE;
disp('R2 for the good shock')
disp(R2g)

mdlb = fitlm(Xb(:,2),Yb);
```

```
R2b = mdlb.Rsquared.Ordinary;
stdb= mdlb.RMSE;
disp('R2 for the bad shock')
disp(R2b)
%% Graphs for the Policy Functions and Asset Distribution

% Evolution of the Assets distribution
%Figure
for t_ind=1:100

  hist(k_grid(reshape(N_state(:,1,t_ind),1,1000)),40)
  legend(num2str(t_ind))
  pause(1)

end

%% Statistics for the distriburion
W = k_grid(reshape(N_state(:,1,t_ind),1,1000));

mean_w = mean(W);
disp('Mean of Asset Distribution')
disp(mean_w)

std_w = std(W);
disp('Standard Deviation of Asset Distribution')
disp(std_w)

skew_w=skewness(W);
disp('Skewness of Asset Distribution');
disp(skew_w)

%% Retrieve the Policy function from the indices.

G0g = k_grid(a(:,26,2,1)); % unemployed, good
G1g = k_grid(a(:,26,1,1)); % employed, good
G0b = k_grid(a(:,26,2,2)); % unemployed, bad
G1b = k_grid(a(:,26,1,2)); % employed, bad

figure(1)
plot(k_grid, G1b,'b' )
hold on
plot(k_grid, G0b, 'r' )
hold on
plot(k_grid,k_grid , 'y--')
```

```
title ('Asset policy function for the bad state')
legend('Employed', 'Unemployed', '45^0')
xlabel('k today ')
ylabel('k tomorrow')
hold off

figure(2)
plot(k_grid, G1g, 'b' )
hold on
plot(k_grid, G0g, 'r' )
hold on
plot(k_grid,k_grid , 'y--')
title ('Asset policy function for the good state')
legend('Employed', 'Unemployed', '45^0')
xlabel('k today ')
ylabel('k tomorrow')
hold off

%% Employment distribution

for t_ind=1:100
    hist(reshape(N_state(:,2,t_ind),1,1000),40)
    title('Employment Distribution for 1000 individuals after 100 periods ')
    xticks([1 2])
    xticklabels({'Employed','Unemployed'})
end
print -dpdf histe_fig6.eps
%% Compare the asset distribution in equilibrium after 7 periods of being in a bad state (low z) as opposed
%to being in the high state.

grouped_b = mat2cell( i_zb, 1, diff( [0, find(diff(i_zb) ~= 1), length(i_zb)] )) ;

for i=1:size(grouped_b,2)
    if size(grouped_b{i},2)==7
     g7b = grouped_b{i};
     break
    end
end

for t=g7b(1):g7b(7)
hist(k_grid(reshape(N_state(:,1,t),1,1000)),40);
```

```
title('Capital Distribution for 1000 individuals after 7 consecutive
    bad periods ')
end
print -dpdf hist7b_fig7.eps

W1=(k_grid(reshape(N_state(:,1,t),1,1000)))
mean_w1 = mean(W1);
disp('Mean of Asset Distribution')
disp(mean_w1)

std_w1 = std(W1);
disp('Standard Deviation of Asset Distribution')
disp(std_w1)

skew_w1 =skewness(W1);
disp('Skewness of Asset Distribution');
disp(skew_w1)

%% For good
grouped_g = mat2cell( i_zg , 1, diff( [0, find(diff(i_zg) ~= 1),
    length(i_zg)] )) ;

for i=1:size(grouped_g,2)
    if size(grouped_g{i},2)==7
     g7g = grouped_g{i};
     break
    end
end


for t=g7g(1):g7g(7)
hist(k_grid(reshape(N_state(:,1,t),1,1000)),40);
title('Capital Distribution for 1000 individuals after 7 consecutive
    good periods ')
end
print -dpdf hist7g_fig8.eps

