

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

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

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.

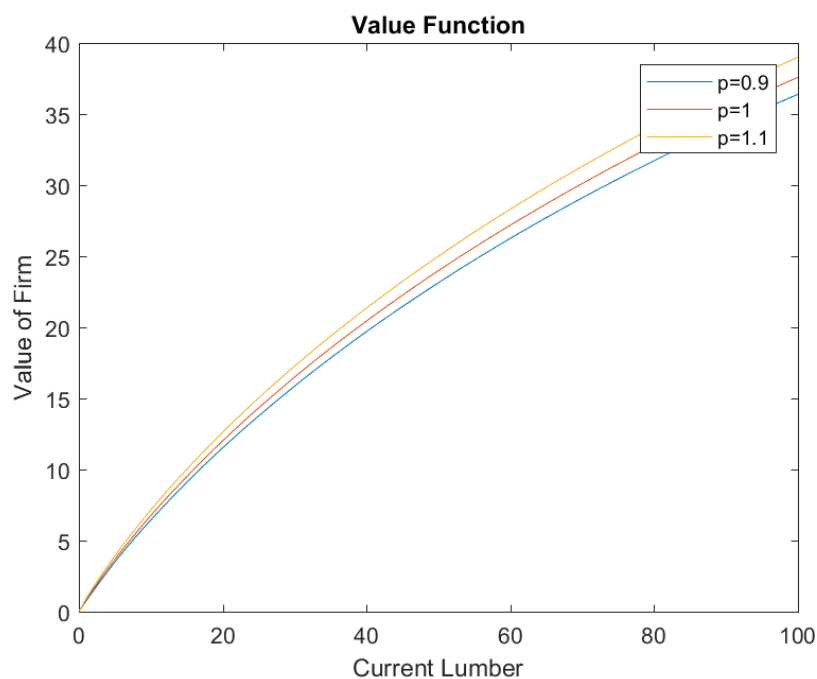
Problem 3: Value function iteration (VIF)

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.

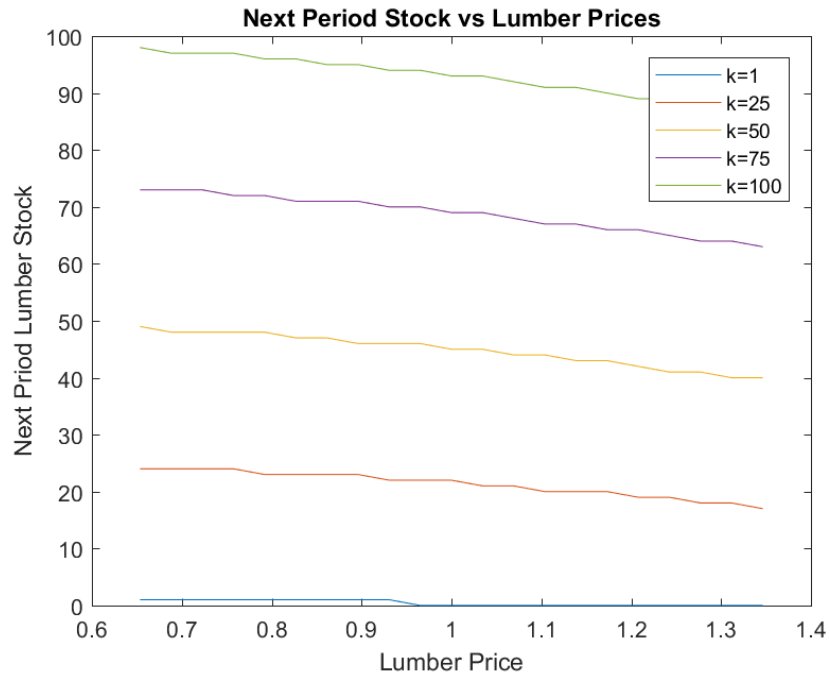
Figure 1: Problem 3



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.

