

## Homework 6

---

Kensuke Suzuki

January 27, 2019

### Problem 1: Dynamic optimization problem

Firm's current period payoff is

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

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

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

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

$$V(k, p) = \max_{k' \in [0, k]} p \cdot (k - k') - 0.2 \cdot (k - k')^{1.5} + \delta \mathbb{E}_{p'} [V(k', p') | p] \quad (3)$$

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.

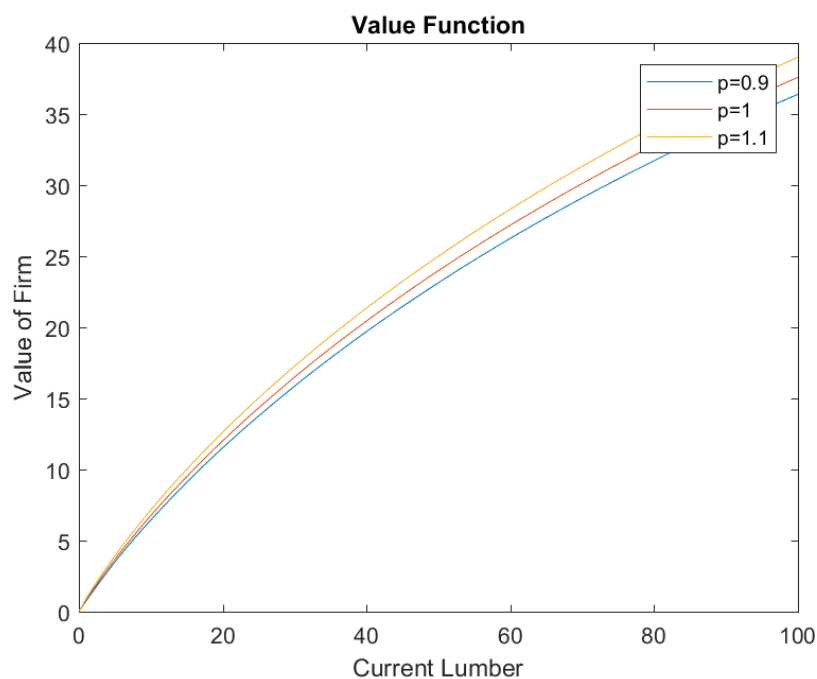
### Problem 3: Value function iteration (VIF)

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 row)
- 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.

