Homework 6

Kensuke Suzuki

January 27, 2019

Problem 1: Dynamic optimization problem

Firm's current period payoff is

$$\pi(x_t, p_t) = p_t \cdot x_t - 0.2 \cdot x_t^{1.5} \tag{1}$$

We want to make the stock of lumber as a state variable, so that $x = k - k'$. Then equation (1) can be written as

$$\pi(k_t, k_{t+1}, p_t) = p \cdot (k_t - k_{t+1}) - 0.2 \cdot (k_t - k_{t+1})^{1.5} \tag{2}$$

Then we can formulate the firm's problem such that it chooses k' (next period stock) given the state variables (k, p). The Bellman's equation is

$$V(k, p) = \max_{k'} p \cdot (k - k') - 0.2 \cdot (k - k')^{1.5} + \delta \mathbb{E}_{p'} \left[V(k', p') | p \right] \tag{3}$$

given initial stock of lumber (between 0 and 100) and transition probability of p which is defined by the AR(1) process ($p' = p_0 + \rho \cdot p + u$). In the Bellman's equation presented in equation (2), expectation is taken over next period price p' given today's price p.

Problem 2: Tauchen

I use `tauchen.m` to generate grid that approximate process for p_t for given AR(1) parameters. Variable `grid` stores the grid points for p_t and `prob` stores the transition probabilities.

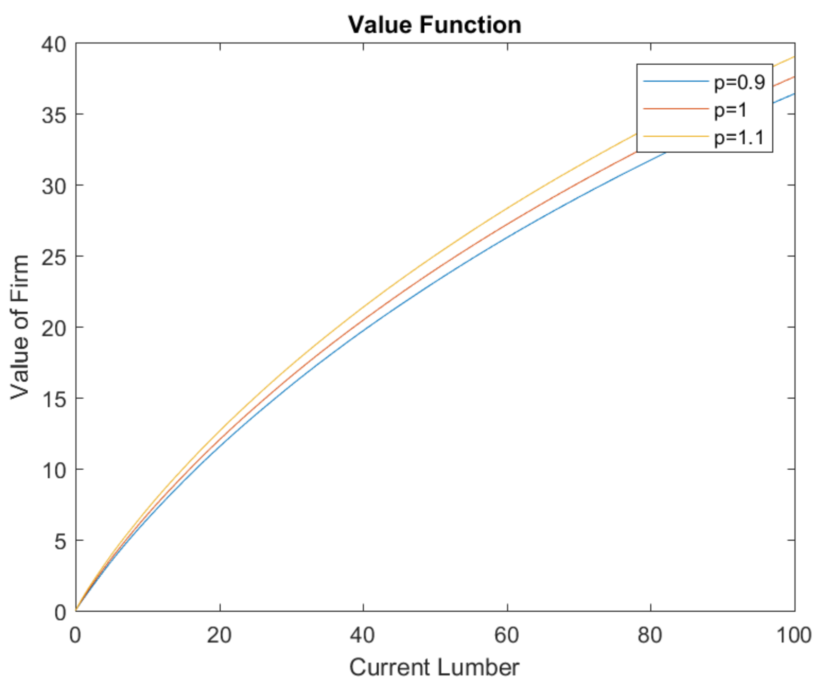
Problem 3: Value function iteration (VIF)

The basic structure of the VIF algorithm is analogous to the one provided in the lecture. Note that:

- `harvest` stores the harvest for given current stock (in column) and next period stock (in raw)
- I penalize the negative revenue by replacing with $-1e6.0$
- In order to avoid imaginary number, before calculating the current payoff (profit), I replace the negative harvest with 0.

I plot the value of the firm depending on its initial stock (x-axis) and the current price of lumber.

