Paul Le Tran (working with Zhenghao Li, Shreeyesh Menon, and Maite Sarmiento)
plt377
08/05/22
ECO384G

Problem Set 7 (Problem Set 4 for spring 2022, Problem Set 2 for Chris)

1). We assume that the share of capital is $\alpha = 0.4$; that the discount factor is $\beta = 0.95$ and interest rate is $r = 1/\beta - 1$; that the depreciation rate is $\delta = 0.05$; and that fixed costs for firms are $F = 4$. We solve the firm's profit maximisation problem in this fixed cost model using basic value function iteration. Figure 1 below displays the value functions calculated for different productivity shock levels (given in the problem). In all three instances, we see the two characteristic "kinks" in the value function. As discussed in class, these threshold values are the values that firms adjust their capital stock holding levels to whenever their current holding levels are not in-between the threshold values. In other words, we see the "region of inaction" in all three value functions.

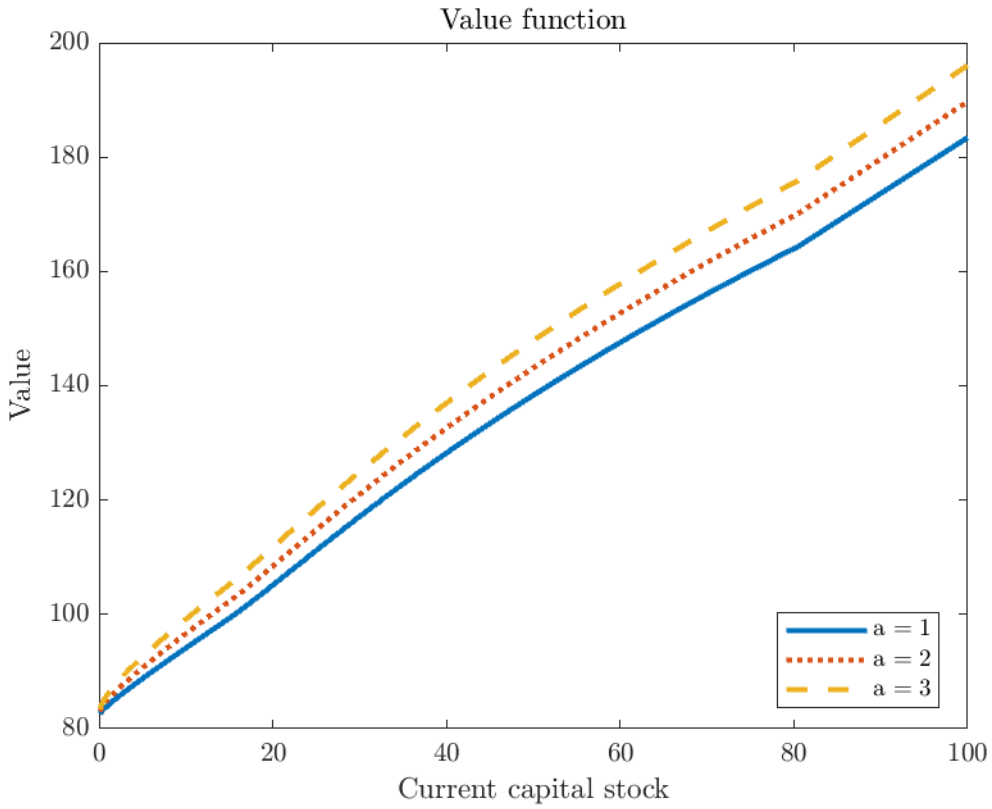

Figure 1: Value functions for different values of productivity shocks a. Used basic value function iteration.

2). We solve the same fixed cost model but with more efficient value function iteration. More specifically, we utilise vectorisation with the same parameter values assumed in problem 1. Figure 2 below verifies that the code produces the same results.