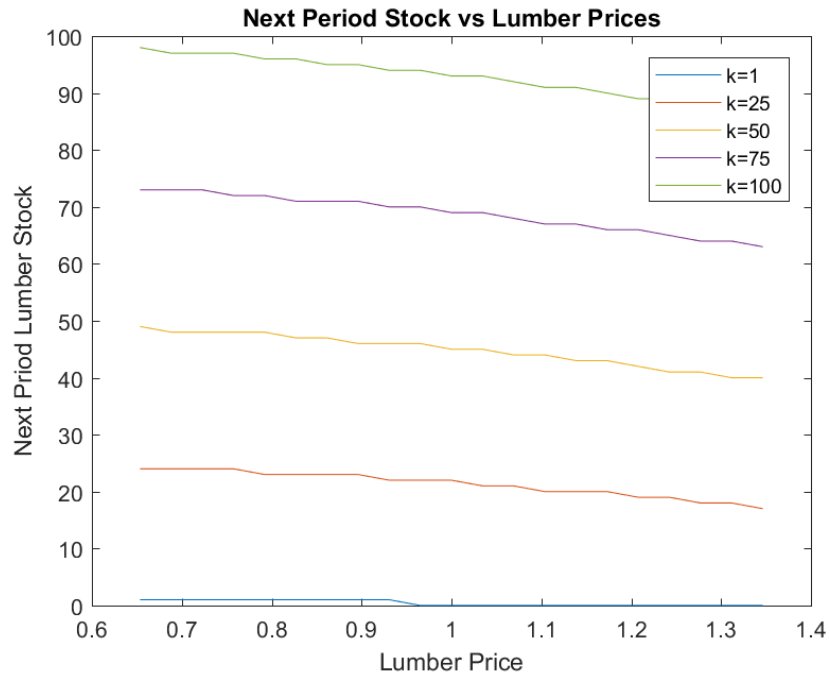
Figure 1: Problem 3



### 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.

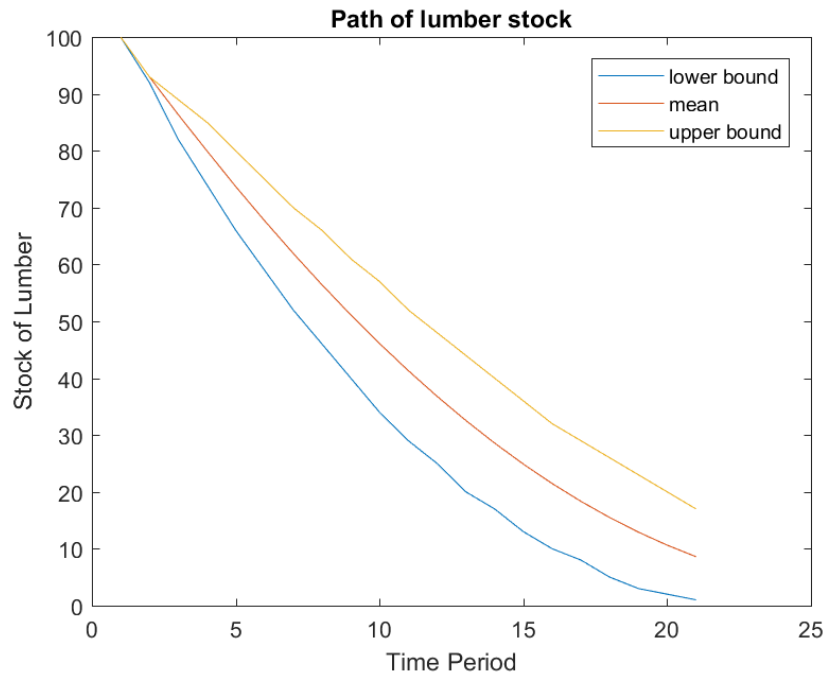
Figure 2: Problem 4



### 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



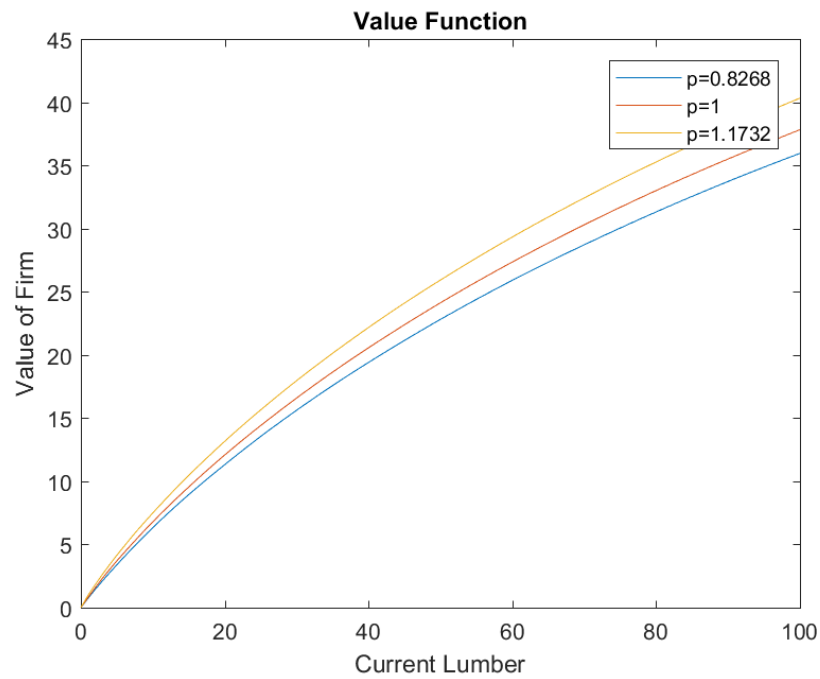