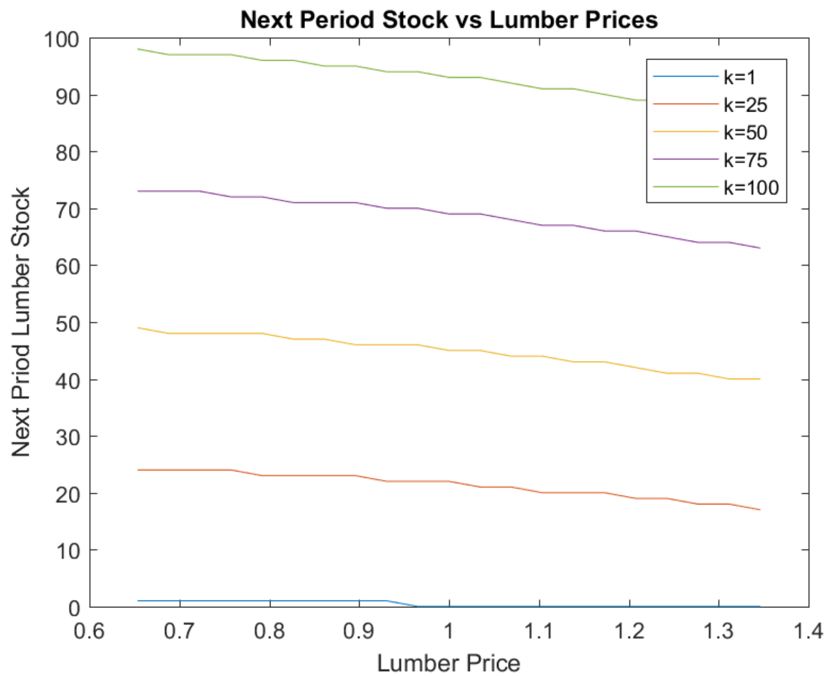
Figure 1: Problem 3



Problem 4: Next period stock as a function of price

I plot next period optimal stock as a function of today's price for different amount of lumber left in stock.

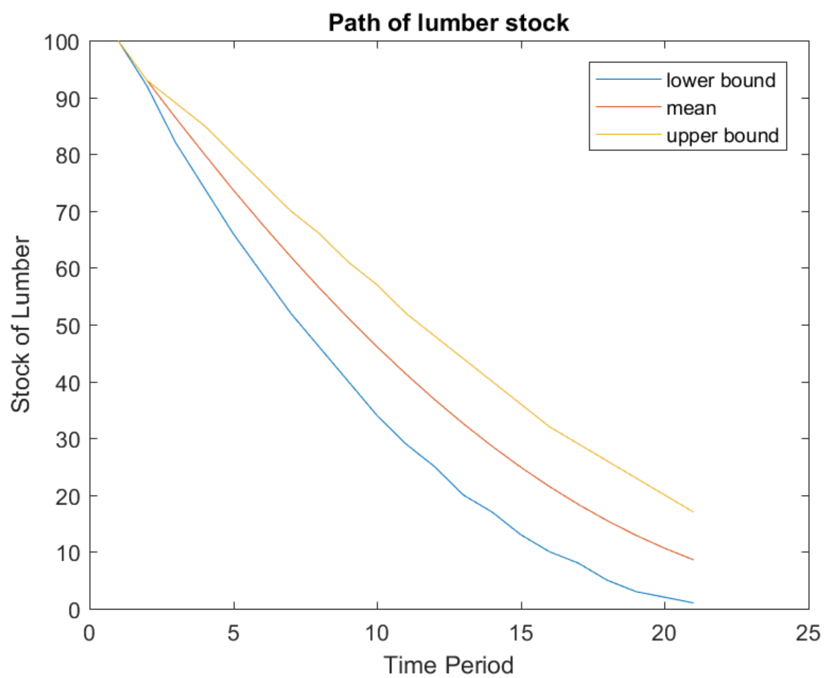
Figure 2: Problem 4



Problem 5: Expected path of lumber stock

Using the code provided in the class, I first construct the transition probability matrix of the state (k,p) by combining the transition matrix of price and the decision rule matrix. Given the initial state ($k = 100, p = 1$), we can construct the initial probability distribution matrix, which has a mass (1) on $k = 100$ and $p = 1$. Then, using the transition matrix of the state, we can generate the next period probability distribution over state (k,p). By integrating (summing) over all possible price values, we can obtain the expected lumber stock. We can also obtain the lower and upper bound of confidence interval by obtaining 5% and 95%tile of the marginal distribution of k in each period. We repeat this for 20 periods. In the figure below, $t = 1$ is the starting point, so we compute the expected path and the confidence interval from $t = 2$ to $t = 21$.

