6.231 Dynamic Programming
Midterm Exam Solutions, Fall 2001
November 1, 2001 2:00-4:00pm
Prof. Dimitri Bertsekas

Problem 1

(a) This problem is the same as the "asset selling" problem on p.168 except it has the same state evolution as the "case of correlated prices" on p. 173. The resulting DP algorithm is then:

$$J_k(x_k) = \begin{cases} \max\{\underbrace{(1+r)^{N-k}x_k}_{\text{sell}}, \underbrace{E_{w_k}[J_{k+1}(\lambda x_k + w_k)]}_{\text{do not sell}}\} & x_k \neq T \\ 0 & x_k = T \end{cases}$$

$$J_N(x_N) = \begin{cases} x_N & x_N \neq T \\ 0 & x_N = T \end{cases}$$

(b) It turns out for each stage k, a threshold for x_k exists above which it is optimal to sell and below which it is optimal to not sell. These thresholds are decreasing as k increases.

Let $V_k(x_k) = \frac{J_k(x_k)}{(1+r)^{N-k}}$. The DP algorithm then becomes:

$$V_k(x_k) = \begin{cases} \max\{\underbrace{x_k}_{\text{sell}}, \underbrace{(1+r)^{-1}E_{w_k}[V_{k+1}(\lambda x_k + w_k)]}_{\text{do not sell}}\} & x_k \neq T \\ 0 & x_k = T \end{cases}$$

$$V_N(x_N) = \begin{cases} x_N & x_N \neq T \\ 0 & x_N = T \end{cases}$$

The optimal stopping set for stage $N-1$ is straightforward to find:

$$\begin{aligned} V_{N-1}(x) &= \max\{x, (1+r)^{-1}E_{w_{N-1}}[\lambda x + w_{N-1}]\} \\ T_{N-1} &= \{x | x \geq \frac{\lambda x + \bar{w}}{1+r}\} \\ &= \{x | x \geq \alpha_{N-1}\} \end{aligned}$$

where $\alpha_{N-1} = \frac{\bar{w}}{1+r-\lambda}$ and $\bar{w} = E[w_k]$ for all k. It can be shown by induction, using $V_{N-1}(x)$ as a base case, that $V_k(x)$ is the following piecewise linear, convex function for $k = 0, 1, \ldots, N-1$:

$$\begin{aligned} V_k(x) &= \max\{\underbrace{x}_{\text{sell}}, \underbrace{f_k(x)}_{\text{do not sell}}\} \\ \text{where } f_k(x) &= \max\{\frac{\lambda}{1+r}x + \frac{\bar{w}}{1+r}, (\frac{\lambda}{1+r})^2 x + \frac{(\lambda+1)\bar{w}}{(1+r)^2}, \\ &\quad \ldots, (\frac{\lambda}{1+r})^{N-k} x + \frac{(\lambda^{N-k-1} + \ldots + \lambda + 1)\bar{w}}{(1+r)^{N-k}}\} \end{aligned}$$

Because $(\frac{\lambda}{1+r})^k < 1$ for $k \geq 1$, all the linear functions of $f_k(x)$ have slope less than 1, meaning $V_k(x) = x$ as $x \to \infty$ and $V_k(x) = f_k(x)$ as $x \to -\infty$. Therefore, we have

$$V_k(x) = \begin{cases} x & x > \alpha_k \\ f_k(x) & x < \alpha_k \end{cases}$$

where α_k satisfies $\alpha_k = f_k(\alpha_k)$ (we know only one such α_k exists because f_k is convex). So the optimal stopping set at stage k is $T_k = \{x | x \geq \alpha_k\}$, corresponding to the optimal policy:

$$\begin{array}{rl} \text{sell} & \text{if } x_k > \alpha_k \\ \text{do not sell} & \text{if } x_k < \alpha_k \end{array}$$

Using $V_{N-1}(x) = \max\{x, (1+r)^{-1} E_{w_{N-1}}[\lambda x + w_{N-1}]\} \geq x = V_N(x)$ for all x as a base case, it can be shown by induction that $V_k(x) \geq V_{k+1}(x)$ for all x, $k = 0, 1, \ldots, N-1$. Therefore, $\alpha_k \geq \alpha_{k+1}$ for $k = 0, 1, \ldots, N-1$.

(c) The optimal policy remains the same. Assume we can sell a fraction β of the stock on one day and the fraction $(1-\beta)$ on a different day. Then it is optimal to sell when the value of the stock exceeds $\beta\alpha_k$ and $(1-\beta)\alpha_k$ respectively. But this will happen simultaneously, so if it is optimal to sell the fraction β it is also optimal to sell the fraction $(1-\beta)$.