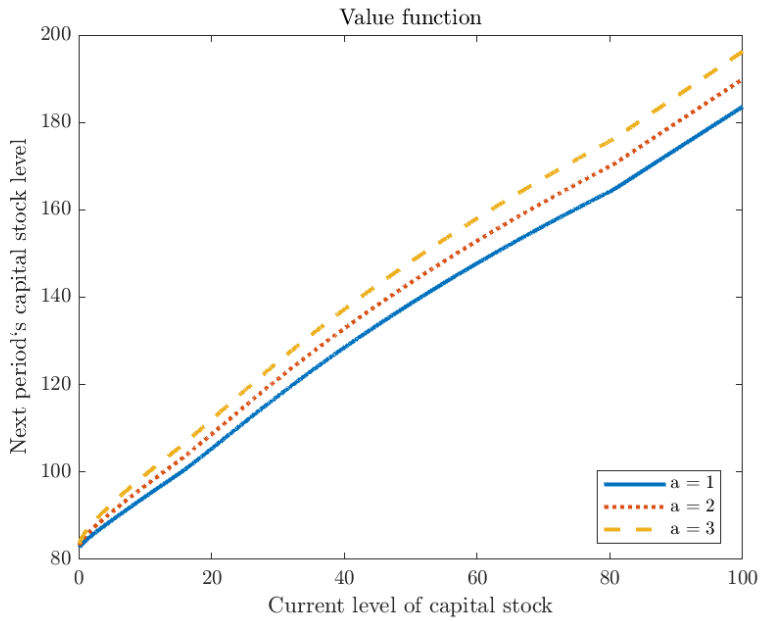

Figure 2: Value functions for different values of productivity shocks a. Used value function iteration, vectorised.

Additionally, we investigate what the policy functions for investment will look like in this problem. Figure 3 below displays the results. In contrast to the value function plots, we immediately see the threshold values of capital stock holdings and consequently, the "region of inaction" more clearly. The threshold values are roughly 17 and 80, respectively.

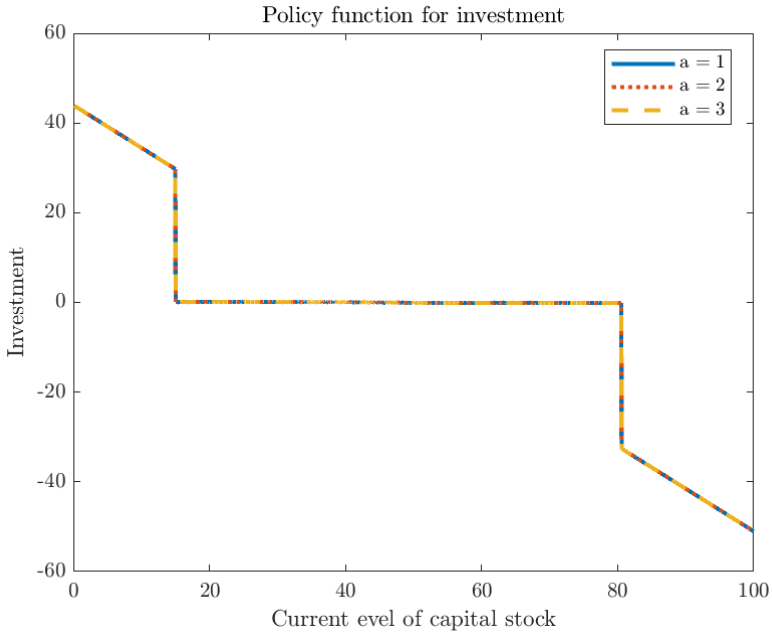

Figure 3: Policy functions for investment for different values of productivity shocks a. Used value function iteration, vectorised.

3). We now introduce uncertainty shocks in this problem, which is a new state variable u. It has two states: A low and high state. The transition probabilities are given in the problem set. Figure 4 below displays the value functions and Figure 5 on the next page displays the policy functions for investment. One interesting thing that we note is that the value functions do not differ whatsoever with respect to varying uncertainty shocks. This is observed in the policy functions for investment as well. Furthermore, we see that both plots match the fixed cost model where there are no uncertainty shocks. It is possible that this is due to coding error, but assuming this is not the case, our results seem to indicate that uncertainty shocks as an additional state variable do not affect the outcomes of the fixed cost investment model.