Figure 3: Problem 5



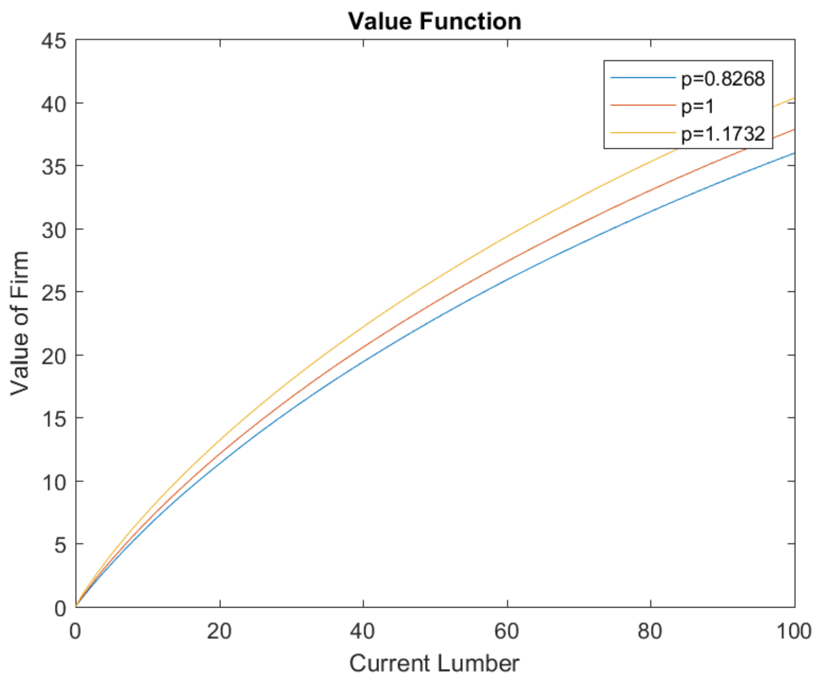
Problem 6: Results with coarse grid

Problem 6-2: Tauchen

I now use 5 grid points and generate the grid and transition probability for p_t.