Problem 2

(a) Because the problem is deterministic, we may model it as a "longest" path problem. For $i \in \{1, 2\}$, we have the following DP algorithm:

$$\begin{aligned} J_k(i) &= \max\{\underbrace{r_{k+1}^i + J_{k+1}(i)}_{\text{stay}}, \underbrace{r_{k+1}^{\bar{i}} - c + J_{k+1}(\bar{i})}_{\text{switch}}\} \text{ for } k = 0, 1, \ldots, N-1 \\ J_N(i) &= 0 \end{aligned}$$

An alternative DP algorithm is:

$$\begin{aligned} J_k(i) &= r_k^i + \max\{\underbrace{J_{k+1}(i)}_{\text{stay}}, \underbrace{-c + J_{k+1}(\bar{i})}_{\text{switch}}\} \text{ for } k = 1, 2, \ldots, N-1 \\ J_N(i) &= r_N^i \end{aligned}$$

(b) If $R_k^i \leq 0$, then it is optimal to stay because there is no reason to pay c to earn a smaller profit. If $R_k^i \geq 2c$, then it is optimal to switch because the increase in profit is greater than the cost of switching to the other location and of switching back.

For the first DP algorithm, we first show $-c \leq J_k(i) - J_k(\bar{i}) \leq c$.

$$\begin{aligned} J_k(i) - J_k(\bar{i}) &= \max\{r_{k+1}^i + J_{k+1}(i), r_{k+1}^{\bar{i}} - c + J_{k+1}(\bar{i})\} - \max\{r_{k+1}^{\bar{i}} + J_{k+1}(\bar{i}), r_{k+1}^i - c + J_{k+1}(i)\} \\ &= -c + \max\{r_{k+1}^i + c + J_{k+1}(i), r_{k+1}^{\bar{i}} + J_{k+1}(\bar{i})\} \\ &\quad - \max\{r_{k+1}^{\bar{i}} + J_{k+1}(\bar{i}), r_{k+1}^i - c + J_{k+1}(i)\} \\ &= -c + \underbrace{\max\{R_{k+1}^i + c + J_{k+1}(i) - J_{k+1}(\bar{i}), 0\} - \max\{R_{k+1}^i - c + J_{k+1}(i) - J_{k+1}(\bar{i}), 0\}}_{\geq 0, \leq 2c} \end{aligned}$$

It is optimal to stay if $r_k^i + J_k(i) \geq r_k^{\bar{i}} - c + J_k(\bar{i}) \Leftrightarrow c + J_k(i) - J_k(\bar{i}) \geq R_k^i$. From above we know $c + J_k(i) - J_k(\bar{i}) \geq c - c = 0$. So if $R_k^i \leq 0$, then it is optimal to stay.

Similarly, it is optimal to switch if $r_k^i + J_k(i) \leq r_k^{\bar{i}} - c + J_k(\bar{i}) \Leftrightarrow c + J_k(i) - J_k(\bar{i}) \leq R_k^i$. From above we know $c + J_k(i) - J_k(\bar{i}) \leq c + c = 2c$. So if $R_k^i \geq 2c$, then it is optimal to switch.

For the second DP algorithm, we compare the quantities $-c + J_k(\bar{i})$, the future cost-to-go if we switch, and $J_k(i)$, the future cost-to-go if we stay.

$$\begin{aligned}(-c + J_k(\bar{i})) - J_k(i) &= -c + r_k^{\bar{i}} + \max\{J_{k+1}(\bar{i}), -c + J_{k+1}(i)\} - r_k^i - \max\{J_{k+1}(i), -c + J_{k+1}(\bar{i})\} \\ &= R_k^i + \underbrace{\max\{0, -c + J_{k+1}(i) - J_{k+1}(\bar{i})\} - \max\{0, c + J_{k+1}(i) - J_{k+1}(\bar{i})\}}_{\leq 0, \geq -2c}\end{aligned}$$

If $R_k^i \leq 0$, then $(-c + J_k(\bar{i})) - J_k(i) \leq 0$ and it is optimal to stay. If $R_k^i \geq 2c$, then $(-c + J_k(\bar{i})) - J_k(i) \geq 0$ and it is optimal to switch.

(c) We simply replace the revenues, r_k^i, with the expected revenues, $(1 - p^i)r_k^i + p^i\beta^i r_k^i$, and we may still formulate the problem as a "longest" path problem.