## Problem 6: Results with coarse grid

### Problem 6-2: Tauchen

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

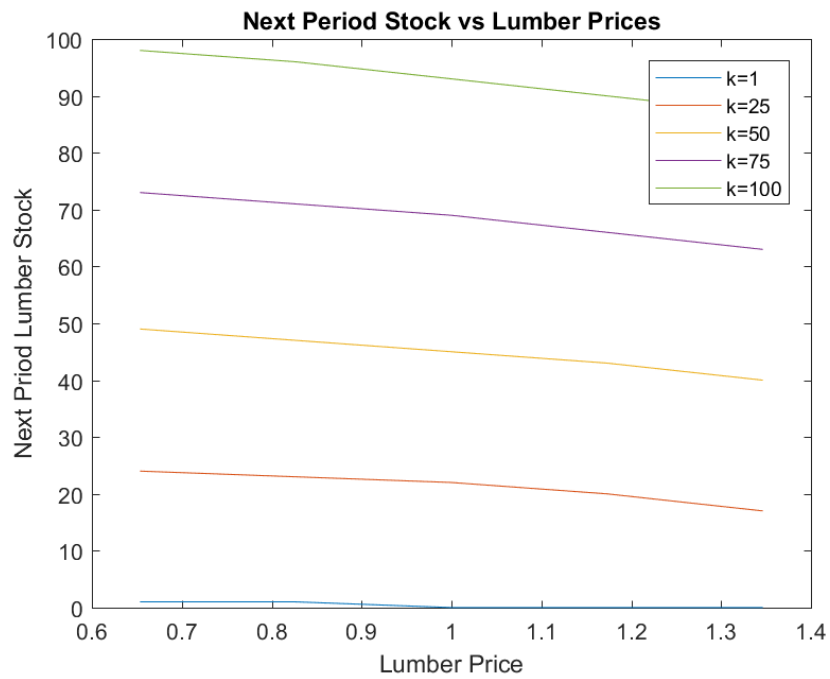
### Problem 6-3: VIF

Figure 4: Problem 6-3



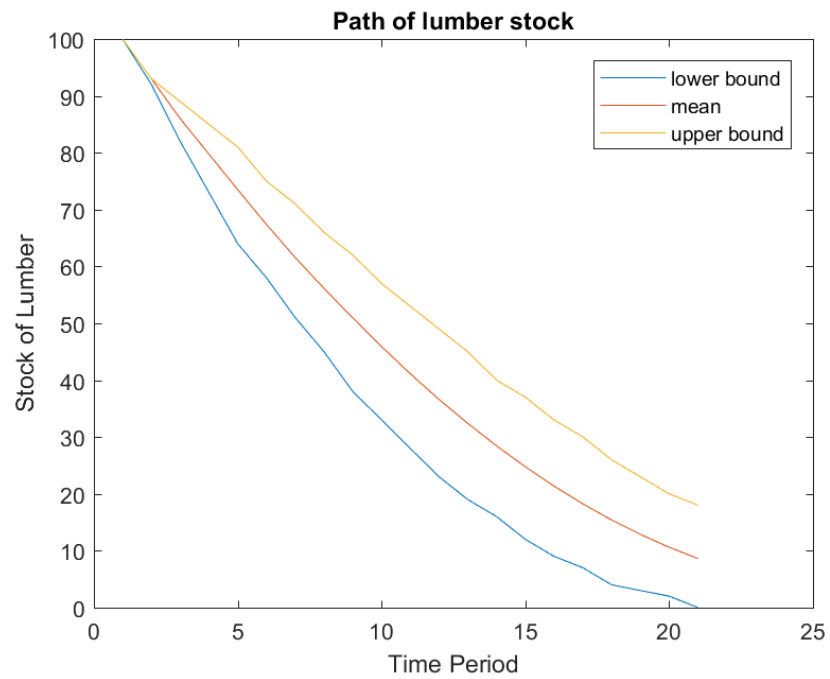
**Problem 6-4: Next period stock as a function of price**