W2=(k_grid(reshape(N_state(:,1,t),1,1000)));
mean_w2 = mean(W1);
disp('Mean of Asset Distribution')
disp(mean_w2)

std_w2 = std(W2);
disp('Standard Deviation of Asset Distribution')
```

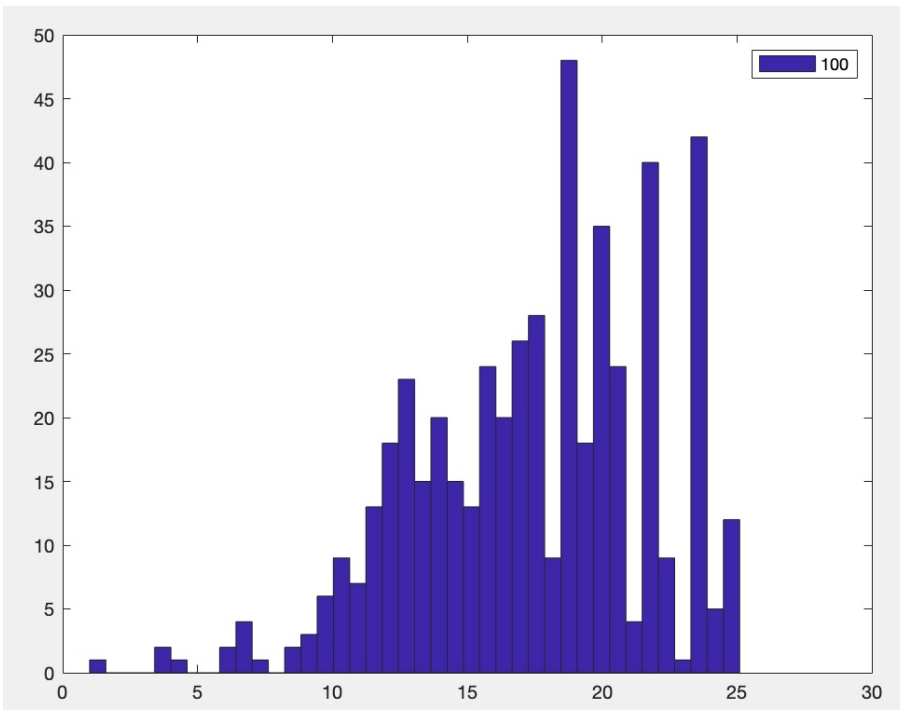
```
disp(std_w2)

skew_w2 =skewness(W2);
disp('Skewness of Asset Distribution');
disp(skew_w2)
```

C.3 Heterogeneous expectations.

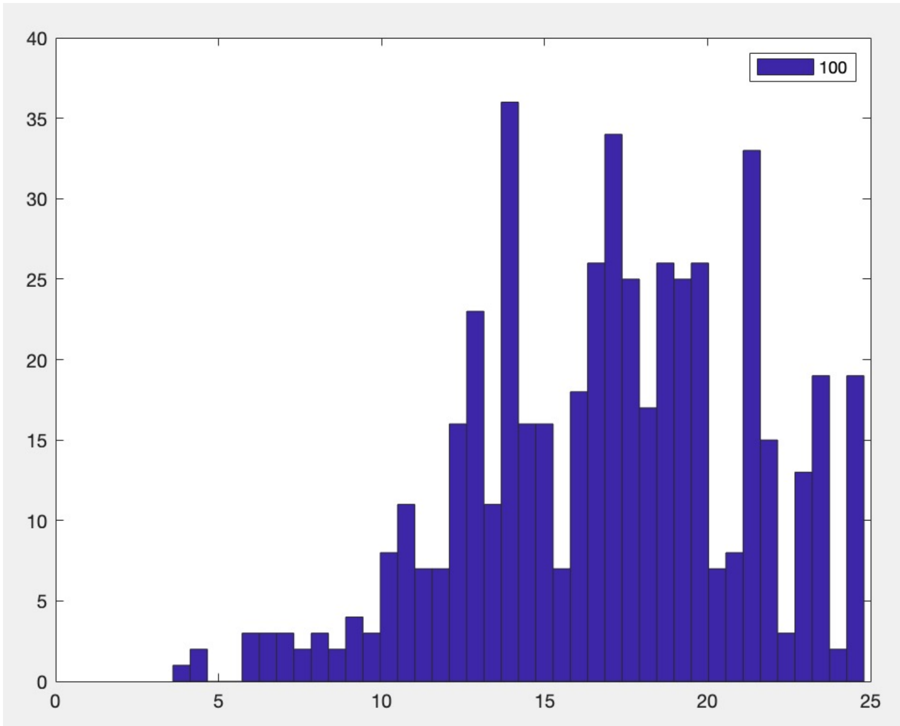
See the Matlab code. We took the assets distribution as an aggregate approximation of the welfare. Thus, figure 6 pictures the distribution for updaters and figure 7 for non-updaters.

Figure 6: Equilibrium asset distribution for updaters



The main differences are that (1) there are more poors between the non-updaters and (2) the average level of wealth is higher for updaters. Thus, the social welfare is going to be lower. Then there are no incentives for a non-updating behavior.

Figure 7: Equilibrium asset distribution for non-updaters