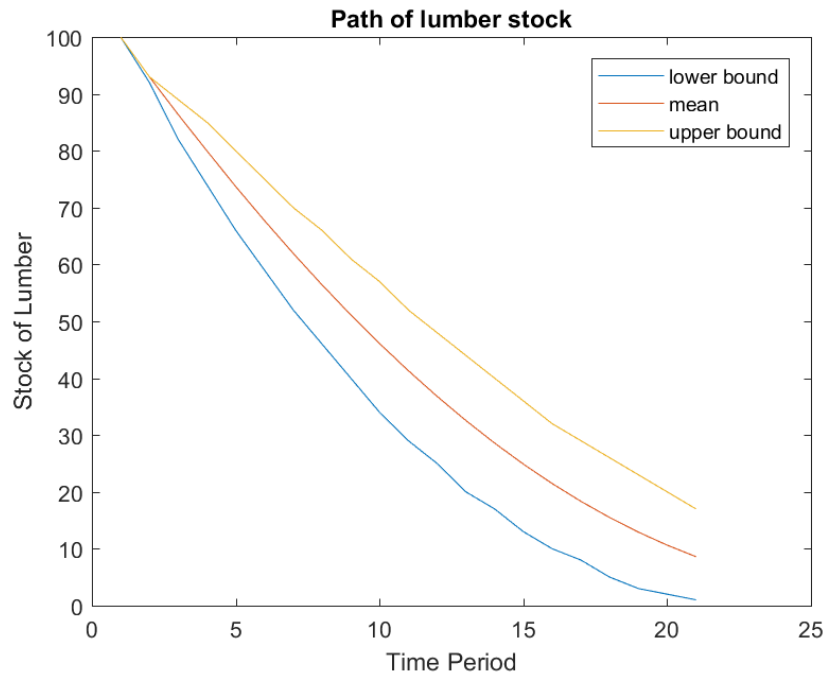
Figure 2: Problem 4



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



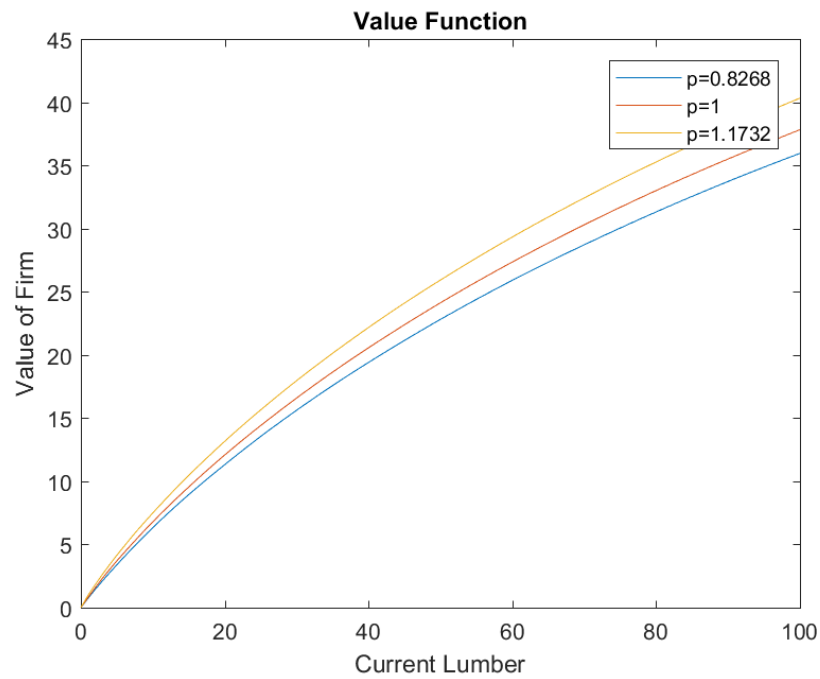
Problem 6: Results with coarse grid

Problem 6-2: Tauchen

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

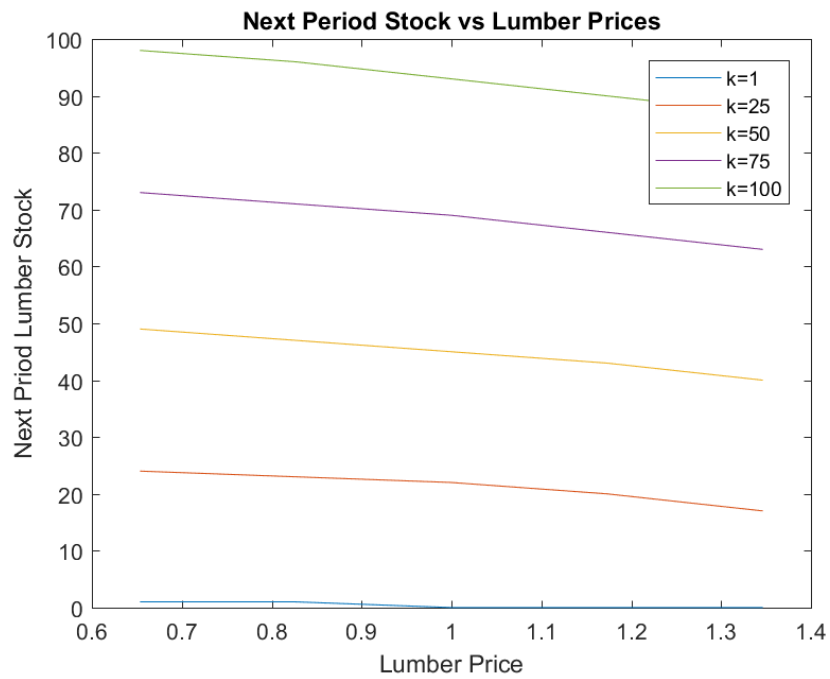
Problem 6-3: VIF

Figure 4: Problem 6-3



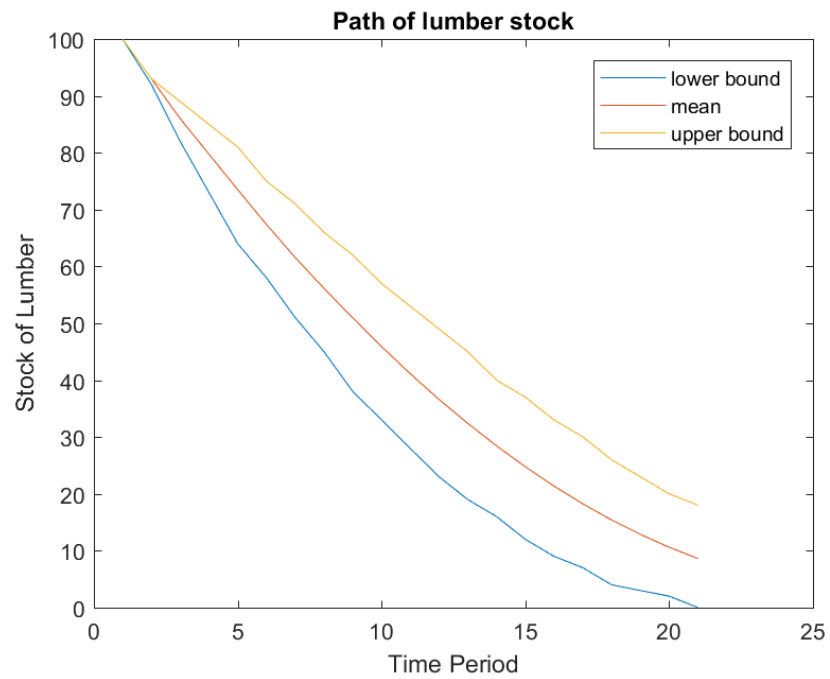
Problem 6-4: Next period stock as a function of price

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



```

49     revenue = (ones(N,1)*grid) .* harvest; % revenue
50     revenue(revenue < 0) = -1e6; % punish negative revenue
51     harvest = max(harvest, 0); % replace negative harvest to avoid
        imaginary num
52     pi = revenue - 0.2 * (harvest.^(1.5)); % current payoff
53     [vrevise(i,:),decision(i,:)] = max( pi + delta*Ev ); % get value
        and policy function
54     decision;
55 end
56     diff = norm(vrevise - vinitial) / norm(vrevise) % check deviance