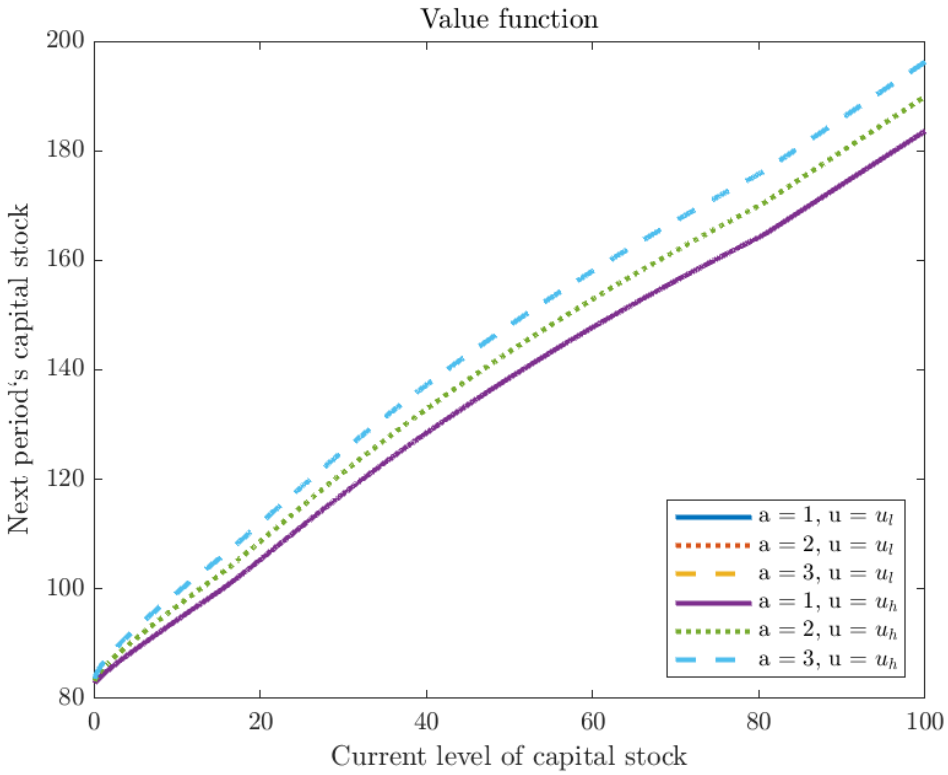

Figure 4: Value functions for different values of productivity shocks a and uncertainty shocks u. Used value function iteration, vectorised.