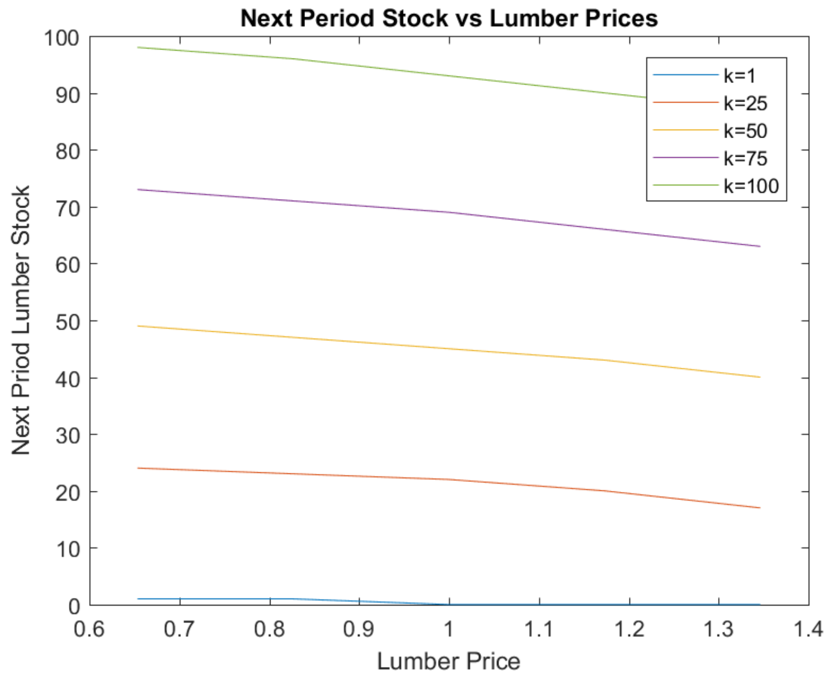
Problem 6-3: VIF

Figure 4: Problem 6-3



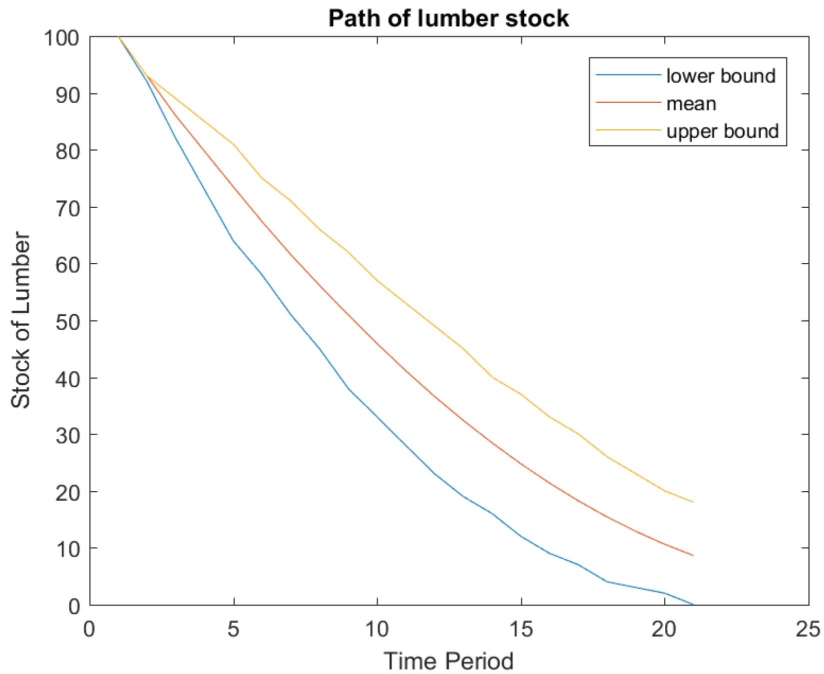
Problem 6-4: Next period stock as a function of price

Figure 5: Problem 6-4



Problem 6-5: Expected stock

Figure 6: Problem 6-5



Matlab Main Code

```
% Spring 2019
% ECON512 Empirical Method
% Homework 6 --- Dynamic Programing
% Kensuke Suzuki
% kxs974@psu.edu

clear all;

%% Basic setup
delta = 0.95;
p0 = 0.5;
rho = 0.5;
sigma = 0.1;

N = 101 ;% number of grid
k = linspace(0,100,N);

%% Problem 1
disp('Refer to the PDF')

%% Problem 2 --- Tauchen

Z = 21; % grid points generated

[prob,grid]=tauchen(Z,p0,rho,sigma);
disp(['The dimensions of prob are ' num2str(size(prob)) ])
disp(['The dimensions of grid are ' num2str(size(grid)) ])


%% Problem 3 --- Value function iteration

vinitial=zeros(N,Z);
vrevised=zeros(N,Z);
decision=zeros(N,Z);

invest=kron(ones(1,Z),k');
disp(['The dimensions of harvest  are ' num2str(size(invest)) ])
ONE=ones(N,1);

diff = 1;
while diff > 1e-6

    Ev = vinitial*prob' ;% (NxZ * ZxZ = NxZ);

    Ev;

    for i = 1:N
        harvest = ( kron(ones(N,Z),k(:,i)') - invest) ; % harvest
```

```
        revenue = (ones(N,1)*grid) .* harvest; % revenue
        revenue(revenue < 0) = -1e6; % punish negative revenue
        harvest = max(harvest, 0); % replace negative harvest to avoide imaginary num
        pi = revenue - 0.2 * (harvest.^(1.5)); % current payoff
        [vrevised(i,:),decision(i,:)] = max( pi + delta*Ev ); % get value and policy function
        decision;
    end
    diff = norm(vrevised - vinitial) / norm(vrevised) % check deviance
    vinitial = vrevised;

end

plot(k,vrevised(:,8)',k,vrevised(:,11)',k,vrevised(:,14)')
xlabel('Current Lumber')
ylabel('Value of Firm')
title('Value Function')
legend('p=0.9','p=1', 'p=1.1')
saveas(gcf,'prob3.png')

%
% %%
%
% plot(grid, decision(2,:), grid, decision(26,:),grid, decision(51,:),grid, decision(76,:),grid, decision(101,:))
% title('Next Period Stock as a Function of Price')
% xlabel('Price')
% ylabel('Next period stock of lumber')


%% Problem 4

drule=zeros(N,Z);

for i=1:Z
    drule(:,i)=k(decision(:,i))'; % get decision rule
end

plot(grid, drule(2,:), grid, drule(26,:),grid, drule(51,:),grid, drule(76,:),grid, drule(101,:))
title('Next Period Stock vs Lumber Prices')
xlabel('Lumber Price')
ylabel('Next Priod Lumber Stock')
legend('k=1','k=25', 'k=50', 'k=75', 'k=100')
saveas(gcf,'prob4.png')


%% Problem 5 Expected path