```
%% Code for solving a simplified version of K&S (1998)
% Quantitative Macroeconomics - IDEA programme

%% Initial values and parameters

%%%%%%%%%%% Finding the transition matrix for the state
    %%%%%%%%%%%%%

% The system of equations
A= [ 1   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0  ;  ...
     0   1   0   0   0   1   0   0   0   0   0   0   0   0   0   0  ;  ...
     0   0   0   0   0   0   0   0   0   0   1   0   0   0   1   0  ;  ...
     0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   1  ;  ...
     0   0   0   0   0   0   0   0   1   0   0   0   1   0   0   0  ;  ...
     0   0   0   0   0   0   0   0   0   1   0   0   0   1   0   0  ;  ...
     0   0   1   0   0   0   1   0   0   0   0   0   0   0   0   0  ;  ...
     0   0   0   1   0   0   0   1   0   0   0   0   0   0   0   0  ;  ...
     1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  ;  ...
     0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0  ;  ...
     0   0   0   0   0   0   0   0  5.6  0  -1   0   0   0   0   0  ;  ...
    -1 0 28/3 0   0   0   0   0   0   0   0   0   0   0   0   0  ;  ...
  .02  .48  .05  .45 0 0   0   0   0   0   0   0   0   0   0   0  ;  ...
     0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0  ;  ...
```

```
      0   0   0   0   0   0   0 0 .02 .48 .05 .45 0 0 0   0 ; ...
      0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0 ];


b= [7/8; 7/8; 7/8; 7/8; 1/8; 1/8; 1/8; 1/8; 7/24; 21/40; 0; 0; 0.02; 0.005; 0.05; 0.02];


pize = reshape(A^-1*b,4,4);



% transtion matrix aggregate state

piZ = [ 7/8   1/8;...
        1/8   7/8];


%%%%%%%%%%% Parameters  %%%%%%%%%%%%%%%%%%%%%%%

betta=0.95;
delta=0.0025;
z=[1.01 0.99];
alfa=0.36;
L=[0.96, 0.9];

%%%%%%%%%%% Starting values for V %%%%%%%%%%%%%%%%

v1g = @(k,K) log( alfa*z(1)*(K/L(1))^(alfa-1)*k+ (1-alfa)*z(1)*(K/L(1))^(alfa) -delta*k )/(1-betta);
v1b = @(k,K) log( alfa*z(2)*(K/L(2))^(alfa-1)*k+ (1-alfa)*z(2)*(K/L(2))^(alfa) -delta*k )/(1-betta);
v0g = @(k,K) log( alfa*z(1)*(K/L(1))^(alfa-1)*k -delta*k )/(1-betta);
v0b = @(k,K) log( alfa*z(2)*(K/L(2))^(alfa-1)*k -delta*k )/(1-betta);


%%%%%%%%%%% Grid for k and K %%%%%%%%%%%%%%%%%%%%%%

k_grid=[0:0.1:5,5.3:0.3:50];
K_grid=[16:0.04:18.5];

% Evaluation of the VF
for j=1:size(K_grid,2)
V1g(:,j)= v1g(k_grid,K_grid(j))';
```

```
V1b(:,j)= v1b(k_grid,K_grid(j))';
V0g(:,j)= v0g(k_grid,K_grid(j))';
V0b(:,j)= v0b(k_grid,K_grid(j))';
end



%%
%%%%% Perceived law of motion  %%%%%%%%%%
% initial values

b0g=0;
b1g=1;
b0b=0;
b1b=1;

%%
for iter_b=1:1000
iter_b
% zi is the index for good shock
H=@(K,zi) exp( (b0g+b1g*log(K))*zi+ (b0b+b1b*log(K))*(1-zi) );

