

February 19, 2019

---

## Empirical Methods HA 7

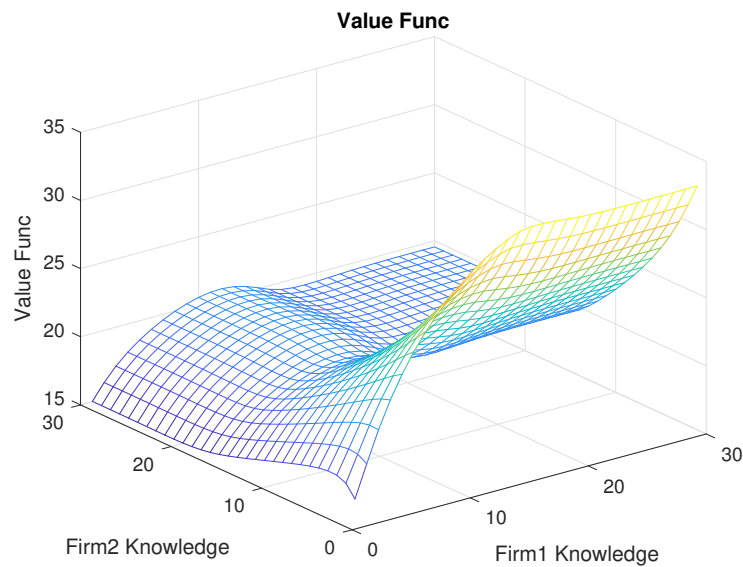
Konstantin Guryev

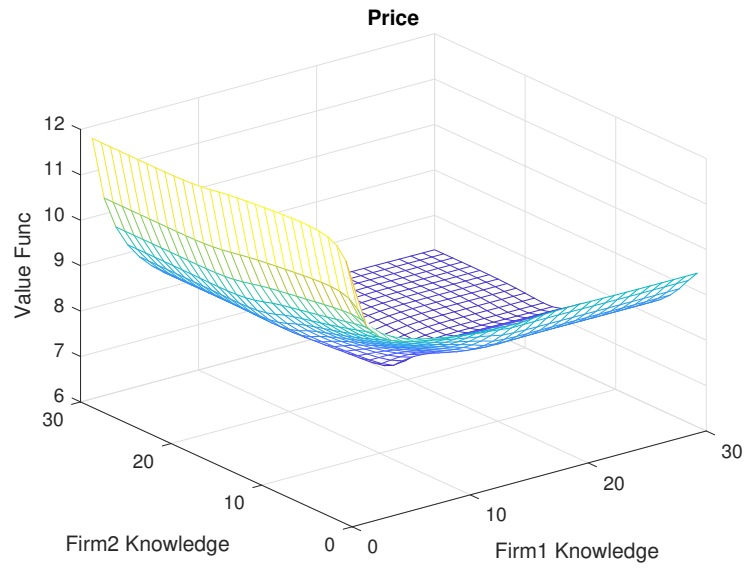
Pennsylvania State University

2019

### Question 1

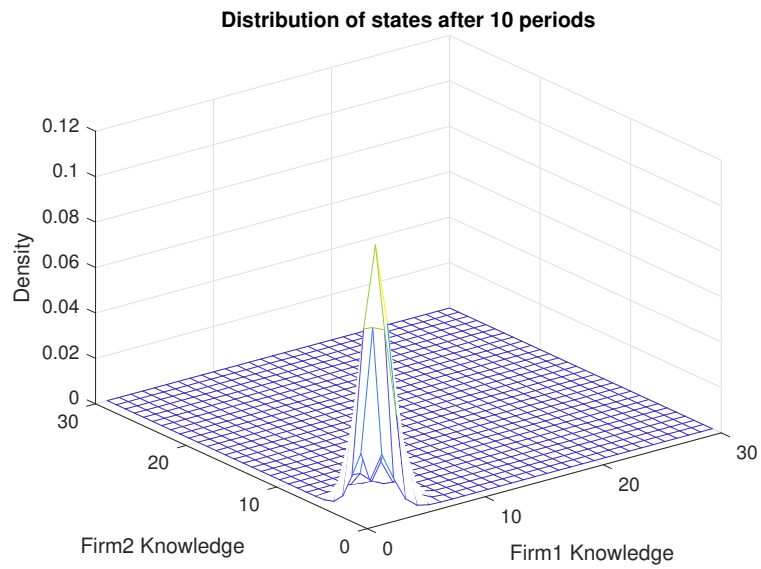
The proposed algorithm converges within 303 seconds and on the 82<sup>th</sup> iteration for  $\lambda = 1$ , i.e. in the case when the dampening is not used. In case when  $\lambda = 0.5$  the number of iterations is twice larger, namely 165, and the time of convergence is 598 seconds.