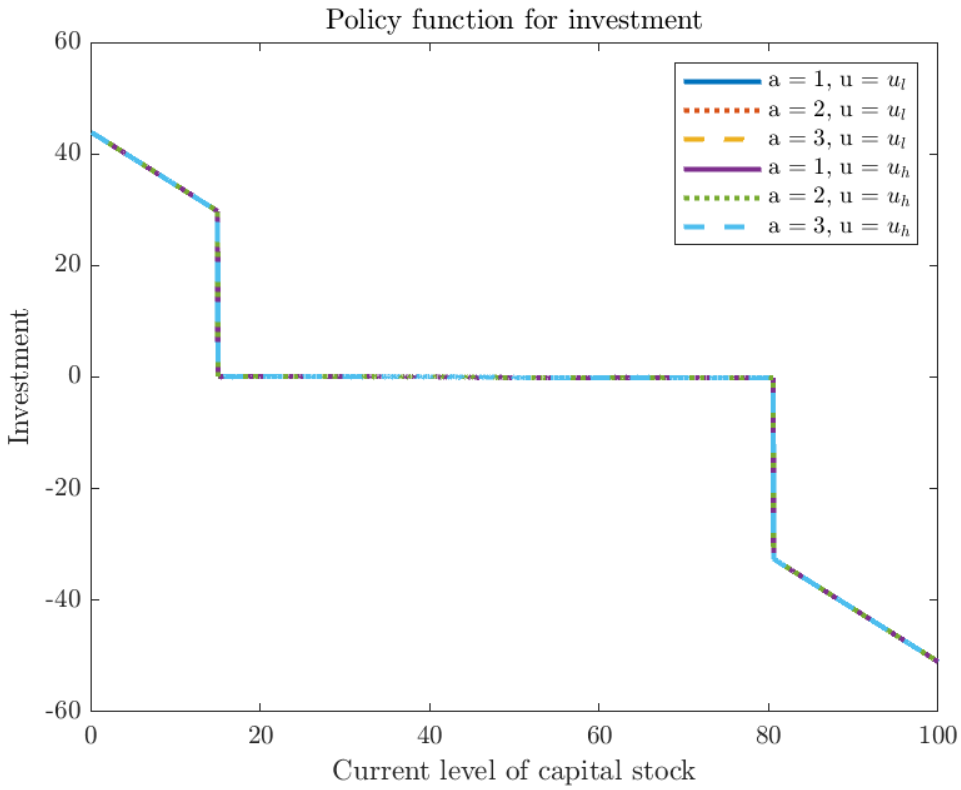

Figure 5: Policy functions for investment for different values of productivity shocks a and uncertainty shocks u. Used value function iteration, vectorised.

4). In the fixed cost investment model without uncertainty shocks, we now introduce productivity shocks, z, that follows the AR(1) process $z_t = 0.9z_{t-1} + \epsilon_t$. Because we are assuming that the model only has two state variables and not three, we assume that $a = 1$ in this model modification. The parameter estimates are the same as in problems 1 and 2. We use Tauchen (1986)'s procedure to discretise the AR(1) process for the transition probabilities. Because we are using basic value function iteration, we are showing only the plots of value functions. These are displayed below in Figure 6. Like in the original model, we see the kinks in the value functions that represent the thresholds for the "region of inaction". However, a difference from the original model is that this region seems to occur at lower levels of current capital stock levels.