% approximation

Ha= @(K,zi) min(abs(K_grid-H(K,zi)));




%% Solution of the consumer problem


% Consumption for each possible decision

% e=1 employed
% g=1 good times  =2 bad times
c= @(i,I,e,g) max(alfa*z(g)*(K_grid(I)/L(g))^(alfa-1).*k_grid(i)+ ...
                 (1-alfa)*z(g)*(K_grid(I)/L(g))^(alfa)*e +(1-delta)*k_grid(i) ...
                 - k_grid,0) ;


for iter=1:1000
for i=1:size(k_grid,2)
```

```
    for I=1:size(K_grid,2)

        % approximation next period capital

        [dif,Ip]=min(abs(K_grid-H(K_grid(I),1)));
        V0gt(i,I)= max(log(c(i,I,0,1))' + betta * ([pize(1,:)]*([V0g(:,Ip),V1g(:,Ip),V0b(:,Ip),V1b(:,Ip)]'))');
        V1gt(i,I)= max(log(c(i,I,1,1))' + betta * ([pize(2,:)]*([V0g(:,Ip),V1g(:,Ip),V0b(:,Ip),V1b(:,Ip)]'))');

        [dif,Ip]=min(abs(K_grid-H(K_grid(I),0)));
        V0bt(i,I)= max(log(c(i,I,0,2))' + betta * ([pize(3,:)]*([V0g(:,Ip),V1g(:,Ip),V0b(:,Ip),V1b(:,Ip)]'))');
        V1bt(i,I)= max(log(c(i,I,1,2))' + betta * ([pize(4,:)]*([V0g(:,Ip),V1g(:,Ip),V0b(:,Ip),V1b(:,Ip)]'))');

    end

end

dev= max(max(abs( [V0gt-V0g,V1gt-V1g,V0bt-V0b,V1bt-V1b])));

if dev<0.00001
    break
else
    V0g=V0gt;
    V1g=V1gt;
    V0b=V0bt;
    V1b=V1bt;
end

end


% Recover the policy function


for i=1:size(k_grid,2)
    for I=1:size(K_grid,2)

        % approximation next period capital

        [dif,Ip]=min(abs(K_grid-H(K_grid(I),1)));
        [V0gt(i,I),a(i,I,2,1)]= max(log(c(i,I,0,1))' + betta * ([pize
```

```
              (1,:)]*([V0g(:,Ip),V1g(:,Ip),V0b(:,Ip),V1b(:,Ip)]'))');
          [V1gt(i,I),a(i,I,1,1)]= max(log(c(i,I,1,1))' + betta * ([pize
              (2,:)]*([V0g(:,Ip),V1g(:,Ip),V0b(:,Ip),V1b(:,Ip)]'))');

          [dif,Ip]=min(abs(K_grid-H(K_grid(I),0)));
          [V0bt(i,I),a(i,I,2,2)]= max(log(c(i,I,0,2))' + betta * ([pize
              (3,:)]*([V0g(:,Ip),V1g(:,Ip),V0b(:,Ip),V1b(:,Ip)]'))');
          [V1bt(i,I),a(i,I,1,2)]= max(log(c(i,I,1,2))' + betta * ([pize
              (4,:)]*([V0g(:,Ip),V1g(:,Ip),V0b(:,Ip),V1b(:,Ip)]'))');

    end

end

%Store the policy functions.
   assets(:,:,2,1) = a(i,I,2,1); %employed, bad
   assets(:,:,1,1)= a(i,I,1,1); % unemployed, bad
   assets(:,:,2,2) = a(i,I,2,2); % employed, good
   assets(:,:,1,2)= a(i,I,1,2); %unemployed, good

if iter==1
    a_NOupdate = a;
end
%% Simulation


% A sequence of TFP
% using the index =1 good ,  =2 bad

if iter_b==1

zt(1)=1;

for t=2:2000
    draw=rand;
    zt(t)= 1+(rand>=piZ(zt(t-1),1));
end
% Splitting the sample for good and bad times

% "burning" the first 200 periods
ztb=zt;
ztb(1:200)=0;
% Construct an index for the good times
i_zg=find(zt==1);
```

```
% Construct an index for the bad times
i_zb=find(zt==2);

% initial distribution of assets and employment
% =1 employed
N_state(1:960,:,1)=ones(960,1)*[26,1];
% =2 unemployed
N_state(961:1000,:,1)=ones(40,1)*[26,2];

K_sim(1)=find(K_grid == 17); %The index is for the generated capital from the simulated series.