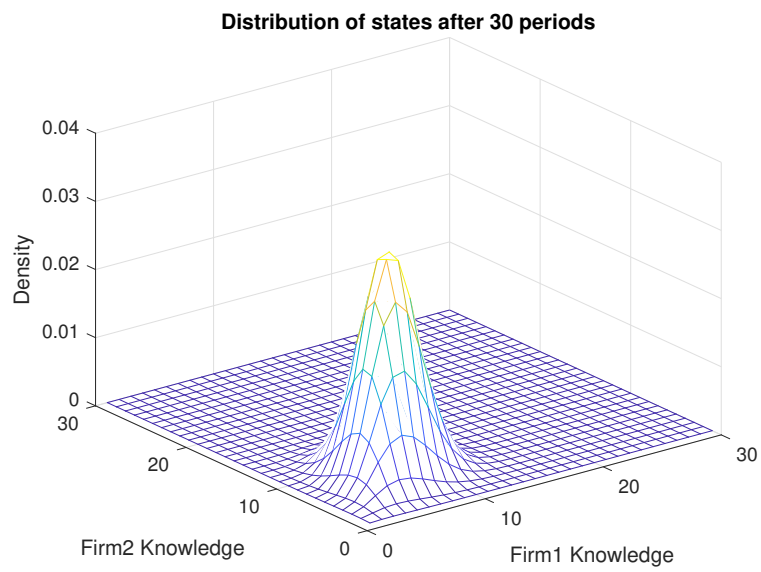
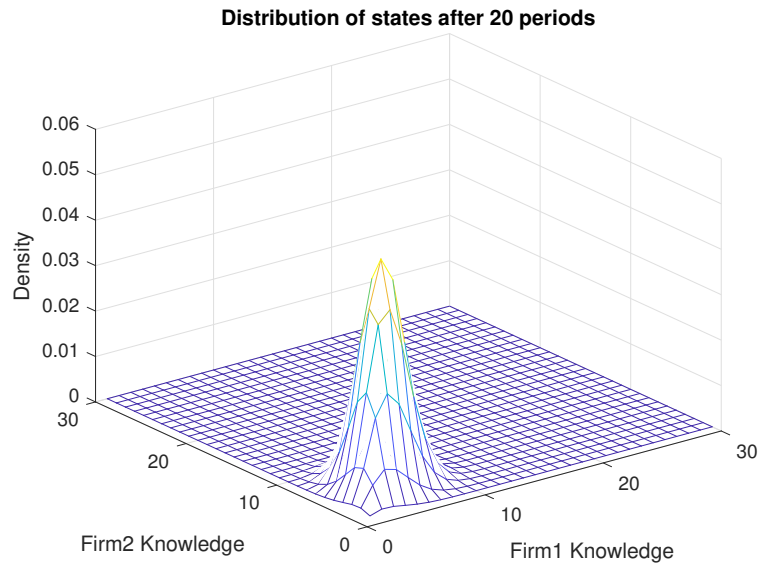