```

```
T = 21; % time period
Time = 1:T;

% Construct transition matrix
P=zeros(Z*N,Z*N);

for i=1:Z
    for j=1:Z
        P((i-1)*N+1:i*N,(j-1)*N+1:j*N)=prob(i,j)*(kron(ones(1,N),drule(:,i)
            )==kron(ones(N,1),k));
    end
end

% transition probability: p=1 and k = 100 (initial)
pint = zeros(1,N*Z); % 11 is index of p=1
pint(1,11*101)=1; % since period 1 state is (1,100), mass is 1 on

% Simulate the model
result = zeros(3,T);
result(:,1) =[100;100;100];
for t=2:21
    pnext = pint*P; % next period probability (1 x Z*N)

    probk=zeros(N,Z); % generate probability distribution for stock
    for i=1:Z
        probk(:,i) = pnext((i-1)*N+1:i*N)';
    end
    probk = sum(probk'); % summing over for all price

    cdf = cumsum(probk); % get cdf

    lowerind = max(find(cdf<=0.05)); % get 5-percentile index
    upperind = min(find(cdf>=0.95)); % get 95-percentile index

    mean = probk*k';

    LB = k(lowerind); % lower bound
    UB = k(upperind);

    result(:,t) = [LB mean UB]; % store result

    pint = pnext; % update today's distirbution
end

plot(1:21, result(1,:), Time, result(2,:), Time, result(3,:))
title('Path of lumber stock')
xlabel('Time Period')
ylabel('Stock of Lumber')
legend( 'lower bound', 'mean', 'upper bound')
```

```
saveas(gcf,'prob5_2.png')

%% Problem 6 Tauchen with coarse grid

clear prob grid vinitial vrevused decision mean

Z = 5; % grid points generated

[prob,grid]=tauchen(Z,p0,rho,sigma);
disp(['The dimensions of prob are ' num2str(size(prob)) ])
disp(['The dimensions of grid are ' num2str(size(grid)) ])

vinitial=zeros(N,Z);
vrevised=zeros(N,Z);
decision=zeros(N,Z);

invest=kron(ones(1,Z),k');
disp(['The dimensions of harvest  are ' num2str(size(invest)) ])
ONE=ones(N,1);

vinitial=zeros(N,Z);
vrevised=zeros(N,Z);
decision=zeros(N,Z);

invest=kron(ones(1,Z),k');
disp(['The dimensions of harvest  are ' num2str(size(invest)) ])
ONE=ones(N,1);

diff = 1;
while diff > 1e-6

    Ev = vinitial*prob' ;% (NxZ * ZxZ = NxZ);

    Ev;

    for i = 1:N
        harvest = ( kron(ones(N,Z),k(:,i)') - invest) ;
        revenue = (ones(N,1)*grid) .* harvest;
        revenue(revenue < 0) = -1e6;
        harvest = max(harvest, 0);
        pi = revenue - 0.2 * (harvest.^(1.5));

        %pi = (ones(N,1)*grid) .* harvest - 0.2 * (harvest_z.^(1.5));
        check = pi + delta*Ev;
        [vrevised(i,:),decision(i,:)] = max( pi + delta*Ev );
        decision;
    end
    diff = norm(vrevised - vinitial) / norm(vrevised)
    vinitial = vrevised;