(d) The problem is now genuinely stochastic and cannot be modeled as a "longest" path problem. We augment the state with the current accurate forecast $w_k = [w_k^1 \; w_k^2]'$, where w_k^i is equal to β^i with probability p^i (the probability it rains) and is equal to 1 otherwise.

$$\begin{aligned}J_k(i, w_k) &= \max\{r_{k+1}^i w_k^i + E[J_{k+1}(i, w_{k+1})], r_{k+1}^{\bar{i}} w_k^{\bar{i}} - c + E[J_{k+1}(\bar{i}, w_{k+1})]\} \text{ for } k = 0, 1, \ldots, N-1 \\ J_N(i, w_N) &= 0\end{aligned}$$

Problem 3

(a) This problem allows two controls at each stage, search or stop searching. Once we decide to stop searching at a particular time, we cannot search at future times. Let k^* be the optimal stage at which to stop searching (assuming the treasure has not yet been found), given the a priori probability is $p_0 < 1$. Let T be the event that the treasure is present at the site. Consider the evolution of p_k for $0 \leq k \leq k^*$, assuming the treasure is not found:

$$\begin{aligned}p_k &= P(T | k \text{ unsuccessful searches}) \\ &= \frac{p_0(1-\beta)^k}{p_0(1-\beta)^k + 1 - p_0}\end{aligned}$$

From the above equation we see that $p_k > p_{k+1}$ for $k = 0, 1, \ldots$ and $p_k \to 0$ as $k \to \infty$. The book tells us we know by induction that we stop searching if $p_k \leq \frac{C}{\beta V}$ where $\frac{C}{\beta V} > 0$. Therefore k^* is the smallest k such that $p_k \leq \frac{C}{\beta V}$. Since $\bar{J}_k(p_k) = 0$ for $k^* \leq k \leq N$, $\bar{J}_0(p_0) = 0$ is the same for all $N \geq k^*$.

(b)

$$\begin{aligned}\bar{J}_0(p_0) &= E[\text{reward from } k^* \text{ searches}] \\ &= VP(\text{find treasure within } k^* \text{ searches}) - CE[\text{number of searches}] \\ &= VP(T)P(\text{find treasure within } k^* \text{ searches} | T) \\ &\quad -C(P(T)E[\text{number of searches}|T] + P(\bar{T})E[\text{number of searches}|\bar{T}]) \\ &= Vp_0(1 - (1-\beta)^{k^*}) - C(p_0\frac{1 - (1-\beta)^{k^*}}{\beta} + (1 - p_0)k^*)\end{aligned}$$

(c) The probabilities p_k^1 and p_k^2 each evolve the same way as p_k from part (a). The possible controls at each stage are search 1, search 2, or do not search. If we do not search, the expected reward-to-go is 0. If we search site i, we incur a cost C, a reward V^i if we find the treasure, and a future reward-to-go $\bar{J}_{k+1}(p_{k+1}^1, p_{k+1}^2)$.

$$
\begin{aligned}
\bar{J}_k(p_k^1, p_k^2) &= \max\{0, \\
&\quad -C + p_k^1\beta^1(V^1 + \bar{J}_{k+1}(0, p_k^2)) + (1 - p_k^1\beta^1)\bar{J}_{k+1}(\frac{p_k^1(1-\beta^1)}{p_k^1(1-\beta^1) + 1 - p_k^1}, p_k^2), \\
&\quad -C + p_k^2\beta^2(V^2 + \bar{J}_{k+1}(p_k^1, 0)) + (1 - p_k^2\beta^2)\bar{J}_{k+1}(p_k^1, \frac{p_k^2(1-\beta^2)}{p_k^2(1-\beta^2) + 1 - p_k^2})\}
\end{aligned}
$$

(d) Let k_1^* be the optimal stage at which to stop searching site 1 (assuming treasure 1 has not yet been found), given p_0^1 and assuming site 1 is the only site. Let k_2^* be the optimal stage at which to stop searching site 2 (assuming treasure 2 has not yet been found), given p_0^2 and assuming site 2 is the only site. For $i = 1, 2$, we have from part (a) that k_i^* is the smallest k such that $p_k^i \leq \frac{C}{\beta^i V^i}$.

The optimal maximum number of searches (for both sites) is $k_1^* + k_2^*$. We no longer search at stage k if $k > k_1^* + k_2^*$. So if $N > k_1^* + k_2^*$, we have $\bar{J}_k(p_k^1, p_k^2) = 0$ for $k_1^* + k_2^* < k \leq N$, meaning $\bar{J}_0(p_0^1, p_0^2)$ is the same for all $N > k_1^* + k_2^*$.

Because $N > k_1^* + k_2^*$, we have enough total number of searches that the search requirements of one site do not force us to stop searching the other site before its optimal maximum number of searches has been reached. Therefore, the optimal policy is to search site 1 k_1^* times or until treasure 1 is found and search site 2 k_2^* times or until treasure 2 is found. The order of the searches does not matter.