57     vinitial = vrevise;
58
59 end
60
61 plot(k,vrevise(:,8)',k,vrevise(:,11)',k,vrevise(:,14)')
62 xlabel('Current Lumber')
63 ylabel('Value of Firm')
64 title('Value Function')
65 legend('p=0.9','p=1','p=1.1')
66 saveas(gcf,'prob3.png')
67
68 %
69 % %%
70 %
71 % plot(grid, decision(2,:), grid, decision(26,:),grid, decision(51,:),grid,
        decision(76,:),grid, decision(101,:))
72 % title('Next Period Stock as a Function of Price')
73 % xlabel('Price')
74 % ylabel('Next period stock of lumber')
75
76
77 %% Problem 4
78
79 drule=zeros(N,Z);
80
81 for i=1:Z
82     drule(:,i)=k(decision(:,i)); % get decision rule
83 end
84
85 plot(grid, drule(2,:), grid, drule(26,:),grid, drule(51,:),grid, drule
        (76,:),grid, drule(101,:))
86 title('Next Period Stock vs Lumber Prices')
87 xlabel('Lumber Price')
88 ylabel('Next Priod Lumber Stock')
89 legend('k=1','k=25','k=50','k=75','k=100')
90 saveas(gcf,'prob4.png')
91
92
93 %% Problem 5 Expected path
94