## Question 2

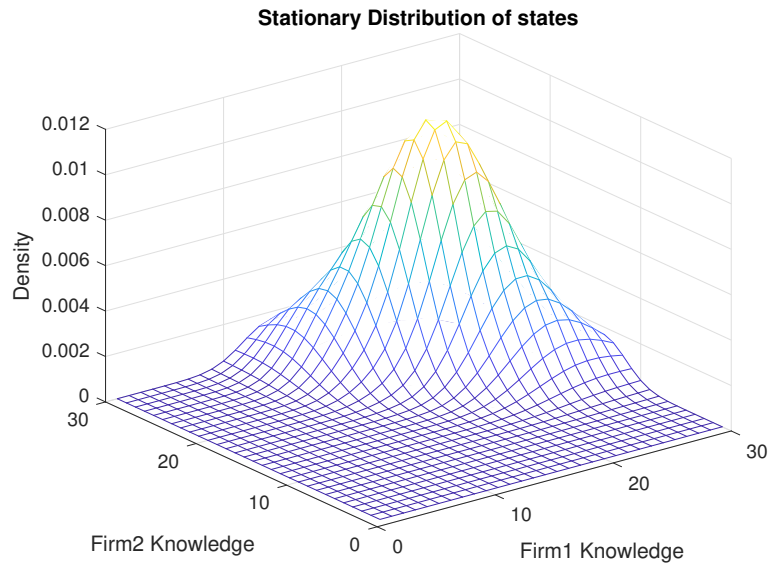
To tackle a problem a 900 by 900 matrix is constructed in which each row contains the probability to going to one of any 900 consequent periods.





### Question 3

To obtain the steady state distribution I just raised the matrix to the power of 900 (relatively big number) and reshaped the rows to be 30 by 30.



## Matlab Code

```

1  global l L c v delta rho beta kappa stop lambda;
2
3  l = 15;
4  L = 30;
5  c = zeros(L,1);
6  v = 10;
7  delta = 0.03;
8  rho = 0.85;
9  beta = 1/1.05;
10 kappa = 10;
11
12 stop = 1e-3;
13 lambda = 1;
14
15 %I also tried lambda = 0.5 as proposed in dampening procedure. The
    number of iterations to converge was twice
16 %larger
17
18 c(1:l)=kappa*[1:1:l]'.^(log(rho)/log(2)); %Constructing cost
    function
19 c(l+1:end) = kappa*(1^(log(rho)/log(2)));
20 tr_pr=zeros(L+1,L); %Constructing the state transition conditional

```

```

        on sales
21  for i=1:L+1
22      if i==1
23          tr_pr(i,1)=1;
24      elseif i==L+1
25          tr_pr(i,L)=1;
26      else
27          tr_pr(i,i-1)=1-(1-delta)^(i);
28          tr_pr(i,i)=(1-delta)^(i);
29      end
30  end

```

## Matlab Code

```
1 function [d0,d1,d2] = Ds(p,p_1,v)
2 d0 = 1./(1+exp(v-p)+exp(v-p_1')) ;
3 d1 = exp(v-p)./(1+exp(v-p)+exp(v-p_1')) ;
4 d2 = exp(v-p_1')./(1+exp(v-p)+exp(v-p_1')) ;
5 end
```

# Matlab Code

```
1  function [w0,w1,w2]=Ws(V,tr_pr)
2
3  s = size(tr_pr,1);
4  t_p = tr_pr;
5  tr_pr_0 = t_p(1:s-1,:);
6  tr_pr_1 = t_p(2:s,:);
7
8  w0 = tr_pr_0*V*tr_pr_0';
9  w1 = tr_pr_1*V*tr_pr_0';
10 w2=tr_pr_0*V*tr_pr_1';
11
12 end
```

## Matlab Code

```
1 function f = FOC(V,p,p-1,L,c,beta,v,tr_pr)
2 c1 = repmat(c',L,1);
3 [d0, d1, d2] = Ds(p,p-1,v);
4 [w0, w1,w2] = Ws(V,tr_pr);
5 FOC=1-(1-d1).*(p-c1)-beta*w1+beta*(d0.*w0+d1.*w1+d2.*w2);
6 f = FOC;
7 end
```