Figure 5: Problem 6-4



**Problem 6-5: Expected stock**

Figure 6: Problem 6-5



## Matlab Main Code

```
1 % Spring 2019
2 % ECON512 Empirical Method
3 % Homework 6 — Dynamic Programing
4 % Kensuke Suzuki
5 % kxs974@psu.edu
6
7 clear all;
8
9 clear all
10 delete HW6log.txt
11 diary('HW6log.txt')
12 diary on
13
14 %% Basic setup
15 delta = 0.95;
16 p0 = 0.5;
17 rho = 0.5;
18 sigma = 0.1;
19
20 N = 101 ;% number of grid
21 k = linspace(0,100,N);
22
23 %% Problem 1
24 disp('Refer to the PDF')
25
26 %% Problem 2 — Tauchen
27
28 Z = 21; % grid points generated
29
30 [prob, grid]=tauchen(Z,p0,rho,sigma);
31 disp(['The dimensions of prob are ' num2str(size(prob)) ])
32 disp(['The dimensions of grid are ' num2str(size(grid)) ])
33
34
35 %% Problem 3 — Value function iteration
36
37 vinitia=zeros(N,Z);
38 vrevise=zeros(N,Z);
39 decision=zeros(N,Z);
40
41 invest=kron(ones(1,Z),k');
42 disp(['The dimensions of harvest are ' num2str(size(invest)) ])
43 ONE=ones(N,1);
44
45 diff = 1;
46 while diff > 1e-6
47
48     Ev = vinitia*prob' ;% (NxZ * ZxZ = NxZ);
```



```

49
50     Ev;
51
52     for i = 1:N
53         harvest = ( kron(ones(N,Z),k(:,i)') - invest) ; % harvest
54         revenue = (ones(N,1)*grid) .* harvest; % revenue
55         revenue(revenue < 0) = -1e6; % punish negative revenue
56         harvest = max(harvest, 0); % replace negative harvest to avoid
            imaginary num
57         pi = revenue - 0.2 * (harvest.^(1.5)); % current payoff
58         [vrevise(i,:),decision(i,:)] = max( pi + delta*Ev ); % get value
            and policy function
59         decision;
60     end
61     diff = norm(vrevise - vinitial) / norm(vrevise) % check deviance
62     vinitial = vrevise;
63
64 end
65
66 plot(k,vrevise(:,8)',k,vrevise(:,11)',k,vrevise(:,14)')
67 xlabel('Current Lumber')
68 ylabel('Value of Firm')
69 title('Value Function')
70 legend('p=0.9','p=1','p=1.1')
71 saveas(gcf,'prob3.png')
72
73 %
74 % %%
75 %
76 % plot(grid, decision(2,:), grid, decision(26,:),grid, decision(51,:),grid,
            decision(76,:),grid, decision(101,:))
77 % title('Next Period Stock as a Function of Price')
78 % xlabel('Price')
79 % ylabel('Next period stock of lumber')
80
81
82 %% Problem 4
83
84 drule=zeros(N,Z);
85
86 for i=1:Z
87     drule(:,i)=k(decision(:,i))'; % get decision rule
88 end
89
90 plot(grid, drule(2,:), grid, drule(26,:),grid, drule(51,:),grid, drule
            (76,:),grid, drule(101,:))
91 title('Next Period Stock vs Lumber Prices')
92 xlabel('Lumber Price')
93 ylabel('Next Period Lumber Stock')
94 legend('k=1','k=25','k=50','k=75','k=100')