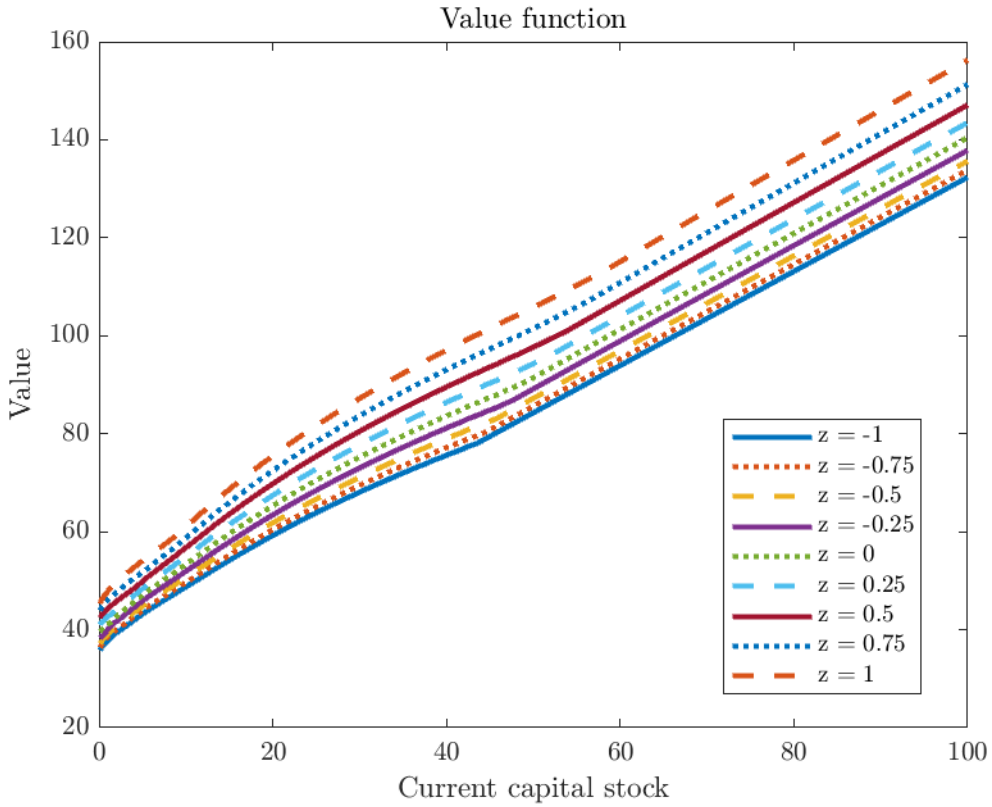

Figure 6: Value functions for different values of productivity shocks z. Assuming z follows the AR(1) process $z_t = 0.9z_{t-1} + \epsilon_t$. Used basic value function iteration.

Appendix: The following contain the main Matlab code used to create the outputs discussed above for this problem set.

Code for problem 1:

```
%=========================================================================
%% Setting up workspace
clear all;
close all;
clc;

home_dir = 'Path\to\programmes';

% Setting text interpreter to latex
set(groot, 'defaultAxesTickLabelInterpreter', 'latex');
set(groot, 'defaulttextinterpreter', 'latex');
set(groot, 'defaultLegendInterpreter', 'latex');

cd(home_dir);
%=========================================================================

%=========================================================================
%% Part 1: Write your own code to solve the problem using value function
iteration
% Initialising variable holding the difference/error during the iteration
% process
diff = 1;
% Setting the maximum allowed error for the iteration process
epsi = 1e-4;
% Setting the maximum number of iterations that can be done
itmax = 1000;

% Initialising parameter for share of capital
alpha = 0.4;
% Initialising discount factor
beta = 0.95;
r = 1/beta - 1;
% Initialising depreciation rate
delta = 0.05;
% Initialising fixed cost
fc = 4;
% Creating profit function
profit = @(k, a, k_next) a*(k^(alpha)) - k_next + (1 - delta)*k -
(abs(k_next - (1 - delta).*k) >= 0.1)*fc;

% Creating discrete grid search for capital stock k and k'
kmin = 0.1;
kmax = 100;
Nk = 500;

% Creating discrete grid search for productivity shock
amin = 1;
amax = 3;
Na = 3;

% Discretisising state space
K = linspace(kmin, kmax, Nk)';
A = linspace(amin, amax, Na)';
```

```
% Creating transition probability matrix
Ptrans = [0.1 0.8 0.1]';

% Initialising first guess for value function
V = ones(Nk, Na);
Vnew = ones(Nk, Na);
Vp = ones(Nk, 1);
pi = ones(Nk, 1);

% Initialising variable to keep track of iterations
iter = 0;

%===============
% Performing VFI
%===============
tic
while  diff >= epsi && iter <= itmax
  iter = iter + 1;
  V = Vnew;

  % Looping over state variable current capital stock k
  for x = 1:Nk
    % Looping over state variable productivity shock a
    for y = 1:Na
      % Looping over next period capital stock k'
      for z = 1:Nk
        pi(z) = profit(K(x), A(y), K(z));
        Vp(z) = pi(z) + (1/(1 + r))*V(z, :)*Ptrans;
      end
      Vnew(x, y) = max(Vp);
    end
  end
diff = max(max(abs(V - Vnew)));
disp([iter diff]);
end
% Saving current version of value function
V = Vnew;
toc

%Initialising policy function
indpol = ones(Nk, Na);

% %===============
% % Performing PFI
% %===============
% % Looping over state variable current capital stock k
% for x = 1:Nk
%   % Looping over state variable productivity shock a
%   for y = 1:Na
%     % Looping over next period capital stock k'
%     for z = 1:Nk
%       pi(z) = profit(K(x), A(y), K(z));
%       Vp(z) = pi(z) + (1/(1 + r))*V(z, :)*Ptrans;
%     end
%     [m indpol(x, y)] = max(Vp);
%     pol(x, y) = K(indpol(x, y));
%   end
% end

% Plotting value function
```

```
plot(K, V(:, 1), '-', 'Linewidth', 2);
hold on
plot(K, V(:, 2), ':', 'Linewidth',2);
plot(K, V(:, 3), '--', 'Linewidth', 2);
title('Value function');
xlabel('Current capital stock');
ylabel('Value');
legend('a = 1', 'a = 2', 'a = 3', 'Location', 'Best');
hold off;
saveas(gcf, 'Path\to\graphics\1a_plot.png');
close(gcf);

% Plotting policy function for capital stock k'
% plot(K, pol(:, 1), '-', 'Linewidth', 2);
% hold on;
% plot(K, pol(:, 2), ':', 'Linewidth', 2);
% plot(K, pol(:, 3), '--', 'Linewidth', 2);
% title('Policy function of capital stock k`');
% xlabel('Current capital stock');
% ylabel('Next period`s capital stock');
% legend('a = 1', 'a = 2', 'a = 3', 'Location', 'Best');
% hold off;
% saveas(gcf, 'Path\to\graphics\1b_plot.png');
% close(gcf);
%=========================================================================
```