# Matlab Code

```
1 function [v,p,it] = VFI(V,P,stop,L,c,lambda,beta,v,tr_pr)
2
3 v_old = V;
4 p_old = P;
5 initial = P;
6 opt = optimset('Disp','None');
7 v_new = zeros(size(V));
8 p_new = zeros(size(P));
9 tol = 1;
10 i = 1;
11
12 while and(tol > stop, i < 1000)
13     f=@(p)FOC(v_old,p,p_old,L,c,beta,v,tr_pr);
14     p_new = fsolve(f,initial,opt);
15     [d0, d1, d2] = Ds(p_new,p_old,v);
16     [w0, w1, w2] = Ws(v_old,tr_pr);
17
18     v_new = d1.*(p_new-repmat(c',L,1))+beta*(d0.*w0+d1.*w1+d2.*w2);
19
20     tol = max(max(max(abs((v_new-v_old)./(1+v_new))))),max(max(abs((
        p_new-p_old)./(1+p_new))))));
21
22     v_old=lambda*v_new+(1-lambda)*v_old; p_old=lambda*p_new+(1-
        lambda)*p_old;
23
24     fprintf('Iteration:%d\n',i);
25     fprintf('Tolerance:%f\n',tol);
26     i=i+1;
27 end
28 v = v_old;
29 p = p_old;
30 it = i;
31 end
```

# Matlab Code

```
1 %% Question 1
2
3 tic;
4 Parameters;
5 V_0 = ( repmat(v,L,L) + repmat(c',L,1) )./(2*beta);
6 P_0 = 1.5*repmat(c',L,1);
7
8 [value, p, it] = VFI(V_0, P_0, stop, L, c, lambda, beta, v, tr_pr);
9
10 figure(1);
11 mesh(value);
12 title('Value Func');
13 xlabel('Firm1 Knowledge');
14 ylabel('Firm2 Knowledge');
15 zlabel('Value Func');
16 figure(2);
17 mesh(p);
18 title('Price');
19 xlabel('Firm1 Knowledge');
20 ylabel('Firm2 Knowledge');
21 zlabel('Value Func');
22 fprintf('Time:%f sec',toc);
23
24 %% Questions 2&3
25
26 [D_0,D_1,D_2] = Ds(p,p,v);
27 T = zeros(L^2,L^2);
28
29 for k=1:30
30     if k==1
31         for i=1:30
32             if i==1
33                 T((k-1)*30+i,(k-1)*30+i)=D_0(k,i) ...
34                     +D_1(k,i)*(1-(1-delta)^(k+1))+D_2(k,i)*(1-(1-
```

```

delta)^(i+1));
35 T((k-1)*30+i,(k-1)*30+i+1)=D_2(k,i)*(1-delta)^(i+1);
36 T((k-1)*30+i,(k)*30+i)=D_1(k,i)*((1-delta)^(k+1));
37 elseif i<30
38 T((k-1)*30+i,(k-1)*30+i)=D_0(k,i)*((1-delta)^i)...
39 +D_1(k,i)*(1-(1-delta)^(k+1))*((1-delta)^i)+D_2(
k,i)*(1-(1-delta)^(i+1));
40 T((k-1)*30+i,(k-1)*30+i+1)=D_2(k,i)*(1-delta)^(i+1);
41 T((k-1)*30+i,(k)*30+i)=D_1(k,i)*((1-delta)^(k+1))
*(1-delta)^i;
42 T((k-1)*30+i,(k-1)*30+i-1)=D_0(k,i)*(1-(1-delta)^i)+
D_1(k,i)*(1-(1-delta)^(k+1))*(1-(1-delta)^i);
43 T((k-1)*30+i,(k)*30+i-1)=D_1(k,i)*((1-delta)^(k+1))
*(1-(1-delta)^i);
44 elseif i==30
45 T((k-1)*30+i,(k-1)*30+i)=D_0(k,i)*((1-delta)^i)...
46 +D_1(k,i)*(1-(1-delta)^(k+1))*((1-delta)^i)+D_2(
k,i);
47 T((k-1)*30+i,(k-1)*30+i-1)=D_0(k,i)*(1-(1-delta)^i)+
D_1(k,i)*(1-(1-delta)^(k+1))*(1-(1-delta)^i);
48 T((k-1)*30+i,(k)*30+i)=D_1(k,i)*((1-delta)^(k+1))
*(1-delta)^i;
49 T((k-1)*30+i,(k)*30+i-1)=D_1(k,i)*((1-delta)^(k+1))
*(1-(1-delta)^i);
50 end
51 end
52 elseif k<30
53 for i=1:30
54 if i==1
55 T((k-1)*30+i,(k-1)*30+i)=D_0(k,i)*((1-delta)^k)...
56 +D_1(k,i)*(1-(1-delta)^(k+1))+D_2(k,i)*((1-delta)^k)
*(1-(1-delta)^(i+1));
57 T((k-1)*30+i,(k-1)*30+i+1)=D_2(k,i)*((1-delta)^k)
*(1-delta)^(i+1);
58 T((k-1)*30+i,(k)*30+i)=D_1(k,i)*((1-delta)^(k+1));