```

```

95 saveas(gcf, 'prob4.png')
96
97
98 %% Problem 5 Expected path
99
100
101 T = 21; % time period
102 Time = 1:T;
103
104 % Construct transition matrix
105 P=zeros(Z*N,Z*N);
106
107 for i=1:Z
108     for j=1:Z
109         P((i-1)*N+1:i*N,(j-1)*N+1:j*N)=prob(i,j)*(kron(ones(1,N),drule(:,i))
110             )==kron(ones(N,1),k));
111     end
112 end
113
114 % transition probability: p=1 and k = 100 (initial)
115 pint = zeros(1,N*Z); % 11 is index of p=1
116 pint(1,11*101)=1; % since period 1 state is (1,100), mass is 1 on
117
118 % Simulate the model
119 result = zeros(3,T);
120 result(:,1) =[100;100;100];
121 for t=2:21
122     pnext = pint*P; % next period probability (1 x Z*N)
123
124     probk=zeros(N,Z); % generate probability distribution for stock
125     for i=1:Z
126         probk(:,i) = pnext((i-1)*N+1:i*N)';
127     end
128     probk = sum(probk'); % summing over for all price
129
130     cdf = cumsum(probk); % get cdf
131
132     lowerind = max(find(cdf<=0.05)); % get 5-percentile index
133     upperind = min(find(cdf>=0.95)); % get 95-percentile index
134
135     mean = probk*k';
136
137     LB = k(lowerind); % lower bound
138     UB = k(upperind);
139
140     result(:,t) = [LB mean UB]; % store result
141
142     pint = pnext; % update today's distribution
143 end

```

```

144 plot(1:21, result(1,:), Time, result(2,:), Time, result(3,:))
145 title('Path of lumber stock')
146 xlabel('Time Period')
147 ylabel('Stock of Lumber')
148 legend('lower bound', 'mean', 'upper bound')
149 saveas(gcf, 'prob5-2.png')
150
151 %% Problem 6 Tauchen with coarse grid
152
153 clear prob grid vinitial vrevused decision mean
154
155 Z = 5; % grid points generated
156
157 [prob, grid]=tauchen(Z,p0,rho,sigma);
158 disp(['The dimensions of prob are ' num2str(size(prob)) ])
159 disp(['The dimensions of grid are ' num2str(size(grid)) ])
160
161 vinitial=zeros(N,Z);
162 vrevused=zeros(N,Z);
163 decision=zeros(N,Z);
164
165 invest=kron(ones(1,Z),k');
166 disp(['The dimensions of harvest are ' num2str(size(invest)) ])
167 ONE=ones(N,1);
168
169 vinitial=zeros(N,Z);
170 vrevused=zeros(N,Z);
171 decision=zeros(N,Z);
172
173 invest=kron(ones(1,Z),k');
174 disp(['The dimensions of harvest are ' num2str(size(invest)) ])
175 ONE=ones(N,1);
176
177 diff = 1;
178 while diff > 1e-6
179
180     Ev = vinitial*prob' ;% (NxZ * ZxZ = NxZ);
181
182     Ev;
183
184     for i = 1:N
185         harvest = ( kron(ones(N,Z),k(:,i)') - invest) ;
186         revenue = (ones(N,1)*grid) .* harvest;
187         revenue(revenue < 0) = -1e6;
188         harvest = max(harvest, 0);
189         pi = revenue - 0.2 * (harvest.^(1.5));
190
191         %pi = (ones(N,1)*grid) .* harvest - 0.2 * (harvest_z.^(1.5));
192         check = pi + delta*Ev;
193         [vrevused(i,:),decision(i,:)] = max( pi + delta*Ev );