Code for problem 2:

```
%=========================================================================
%% Part 2: Speeding up VFI computation using vectorisation
% Initialising variable holding the difference/error during the iteration
% process
diff = 1;
% Setting the maximum allowed error for the iteration process
epsi = 1e-4;

% Initialising parameter for share of capital
alpha = 0.4;
% Initialising discount factor
beta = 0.95;
r = 1/beta - 1;
% Initialising depreciation rate
delta = 0.05;
% Initialising fixed cost
F = 4;

% Creating discrete grid search for capital stock k and k'
kmin = 0.1;
kmax = 100;
nk = 30*kmax;

% Setting discrete search grid for productivity shocks a
a = [1 2 3]';
na = length(a);

% Discretisising state space
k = linspace(kmin, kmax, nk)';
k1 = repmat(k', na, 1);
k1 = k1(:);
a1 = repmat(a, nk, 1);

% Creating transition probability matrix
Pa = [0.1 0.8 0.1];

% Initialising vector to hold fixed costs when applicable
fc = zeros(length(k1), length(k));

% Looping to see what combination of state variables and k' result in fixed
% costs to be applicable
for i = 1:length(k1)
  for j = 1:length(k)
    if abs(k(j) - (1 - delta)*k1(i)) < 0.1
      fc(i, j) = 0;
    else
      fc(i, j) = F;
    end
  end
end

% Creating profit function
Profit = a1.*(k1.^alpha) - k' + (1 - delta)*k1 - fc;

% Initialising first guess for value function
v = zeros(na, nk);
%P = repmat(Pa, na, 1);
```

```
% Performing VFI
tic
while diff >= epsi
 Ev = Pa*v(:,:);
 Evtemp = repmat(Ev, na*nk, 1);

 [vnew, b] = max(Profit + 1/(1 + r)*Evtemp, [], 2);

 diff = max(abs(v(:) - vnew));

 v = reshape(vnew, na, nk);
end
toc

b1 = reshape(b, na, nk);
kp = zeros(na, nk);

for i = 1:size(b1, 1)
  for j = 1:size(b1, 2)
    kp(i, j) = k(b1(i, j));
  end
end

% Plotting value function
plot(k', v(1, :), '-', 'Linewidth', 2);
hold on;
plot(k', v(2, :), ':', 'Linewidth', 2);
plot(k', v(3, :), '--', 'Linewidth', 2);
title('Value function');
xlabel('Current level of capital stock');
ylabel('Next period`s capital stock level');
legend('a = 1', 'a = 2', 'a = 3', 'Location', 'Best');
hold off;
saveas(gcf, 'Path\to\graphics\2a_plot.png');
close(gcf);

% Calculating investment for policy function display
xt = kp - (1 - delta)*repmat(k', na, 1);

% Plotting policy function for investment
plot(k', xt(1, :), '-', 'Linewidth', 2);
hold on;
plot(k', xt(2, :), ':', 'Linewidth', 2);
plot(k', xt(3, :), '--', 'Linewidth', 2);
title('Policy function for investment');
xlabel('Current evel of capital stock');
ylabel('Investment');
legend('a = 1', 'a = 2', 'a = 3', 'Location', 'Best');
hold off;
saveas(gcf, 'Path\to\graphics\2b_plot.png');
close(gcf);
%=========================================================================
```