```

```

59      T((k-1)*30+i, (k-2)*30+i)=D_0(k, i)*(1-(1-delta)^k)+
        D_2(k, i)*(1-(1-delta)^k)*(1-(1-delta)^(i+1));
60      T((k-1)*30+i, (k-2)*30+i+1)=D_2(k, i)*(1-(1-delta)^k)
        *((1-delta)^(i+1));
61
62      elseif i<30
63          T((k-1)*30+i, (k-1)*30+i)=D_0(k, i)*((1-delta)^k)*((1-
        delta)^i) ...
64          +D_1(k, i)*(1-(1-delta)^(k+1))*((1-delta)^i)+D_2(
        k, i)*((1-delta)^k)*(1-(1-delta)^(i+1));
65      T((k-1)*30+i, (k-1)*30+i+1)=D_2(k, i)*((1-delta)^k)
        *(1-delta)^(i+1);
66      T((k-1)*30+i, (k)*30+i)=D_1(k, i)*((1-delta)^(k+1))
        *(1-delta)^i;
67      T((k-1)*30+i, (k)*30+i-1)=D_1(k, i)*((1-delta)^(k+1))
        *(1-(1-delta)^i);
68      T((k-1)*30+i, (k-1)*30+i-1)=D_0(k, i)*((1-delta)^k)
        *(1-(1-delta)^i)+D_1(k, i)*(1-(1-delta)^(k+1))
        *(1-(1-delta)^i);
69      T((k-1)*30+i, (k-2)*30+i)=D_0(k, i)*(1-(1-delta)^k)
        *((1-delta)^i)+D_2(k, i)*(1-(1-delta)^k)*(1-(1-
        delta)^(i+1));
70      T((k-1)*30+i, (k-2)*30+i-1)=D_0(k, i)*(1-(1-delta)^k)
        *(1-(1-delta)^i);
71      T((k-1)*30+i, (k-2)*30+i+1)=D_2(k, i)*(1-(1-delta)^k)
        *((1-delta)^(i+1));
72
73      elseif i==30
74          T((k-1)*30+i, (k-1)*30+i)=D_0(k, i)*((1-delta)^k)*((1-
        delta)^i) ...
75          +D_1(k, i)*(1-(1-delta)^(k+1))*((1-delta)^i)+D_2(
        k, i)*((1-delta)^k);
76      T((k-1)*30+i, (k)*30+i)=D_1(k, i)*((1-delta)^(k+1))
        *(1-delta)^i;
77      T((k-1)*30+i, (k)*30+i-1)=D_1(k, i)*((1-delta)^(k+1))