```

```

194         decision;
195     end
196     diff = norm(vrevised - vinitial) / norm(vrevised)
197     vinitial = vrevised;
198
199 end
200
201 plot(k,vrevised(:,2)',k,vrevised(:,3)',k,vrevised(:,4)')
202 xlabel('Current Lumber')
203 ylabel('Value of Firm')
204 title('Value Function')
205 legend('p=0.8268','p=1','p=1.1732')
206 saveas(gcf,'prob6_1.png')
207
208
209 drule=zeros(N,Z);
210
211 for i=1:Z
212     drule(:,i)=k(decision(:,i))';
213 end
214
215
216 plot(grid, drule(2,:), grid, drule(26,:),grid, drule(51,:),grid, drule
    (76,:),grid, drule(101,:))
217 title('Next Period Stock vs Lumber Prices')
218 xlabel('Lumber Price')
219 ylabel('Next Priod Lumber Stock')
220 legend('k=1','k=25','k=50','k=75','k=100')
221 saveas(gcf,'prob6_2.png')
222
223 T = 21; % time period
224 Time = 1:T;
225
226 % Construct transition matrix
227 P=zeros(Z*N,Z*N);
228
229 for i=1:Z
230     for j=1:Z
231         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));
232     end
233 end
234
235 % transition probability: p=1 and k = 100 (initial)
236 pint = zeros(1,N*Z); % 3 is index of p=1
237 pint(1,3*101)=1; % since period 1 state is (1,100), mass is 1 on
238
239 % Simulate the model
240 result = zeros(3,T);
241 result(:,1) =[100;100;100];

```

```

242 for t=2:21
243     pnext = pint*P; % next period probability (1 x Z*N)
244
245     probk=zeros(N,Z); % generate probability distribution for stock
246     for i=1:Z
247         probk(:,i) = pnext((i-1)*N+1:i*N)';
248     end
249     probk = sum(probk'); % summing over for all price
250
251     cdf = cumsum(probk); % get cdf
252
253     lowerind = max(find(cdf<=0.05)); % get 5-percentile index
254     upperind = min(find(cdf>=0.95)); % get 95-percentile index
255
256     mean = probk*k';
257
258     LB = k(lowerind); % lower bound
259     UB = k(upperind);
260
261     result(:,t) = [LB mean UB]; % store result
262
263     pint = pnext; % update today's distribution
264 end
265
266 plot(1:21, result(1,:), Time, result(2,:), Time, result(3,:))
267 title('Path of lumber stock')
268 xlabel('Time Period')
269 ylabel('Stock of Lumber')
270 legend('lower bound', 'mean', 'upper bound')
271 saveas(gcf, 'prob6_3_2.png')
272
273 diary off
274
275 %%
276
277 % T = 21; % time period
278 % Time = 1:T;
279 % sim = 100; % number of simulation
280 %
281 % path_p = zeros(sim,T);
282 % path_p(:,1) = 11;
283 %
284 % generate the price path (20 period) 1000 times
285 % for s = 1:sim
286 %     for i = 2:21
287 %         rng(s * 2019 + i);
288 %         pmf = prob(path_p(s,i-1),:);
289 %         cdf = cumsum(pmf);
290 %         r = rand;
291 %         path_p(s,i) = find(cdf>r,1);

```

```
292 %     end
293 % end
294 %
295 % % generate the lumber stock path
296 % path_k = zeros(sim,T);
297 % path_k(:,1) = N; % this is index
298 % path_stock = zeros(sim,T);
299 % path_stock(:,1) = 100;
300 %
301 % for s = 1:sim
302 %     for i = 2:21
303 %         path_k(s,i) = decision(path_k(s,i-1), path_p(s,i-1));
304 %         path_stock(s,i) = k(path_k(s,i));
305 %     end
306 % end
307 %
308 % mean = mean(path_stock);
309 % stderr = std(path_stock);
310 % ts = tinv([0.05 0.95],length(path_stock)-1)';
311 %
312 % CI = ones(2,1)*mean + ts*stderr/sqrt(length(path_stock)) ;
313 %
314 % plot(Time, mean, Time, CI(1,:), Time, CI(2,:))
315 % title('Path of lumber stock')
316 % xlabel('Time Period')
317 % ylabel('Stock of Lumber')
318 % legend('mean', 'lower bound', 'upper bound')
319 % saveas(gcf,'prob5.png')
```