```

```

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

```

```
144 saveas(gcf, 'prob5_2.png')
145
146 %% Problem 6 Tauchen with coarse grid
147
148 clear prob grid vinitial vrevused decision mean
149
150 Z = 5; % grid points generated
151
152 [prob, grid]=tauchen(Z,p0,rho,sigma);
153 disp(['The dimensions of prob are ' num2str(size(prob)) ])
154 disp(['The dimensions of grid are ' num2str(size(grid)) ])
155
156 vinitial=zeros(N,Z);
157 vrevused=zeros(N,Z);
158 decision=zeros(N,Z);
159
160 invest=kron(ones(1,Z),k');
161 disp(['The dimensions of harvest are ' num2str(size(invest)) ])
162 ONE=ones(N,1);
163
164 vinitial=zeros(N,Z);
165 vrevused=zeros(N,Z);
166 decision=zeros(N,Z);
167
168 invest=kron(ones(1,Z),k');
169 disp(['The dimensions of harvest are ' num2str(size(invest)) ])
170 ONE=ones(N,1);
171
172 diff = 1;
173 while diff > 1e-6
174
175     Ev = vinitial*prob' ;% (NxZ * ZxZ = NxZ);
176
177     Ev;
178
179     for i = 1:N
180         harvest = ( kron(ones(N,Z),k(:,i)') - invest) ;
181         revenue = (ones(N,1)*grid) .* harvest;
182         revenue(revenue < 0) = -1e6;
183         harvest = max(harvest, 0);
184         pi = revenue - 0.2 * (harvest.^(1.5));
185
186         %pi = (ones(N,1)*grid) .* harvest - 0.2 * (harvest_z.^(1.5));
187         check = pi + delta*Ev;
188         [vrevused(i,:),decision(i,:)] = max( pi + delta*Ev );
189         decision;
190     end
191     diff = norm(vrevused - vinitial) / norm(vrevused)
192     vinitial = vrevused;
193
```

```

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

```

```

242     probk(:,i) = pnext((i-1)*N+1:i*N)';
243 end
244 probk = sum(probk'); % summing over for all price
245
246 cdf = cumsum(prob); % get cdf
247
248 lowerind = max(find(cdf <= 0.05)); % get 5-percentile index
249 upperind = min(find(cdf >= 0.95)); % get 95-percentile index
250
251 mean = probk*k';
252
253 LB = k(lowerind); % lower bound
254 UB = k(upperind);
255
256 result(:,t) = [LB mean UB]; % store result
257
258 pint = pnext; % update today's distribution
259 end
260
261 plot(1:21, result(1,:), Time, result(2,:), Time, result(3,:))
262 title('Path of lumber stock')
263 xlabel('Time Period')
264 ylabel('Stock of Lumber')
265 legend('lower bound', 'mean', 'upper bound')
266 saveas(gcf, 'prob6_3-2.png')
267
268 %%%
269
270 % T = 21; % time period
271 % Time = 1:T;
272 % sim = 100; % number of simulation
273 %
274 % path_p = zeros(sim,T);
275 % path_p(:,1) = 11;
276 %
277 % % generate the price path (20 period) 1000 times
278 % for s = 1:sim
279 %     for i = 2:21
280 %         rng(s * 2019 + i);
281 %         pmf = prob(path_p(s,i-1),:);
282 %         cdf = cumsum(pmf);
283 %         r = rand;
284 %         path_p(s,i) = find(cdf > r, 1);
285 %     end
286 % end
287 %
288 % % generate the lumber stock path
289 % path_k = zeros(sim,T);
290 % path_k(:,1) = N; % this is index
291 % path_stock = zeros(sim,T);

```

```
292 % path_stock(:,1) = 100;
293 %
294 % for s = 1:sim
295 %     for i = 2:21
296 %         path_k(s,i) = decision(path_k(s,i-1), path_p(s,i-1));
297 %         path_stock(s,i) = k(path_k(s,i));
298 %     end
299 % end
300 %
301 % mean = mean(path_stock);
302 % stderr = std(path_stock);
303 % ts = tinv([0.05 0.95],length(path_stock)-1)';
304 %
305 % CI = ones(2,1)*mean + ts*stderr/sqrt(length(path_stock)) ;
306 %
307 % plot(Time, mean, Time, CI(1,:), Time, CI(2,:))
308 % title('Path of lumber stock')
309 % xlabel('Time Period')
310 % ylabel('Stock of Lumber')
311 % legend('mean', 'lower bound', 'upper bound')
312 % saveas(gcf, 'prob5.png')
```