Code for problem 3:

```
%=========================================================================
%% Part 3: Introducing uncertainty shocks
% Initialising variable holding the difference/error during the iteration
% process
diff = 1;
% Setting the maximum allowed error for the iteration process
epsi = 1e-4;

% Initialising probabilities and accompanying matrices
Phh = 0.8;
Pll = 0.8;
Phl = 1 - Phh;
Plh = 1 - Pll;
Pal = [0.1 0.8 0.1];
Pah = [0.2 0.6 0.2];

% Initialising first guesses for value functions
vl = zeros(na,nk);
vh = vl;

% Performing VFI
tic
while diff > epsi
  Evl = Pal*vl(:, :);
  Evltemp = repmat(Evl, na*nk, 1);
  Evh = Pah*vh(:, :);
  Evhtemp = repmat(Evh, na*nk, 1);

  [vnewl, bl] = max(Profit + 1/(1 + r)*(Plh*Evhtemp+Pll*Evltemp), [], 2);
  [vnewh, bh] = max(Profit + 1/(1 + r)*(Phh*Evhtemp+Phl*Evltemp), [], 2);

  diff = max(abs(vl(:) - vnewl) + abs(vh(:) - vnewh));

  vl = reshape(vnewl, na, nk);
  vh = reshape(vnewh, na, nk);
end
toc

bl1 = reshape(bl, na, nk);
bh1 = reshape(bh, na, nk);
kpl = zeros(na, nk);
kph = zeros(na, nk);

for i = 1:na
  for j = 1:nk
    kpl(i, j) = k(bl1(i, j));
    kph(i, j) = k(bh1(i, j));
  end
end

xtl = kpl - (1 - delta)*repmat(k', na, 1);
xth = kph - (1 - delta)*repmat(k', na, 1);

% Plotting value function
plot(k', vl(1, :), '-', 'Linewidth', 2);
hold on;
plot(k', vl(2, :), ':', 'Linewidth', 2);
```

```
plot(k', vl(3, :), '--', 'Linewidth', 2);
plot(k', vh(1, :), '-', 'Linewidth', 2);
plot(k', vh(2, :), ':', 'Linewidth', 2);
plot(k', vh(3, :), '--', 'Linewidth', 2);
title('Value function');
xlabel('Current level of capital stock');
ylabel('Next period`s capital stock');
legend('a = 1, u = $u_{l}$', 'a = 2, u = $u_{l}$', 'a = 3, u = $u_{l}$',...
  'a = 1, u = $u_{h}$', 'a = 2, u = $u_{h}$', 'a = 3, u = $u_{h}$',
'Location', 'Best');
hold off;
saveas(gcf, 'Path\to\graphics\3a_plot.png');
close(gcf);

% Plotting policy function for investment
plot(k', xtl(1, :), '-', 'Linewidth', 2);
hold on;
plot(k', xtl(2, :), ':', 'Linewidth', 2);
plot(k', xtl(3, :), '--', 'Linewidth', 2);
plot(k', xth(1, :), '-', 'Linewidth', 2);
plot(k', xth(2, :), ':', 'Linewidth', 2);
plot(k', xth(3, :), '--', 'Linewidth', 2);
title('Policy function for investment');
xlabel('Current level of capital stock');
ylabel('Investment');
legend('a = 1, u = $u_{l}$', 'a = 2, u = $u_{l}$', 'a = 3, u = $u_{l}$',...
  'a = 1, u = $u_{h}$', 'a = 2, u = $u_{h}$', 'a = 3, u = $u_{h}$',
'Location', 'Best');
hold off;
saveas(gcf, 'Path\to\graphics\3b_plot.png');
close(gcf);
%=========================================================================
```