```

```
end

plot(k,vrevised(:,2)',k,vrevised(:,3)',k,vrevised(:,4)')
xlabel('Current Lumber')
ylabel('Value of Firm')
title('Value Function')
legend('p=0.8268','p=1', 'p=1.1732')
saveas(gcf,'prob6_1.png')


drule=zeros(N,Z);

for i=1:Z
    drule(:,i)=k(decision(:,i))';
end


plot(grid, drule(2,:), grid, drule(26,:),grid, drule(51,:),grid, drule
    (76,:),grid, drule(101,:))
title('Next Period Stock vs Lumber Prices')
xlabel('Lumber Price')
ylabel('Next Priod Lumber Stock')
legend('k=1','k=25', 'k=50', 'k=75', 'k=100')
saveas(gcf,'prob6_2.png')

T = 21; % time period
Time = 1:T;

% Construct transition matrix
P=zeros(Z*N,Z*N);

for i=1:Z
    for j=1:Z
        P((i-1)*N+1:i*N,(j-1)*N+1:j*N)=prob(i,j)*(kron(ones(1,N),drule(:,i)
            )==kron(ones(N,1),k));
    end
end

% transition probability: p=1 and k = 100 (initial)
pint = zeros(1,N*Z); % 3 is index of p=1
pint(1,3*101)=1; % since period 1 state is (1,100), mass is 1 on

% Simulate the model
result = zeros(3,T);
result(:,1) =[100;100;100];
for t=2:21
    pnext = pint*P; % next period probability (1 x Z*N)

    probk=zeros(N,Z); % generate probability distribution for stock
    for i=1:Z
```

```
            probk(:,i) = pnext((i-1)*N+1:i*N)';
        end
        probk = sum(probk'); % summing over for all price

        cdf = cumsum(probk); % get cdf

        lowerind = max(find(cdf<=0.05)); % get 5-percentile index
        upperind = min(find(cdf>=0.95)); % get 95-percentile index

        mean = probk*k';

        LB = k(lowerind); % lower bound
        UB = k(upperind);

        result(:,t) = [LB mean UB]; % store result

        pint = pnext; % update today's distirbution
end

plot(1:21, result(1,:), Time, result(2,:), Time, result(3,:))
title('Path of lumber stock')
xlabel('Time Period')
ylabel('Stock of Lumber')
legend( 'lower bound', 'mean', 'upper bound')
saveas(gcf,'prob6_3_2.png')

%%

% T = 21; % time period
% Time = 1:T;
% sim = 100; % number of simulation
%
% path_p = zeros(sim,T);
% path_p(:,1) = 11;
%
% % generate the price path (20 period)  1000 times
% for s = 1:sim
%     for i = 2:21
%         rng(s * 2019 + i );
%         pmf = prob(path_p(s,i-1),:);
%         cdf = cumsum(pmf);
%         r = rand;
%         path_p(s,i) = find(cdf>r,1);
%     end
% end
%
% % generate the lumber stock path
% path_k = zeros(sim,T);
% path_k(:,1) = N; % this is index
% path_stock = zeros(sim,T);
```

```
% path_stock(:,1) = 100;
%
% for s = 1:sim
%     for i = 2:21
%         path_k(s,i) = decision(path_k(s,i-1), path_p(s,i-1));
%         path_stock(s,i) = k(path_k(s,i));
%     end
% end
%
% mean = mean(path_stock);
% stderr = std(path_stock);
% ts = tinv([0.05  0.95],length(path_stock-1))';
%
% CI = ones(2,1)*mean + ts*stderr/sqrt(length(path_stock)) ;
%
% plot(Time, mean, Time, CI(1,:), Time, CI(2,:))
% title('Path of lumber stock')
% xlabel('Time Period')
% ylabel('Stock of Lumber')
% legend('mean', 'lower bound', 'upper bound')
% saveas(gcf,'prob5.png')
```