%%%%%%%%%%%%%%%%%% Simulating the Distributions %%%%%%%%%%%%%%%%%%%%%%%%%%%

for t=2:2000
for n=1:1000
    if n<500
% Evolution of assets
    N_state(n,1,t)=a_NOupdate(N_state(n,1,t-1),K_sim(t-1),N_state(n,2,t-1),zt(t-1));
% Evolution of the employment status
    N_state(n,2,t)= 2-(rand>=pize(1 + zt(t-1)*2 - N_state(n,2,t-1),zt(t)*2-1)/piZ(zt(t-1),zt(t)));
    else
    % Evolution of assets
    N_state(n,1,t)=a_NOupdate(N_state(n,1,t-1),K_sim(t-1),N_state(n,2,t-1),zt(t-1));
% Evolution of the employment status
    N_state(n,2,t)= 2-(rand>=pize(1 + zt(t-1)*2 - N_state(n,2,t-1),zt(t)*2-1)/piZ(zt(t-1),zt(t)));
    end

end

% Storage of the sequence of aggregate capital
[dev2, K_sim(t)]=min(abs(k_grid(round(mean(N_state(:,1,t))))-K_grid));


end

else
```

```


for t=2:2000
for n=1:1000
    if n<=500
% Evolution of assets
    N_state(n,1,t)=a_NOupdate(N_state(n,1,t-1),K_sim(t-1),N_state(n,2,t-1),zt(t-1));
    N_state_NU(n,1,t)=a_NOupdate(N_state(n,1,t-1),K_sim(t-1),N_state(n,2,t-1),zt(t-1));
    else
    N_state(n,1,t)=a(N_state(n,1,t-1),K_sim(t-1),N_state(n,2,t-1),zt(t-1));
    end
end

% Storage of the sequence of aggregate capital
[dev2, K_sim(t)]=min(abs(k_grid(round(mean(N_state(:,1,t))))-K_grid));


end

end

% Regression model for the evolution of aggregate capital

% regression for good times (burning the first 20 periods of g times)

Yg=log(K_grid(K_sim(i_zg(20:end)))');
Xg=[ones(size(i_zg(20:end),2),1),log(K_grid(K_sim(i_zg(20:end)-1))')];
Bg=Xg\Yg
b0gp=Bg(1);
b1gp=Bg(2);
% regression for bad times (burning the first 20 periods of bad times.

Yb=log(K_grid(K_sim(i_zb(20:end)))');
Xb=[ones(size(i_zb(20:end),2),1),log(K_grid(K_sim(i_zb(20:end)-1))')];
Bb=Xb\Yb
b0bp=Bb(1);
b1bp=Bb(2);
```

```
dev_b=max(abs([b0g-b0gp b1g-b1gp b0b-b0bp b1b-b1bp]))

pause(1)
if dev_b<=0.01
    break
end

b0g=0.1*b0gp+0.9*b0g;
b1g=0.1*b1gp+0.9*b1g;
b0b=0.1*b0bp+0.9*b0b;
b1b=0.1*b1bp+0.9*b1b;


end

%% Number of Iterations
disp('Number of Iterations')
disp(iter_b)

%% Mean of the Simulated series
mean_s = mean(K_grid(K_sim));
disp('Simulated series Aggregate/Average Capital is')
disp(mean_s)

%% R2 for the regression

mdlg = fitlm(Xg(:,2),Yg);
R2g = mdlg.Rsquared.Ordinary;
stdg= mdlg.RMSE;
disp('R2 for the good shock')
disp(R2g)

mdlb = fitlm(Xb(:,2),Yb);
R2b = mdlb.Rsquared.Ordinary;
stdb= mdlb.RMSE;
disp('R2 for the bad shock')
disp(R2b)
%% Graphs for the Policy Functions and Asset Distribution

% Evolution of the Assets distribution
%Figure
for t_ind=1:100
```

```
   hist(k_grid(reshape(N_state(:,1,t_ind),1,1000)),40)
   legend(num2str(t_ind))
   pause(1)

end

%% Statistics for the distriburion
W = k_grid(reshape(N_state(:,1,t_ind),1,1000));

mean_w = mean(W);
disp('Mean of Asset Distribution')
disp(mean_w)

std_w = std(W);
disp('Standard Deviation of Asset Distribution')
disp(std_w)

skew_w=skewness(W);
disp('Skewness of Asset Distribution');
disp(skew_w)

%% Retrieve the Policy function from the indices.

