Dynamic Programming and Optimal Control **Autumn 2014**

Richard Weber

Problems Sheet 1

1. A warehouse stocking a single commodity has total capacity M. That is, the stock level is at all times must lie between 0 and M. Let u_t be the amount released from the warehouse during week t, at unit price b_t, and let v_t be the amount subsequently bought in at the end of that week, at unit price c_t. All these prices are known at the start of week 1; stock left at the end of week h has unit value a. The amounts bought and sold are non-negative and chosen by the warehouse manager to maximize total net profit, subject to the capacity constraint. Let x_t denote the stock held at the beginning of week t. Show that the the terminal value, plant equation and dynamic programming equations are:

$$F(x,h) = ax, \qquad x_{t+1} = x_t - u_t + v_t$$

$$F(x,t) = \max_{0\le u\le x} \max_{0\le v\le M-x+u} \Big[b_t u - c_t v + F(x-u+v, t+1) \Big], \quad t < h.$$

Hence show that the value function takes the linear form, $F(x_t,t) = \alpha_t + \beta_t x_t$, where the coefficients can be calculated recursively from

$$\alpha_t = \alpha_{t+1} + \max\big[0, M(\beta_{t+1} - c_t)\big] \quad \text{and} \quad \beta_t = \max\big[b_t, \min(\beta_{t+1}, c_t)\big],$$

where $\alpha_h = 0$, $\beta_h = a$, and find the optimal policy in terms of these. Is the optimal control bang-bang?

2. Suppose that the matrix M_k is of dimension $n_k \times n_{k+1}$, $k \in \{1, \dots, h\}$. We wish to compute the product $M_1 M_2 \cdots M_h$. Notice that the order of multiplication makes a difference. For example, if $(n_1, n_2, n_3, n_4) = (1, 10, 1, 10)$, the calculation $(M_1 M_2) M_3$ requires 20 scalar multiplications, but the calculation $M_1(M_2 M_3)$ requires 200 scalar multiplications. Indeed, multiplying a $m \times n$ matrix by a $n \times k$ matrix requires mnk scalar multiplications. Let $F(n_1, n_2, \dots, n_{h+1}; h)$ be the minimal total number of scalar multiplications required to compute $M_1 M_2 \cdots M_h$. Explain why the dynamic programming equation is

$$F(n_1, n_2, \dots, n_{k+1}; k) = \min_{1<i<k+1} \left\{ n_{i-1} n_i n_{i+1} + F(n_1, \dots, n_{i-1}, n_{i+1}, \dots, n_{k+1}; k-1) \right\},$$

$k = 1, \dots, h$. Hence describe an algorithm which finds the multiplication order requiring least scalar multiplications. Solve the problem for

(a) $h = 3$, $(n_1, n_2, n_3, n_4) = (2, 10, 5, 1)$;

(b) $h = 4$, $(n_1, n_2, n_3, n_4, n_5) = (2, 10, 1, 5, 1)$.

Show that as h increases the amount of effort required to find the optimal order increases faster than any polynomial function of h.

3. A deck of cards is thoroughly shuffled and placed face down on the table. You turn over cards one by one, counting the numbers of reds and blacks you have seen so far. Exactly once, whenever you like, you may bet that the next card you turn over will be red. If correct you win £1000.

Let $F(r, b)$ be the probability of winning if you play optimally, beginning from a point at which you have not yet bet and you know that exactly r red and b black cards remain in the face down pack. Find $F(26, 26)$ and your optimal strategy.

Arguably, it should be possible to win the £1000 with a probability greater than 1/2 because you can wait until you have seen more black cards than red and then bet that the next card is red. Explain why this argument is wrong.

4. A gambler has the opportunity to bet on a sequence on N coin tosses. The probability of heads on the nth toss is known to be p_n, $n = 1, \ldots, N$. For the nth toss he may stake any non-negative amount not exceeding his current capital (which is his initial capital plus his winnings so far) and call 'heads' or 'tails'. If he calls correctly then he retains his stake and wins an amount equal to it, but if he calls incorrectly he loses his stake. Let $X_0 \geq 0$ denote his initial capital and X_N his capital after the final toss. Determine how the gambler should call and how much he should stake for each toss in order to maximize $E[\log X_N]$.

How would your answer differ if the aim is to maximize $E[X_N]$?

5. Initially an investor has one unit of capital and at the beginning of each of n periods of time he must divide his capital between fixed interest bonds, or shares in a certain company. If u units of capital are invested in bonds at time t $(y = 0, 1, \ldots, n-1)$, they yield bu units at time $t+1$, while u units of capital invested in shares at time t yields $S_r u$ units at time $t+1$. Here b is a constant and $S_0, \ldots, S_{n-1}$ are independent identically distributed non-negative random variables. The value of S_t is not known until time $t+1$. Let X_t denote the capital at time $t = 0, 1, \ldots, n$. Find the investment policy that maximizes $E\sqrt{X_n}$.

Show that if the investor adopts this investment policy then in each period he will direct his capital into *both* investments provided

$$E\sqrt{S_0}\Big/E\left(1/\sqrt{S_0}\right) < b < ES_0.$$

[You may assume that $ES_0 < \infty$ and $E(1/S_0) < \infty$.]

6. The Greek adventurer Theseus is trapped in a room from which lead n passages. Theseus knows that if he enters passage i $(i = 1, \ldots, n)$ one of three fates will befall him: he will escape with probability p_i, he will be killed with probability q_i, and with probability r_i $(= 1 - p_i - q_i)$ he will find the passage to be a dead end and be forced to return to the room. The fates associated with different passages are independent. Establish the order in which Theseus should attempt the passages if he wishes to maximize his probability of eventual escape.

7. Consider a burglar who loots some house every night. His profit from successive crimes forms a sequence of independent random variables, each having the exponential distribution with mean $1/\lambda$. Each night there is a probability q, $0 < q < 1$, of his being caught and forced to return his whole profit. If he has the choice, when should the burglar retire so as to maximize his total expected profit?

8. A busy student has to complete N tasks. Each task k has a deadline d_k and the time it takes the student to complete it is t_k. The student can work on only one task at a time and must complete it before moving on to a new task. For a given order of completion of the tasks, denote by c_k the time of completion of task k.

The student wants to order the tasks so as to minimize the maximum tardinesss, given by

$$\max_{k \in \{1, \ldots, N\}} \max(0, c_k - d_k).$$

Use an interchange argument to show that it is optimal to complete the tasks in the order of their deadlines (i.e. to do the task with the closest deadline first).