Code for problem 4:

```
%=========================================================================
%% Part 4: Production function now has AR(1) process component
%=====
% NOTE
%=====
% We are assuming that the included productivity shock, a_{t}, included in
% the production function is a mistake on the problem set. This is because
% that would make the model substantially more difficult with three state
% variables. As such, we assume a_{t} = 1.
%=========
% END NOTE
%=========
% Setting the maximum allowed error for the iteration process
epsi = 1e-4;
% Setting the maximum number of iterations that can be done
itmax = 1000;

% Initialising parameter for share of capital
alpha = 0.4;
% Initialising discount factor
beta = 0.95;
r = 1/beta - 1;
% Initialising depreciation rate
delta = 0.05;
% Initialising fixed cost
fc = 4;

% Creating profit function
profit = @(k, z, k_next) exp(z)*(k^(alpha)) - k_next + (1 - delta)*k -
(abs(k_next - (1 - delta).*k) >= 0.1)*fc;

% Creating discrete grid search for capital stock k and k'
kmin = 0.1;
kmax = 100;
Nk = 500;

% Creating discrete grid search for AR(1) process variable z
zmin = -1;
zmax = 1;
Nz = zmax*9;

% Discretisising state space
K = linspace(kmin, kmax, Nk)';
Z = linspace(zmin, zmax, Nz)';

% Setting various values for z AR(1) process that Tauchen's procedure
% will use
rho = 0.9;
sd = 1;
ss_val = 0;

% Initialising first guess for value function
V = ones(Nk, Nz);
Vnew = ones(Nk, Nz);
Vp = ones(Nk, 1);
pi = ones(Nk, 1);
```

```
% Initialising vector to hold probability transition matrix
Ptrans = zeros(Nz, Nz);

% Keeping track of iterations
iter = 0;

%===============
% Performing VFI
%===============
tic
for i = 1:length(rho)
  for j = 1:length(sd)
    % Initialising variable holding the difference/error during the
iteration
    % process
    diff = 1;
    iter = iter + 1;
    disp([iter rho(i) sd(j)]);
    % Using Tauchen's procedure to discretise AR(1) process of z
    %=====
    % NOTE
    %=====
    % Each row of Tauchen (1986) procedure's output is the transition
    % probabilities, given the current state of variable z.
    %=========
    % END NOTE
    %=========
    Ptrans(:, :, iter) = tauchen1986(rho(i), sd(j), ss_val, Z');

    %===============
    % Performing VFI
    %===============
    while  diff >= epsi && iter <= itmax
      V = Vnew;

      % Looping over state variable current capital stock k
      for x = 1:Nk
        % Looping over state variable z
        for y = 1:Nz
          % Looping over next period capital stock k'
          for z = 1:Nk
            pi(z) = profit(K(x), Z(y), K(z));
            Vp(z) = pi(z) + (1/(1 + r))*V(z, :)*Ptrans(y, :)';
          end
          Vnew(x, y) = max(Vp);
        end
      end
      diff = max(max(abs(V - Vnew)));
      disp([iter diff]);
    end
    % Saving current version of value function
    V = Vnew;
  end
end
toc

% Plotting value function
plot(K, V(:, 1), '-', 'Linewidth', 2);
hold on
plot(K, V(:, 2), ':', 'Linewidth',2);
```

```
plot(K, V(:, 3), '--', 'Linewidth', 2);
plot(K, V(:, 4), '-', 'Linewidth', 2);
plot(K, V(:, 5), ':', 'Linewidth',2);
plot(K, V(:, 6), '--', 'Linewidth', 2);
plot(K, V(:, 7), '-', 'Linewidth', 2);
plot(K, V(:, 8), ':', 'Linewidth',2);
plot(K, V(:, 9), '--', 'Linewidth', 2);
title('Value function');
xlabel('Current capital stock');
ylabel('Value');
legend('z = -1', 'z = -0.75', 'z = -0.5', 'z = -0.25', 'z = 0', 'z = 0.25',
'z = 0.5', 'z = 0.75', 'z = 1', 'Location', 'Best');
hold off;
saveas(gcf, 'Path\to\graphics\4a_plot.png');
close(gcf);
%=========================================================================
```