G0g = k_grid(a(:,26,2,1)); % unemployed, good
G1g = k_grid(a(:,26,1,1)); % employed, good
G0b = k_grid(a(:,26,2,2)); % unemployed, bad
G1b = k_grid(a(:,26,1,2)); % employed, bad

figure(1)
plot(k_grid, G1b,'b' )
hold on
plot(k_grid, G0b, 'r' )
hold on
plot(k_grid,k_grid , 'y—')
title ('Asset policy function for the bad state')
legend('Employed', 'Unemployed', '45^0')
xlabel('k today ')
ylabel('k tomorrow')
hold off

figure(2)
plot(k_grid, G1g, 'b' )
hold on
```

```
plot(k_grid, G0g, 'r' )
hold on
plot(k_grid,k_grid , 'y--')
title ('Asset policy function for the good state')
legend('Employed', 'Unemployed', '45^0')
xlabel('k today ')
ylabel('k tomorrow')
hold off

%% Employment distribution

for t_ind=1:100
    hist(reshape(N_state(:,2,t_ind),1,1000),40)
    title('Employment Distribution for 1000 individuals after 100 periods ')
    xticks([1 2])
    xticklabels({'Employed','Unemployed'})
end
print -dpdf histe_fig6.eps
%% Compare the asset distribution in equilibrium after 7 periods of being in a bad state (low z) as opposed
%to being in the high state.

grouped_b = mat2cell( i_zb, 1, diff( [0, find(diff(i_zb) ~= 1), length(i_zb)] )) ;

for i=1:size(grouped_b,2)
    if size(grouped_b{i},2)==7
     g7b = grouped_b{i};
     break
    end
end

for t=g7b(1):g7b(7)
hist(k_grid(reshape(N_state(:,1,t),1,1000)),40);
title('Capital Distribution for 1000 individuals after 7 consecutive bad periods ')
end
print -dpdf hist7b_fig7.eps

%% For good
grouped_g = mat2cell( i_zg, 1, diff( [0, find(diff(i_zg) ~= 1), length(i_zg)] )) ;

```

```
for i=1:size(grouped_g,2)
    if size(grouped_g{i},2)==7
     g7g = grouped_g{i};
     break
    end
end


for t=g7g(1):g7g(7)
hist(k_grid(reshape(N_state(:,1,t),1,1000)),40);
title('Capital Distribution for 1000 individuals after 7 consecutive good periods ')
end
print -dpdf hist7g_fig8.eps

%% Assets for those who do not update
G0gNU = k_grid(a_NOupdate(:,26,2,1)); % unemployed, good
G1gNU = k_grid(a_NOupdate(:,26,1,1)); % employed, good
G0bNU = k_grid(a_NOupdate(:,26,2,2)); % unemployed, bad
G1bNU = k_grid(a_NOupdate(:,26,1,2)); % employed, bad

figure(1)
plot(k_grid, G1bNU,'b' )
hold on
plot(k_grid, G0bNU, 'r' )
hold on
plot(k_grid,k_grid , 'y--')
title ('Asset policy function for the bad state')
legend('Employed', 'Unemployed', '45^0')
xlabel('k today ')
ylabel('k tomorrow')
hold off

figure(2)
plot(k_grid, G1gNU, 'b' )
hold on
plot(k_grid, G0gNU, 'r' )
hold on
plot(k_grid,k_grid , 'y--')
title ('Asset policy function for the good state')
legend('Employed', 'Unemployed', '45^0')
xlabel('k today ')
ylabel('k tomorrow')
hold off
```

```
for  t_ind=1:100

  hist(k_grid(reshape(N_state(1:500,1,t_ind),1,500)),40)
  legend(num2str(t_ind))
  pause(1)

end

for  t_ind=1:100

  hist(k_grid(reshape(N_state(501:1000,1,t_ind),1,500)),40)
  legend(num2str(t_ind))
  pause(1)

end
```