```

```

78      *(1-(1-delta)^i);
      T((k-1)*30+i,(k-1)*30+i-1)=D_0(k,i)*((1-delta)^k)
      *(1-(1-delta)^i)+D_1(k,i)*(1-(1-delta)^(k+1))
      *(1-(1-delta)^i);
79      T((k-1)*30+i,(k-2)*30+i)=D_0(k,i)*(1-(1-delta)^k)
      *((1-delta)^i)+D_2(k,i)*(1-(1-delta)^k);
80      T((k-1)*30+i,(k-2)*30+i-1)=D_0(k,i)*(1-(1-delta)^k)
      *(1-(1-delta)^i);
81      end
82      end
83      elseif k==30
84          for i=1:30
85              if i==1
86                  T((k-1)*30+i,(k-1)*30+i)=D_0(k,i)*((1-delta)^k)...
87                  +D_1(k,i)+D_2(k,i)*((1-delta)^k)*(1-(1-delta)^(i+1))
                        ;
88                  T((k-1)*30+i,(k-1)*30+i+1)=D_2(k,i)*((1-delta)^k)
                        *(1-delta)^(i+1);
89                  T((k-1)*30+i,(k-2)*30+i)=D_0(k,i)*(1-(1-delta)^k)+
                        D_2(k,i)*(1-(1-delta)^k)*(1-(1-delta)^(i+1));
90                  T((k-1)*30+i,(k-2)*30+i+1)=D_2(k,i)*(1-(1-delta)^k)
                        *((1-delta)^(i+1));
91
92                  elseif i<30
93                      T((k-1)*30+i,(k-1)*30+i)=D_0(k,i)*((1-delta)^k)*((1-
                        delta)^i)...
94                      +D_1(k,i)*((1-delta)^i)+D_2(k,i)*((1-delta)^k)
                        *(1-(1-delta)^(i+1));
95                      T((k-1)*30+i,(k-1)*30+i+1)=D_2(k,i)*((1-delta)^k)
                        *(1-delta)^(i+1);
96                      T((k-1)*30+i,(k-1)*30+i-1)=D_0(k,i)*((1-delta)^k)
                        *(1-(1-delta)^i)+D_1(k,i)*(1-(1-delta)^i);
97                      T((k-1)*30+i,(k-2)*30+i)=D_0(k,i)*(1-(1-delta)^k)
                        *((1-delta)^i)+D_2(k,i)*(1-(1-delta)^k)*(1-(1-
                        delta)^(i+1));

```

```

98         T((k-1)*30+i,(k-2)*30+i-1)=D_0(k,i)*(1-(1-delta)^k)
           *(1-(1-delta)^i);
99         T((k-1)*30+i,(k-2)*30+i+1)=D_2(k,i)*(1-(1-delta)^k)
           *((1-delta)^(i+1));
100     elseif i==30
101         T((k-1)*30+i,(k-1)*30+i)=D_0(k,i)*((1-delta)^k)*((1-
           delta)^i)...
           +D_1(k,i)*((1-delta)^i)+D_2(k,i)*((1-delta)^k);
102         T((k-1)*30+i,(k-1)*30+i-1)=D_0(k,i)*((1-delta)^k)
           *(1-(1-delta)^i)+D_1(k,i)*(1-(1-delta)^i);
103         T((k-1)*30+i,(k-2)*30+i)=D_0(k,i)*(1-(1-delta)^k)
           *((1-delta)^i)+D_2(k,i)*(1-(1-delta)^k);
104         T((k-1)*30+i,(k-2)*30+i-1)=D_0(k,i)*(1-(1-delta)^k)
           *(1-(1-delta)^i);
105     end
106 end
107 end
108 end
109 end
110
111 initial = [1,zeros(1,899)];
112
113 figure(3);%10 periods
114 A1=reshape((initial*(T^10))',30,30);
115 mesh(A1);
116 xlabel('Firm1 Knowledge'); ylabel('Firm2 Knowledge'); zlabel('
           Density');
117 title('Distribution of states after 10 periods');
118
119 figure(4);%20 periods
120 A2=reshape((initial*(T^20))',30,30);
121 mesh(A2);
122 xlabel('Firm1 Knowledge'); ylabel('Firm2 Knowledge'); zlabel('
           Density');
123 title('Distribution of states after 20 periods');
124

```

```

125
126 figure(5);%30 periods
127 A3=reshape((initial*(T^30))',30,30);
128 mesh(A3);
129 xlabel('Firm1 Knowledge'); ylabel('Firm2 Knowledge'); zlabel('
    Density');
130 title('Distribution of states after 30 periods');
131
132 figure(6);%Steady State
133 A4=reshape((initial*(T^900))',30,30);
134 mesh(A4);
135 xlabel('Firm1 Knowledge'); ylabel('Firm2 Knowledge'); zlabel('
    Density');
136 title('Stationary Distribution of states');

```