1.3 THE DYNAMIC PROGRAMMING ALGORITHM

The dynamic programming (DP) technique rests on a very simple idea, the *principle of optimality*. The name is due to Bellman, who contributed a great deal to the popularization of DP and to its transformation into a systematic tool. Roughly, the principle of optimality states the following rather obvious fact.

Principle of Optimality

Let $\pi^* = \{\mu_0^*, \mu_1^*, \ldots, \mu_{N-1}^*\}$ be an optimal policy for the basic problem, and assume that when using π^*, a given state x_i occurs at time i with positive probability. Consider the subproblem whereby we are at x_i at time i and wish to minimize the "cost-to-go" from time i to time N

$$E\left\{g_N(x_N) + \sum_{k=i}^{N-1} g_k\big(x_k, \mu_k(x_k), w_k\big)\right\}.$$

Then the truncated policy $\{\mu_i^*, \mu_{i+1}^*, \ldots, \mu_{N-1}^*\}$ is optimal for this subproblem.

The intuitive justification of the principle of optimality is very simple. If the truncated policy $\{\mu_i^*, \mu_{i+1}^*, \ldots, \mu_{N-1}^*\}$ were not optimal as stated, we would be able to reduce the cost further by switching to an optimal policy for the subproblem once we reach x_i. For an auto travel analogy, suppose that the fastest route from Los Angeles to Boston passes through Chicago. The principle of optimality translates to the obvious fact that the Chicago to Boston portion of the route is also the fastest route for a trip that starts from Chicago and ends in Boston.

The principle of optimality suggests that an optimal policy can be constructed in piecemeal fashion, first constructing an optimal policy for the "tail subproblem" involving the last stage, then extending the optimal policy to the "tail subproblem" involving the last two stages, and continuing in this manner until an optimal policy for the entire problem is constructed. The DP algorithm is based on this idea. We introduce the algorithm by means of an example.

The DP Algorithm for the Inventory Control Example

Consider the inventory control example of the previous section and the following procedure for determining the optimal ordering policy starting with the last period and proceeding backward in time.

Period $N-1$: Assume that at the beginning of period $N-1$ the stock is x_{N-1}. Clearly, no matter what happened in the past, the inventory manager should order the amount of inventory that minimizes over $u_{N-1} \geq 0$ the sum of the ordering cost and the expected terminal holding/shortage cost $cu_{N-1} + E\{R(x_N)\}$, which can be written as

$$cu_{N-1} + \mathop{E}_{w_{N-1}}\big\{R(x_{N-1} + u_{N-1} - w_{N-1})\big\}.$$

Adding the holding/shortage cost of period $N-1$, the optimal cost for the last period (plus the terminal cost) is

$$J_{N-1}(x_{N-1}) = r(x_{N-1}) + \min_{u_{N-1}\geq 0}\left[cu_{N-1} + \mathop{E}_{w_{N-1}}\big\{R(x_{N-1} + u_{N-1} - w_{N-1})\big\}\right].$$

Naturally, J_{N-1} is a function of the stock x_{N-1}. It is calculated either analytically or numerically (in which case a table is used for computer storage of the function J_{N-1}). In the process of calculating J_{N-1}, we obtain the optimal inventory policy $\mu_{N-1}^*(x_{N-1})$ for the last period; $\mu_{N-1}^*(x_{N-1})$ is the value of u_{N-1} that minimizes the right-hand side of the preceding equation for a given value of x_{N-1}.

Period $N-2$: Assume that at the beginning of period $N-2$ the stock is x_{N-2}. It is clear that the inventory manager should order the amount of inventory that minimizes not just the expected cost of period $N-2$ but rather the

(expected cost of period $N-2$) + (expected cost of period $N-1$, given that an optimal policy will be used at period $N-1$),

which is equal to

$$r(x_{N-2}) + cu_{N-2} + E\big\{J_{N-1}(x_{N-1})\big\}.$$

Using the system equation $x_{N-1} = x_{N-2} + u_{N-2} - w_{N-2}$, the last term is also written as $J_{N-1}(x_{N-2} + u_{N-2} - w_{N-2})$.

Thus the optimal cost for the last two periods given that we are at state x_{N-2}, denoted $J_{N-2}(x_{N-2})$, is given by

$$J_{N-2}(x_{N-2}) = r(x_{N-2}) + \min_{u_{N-2}\geq 0}\left[cu_{N-2} + \mathop{E}_{w_{N-2}}\big\{J_{N-1}(x_{N-2} + u_{N-2} - w_{N-2})\big\}\right]$$

Again $J_{N-2}(x_{N-2})$ is calculated for every x_{N-2}. At the same time, the optimal policy $\mu_{N-2}^*(x_{N-2})$ is also computed.

Period k: Similarly, we have that at period k, when the stock is x_k, the inventory manager should order u_k to minimize

(expected cost of period k) + (expected cost of periods $k+1, \ldots, N-1$, given that an optimal policy will be used for these periods).

By denoting by $J_k(x_k)$ the optimal cost, we have

$$J_k(x_k) = r(x_k) + \min_{u_k \geq 0} \left[cu_k + \mathop{E}_{w_k} \{ J_{k+1}(x_k + u_k - w_k) \} \right], \qquad (3.1)$$

which is actually the dynamic programming equation for this problem.

The functions $J_k(x_k)$ denote the optimal expected cost for the remaining periods when starting at period k and with initial inventory x_k. These functions are computed recursively backward in time, starting at period $N-1$ and ending at period 0. The value $J_0(x_0)$ is the optimal expected cost for the process when the initial stock at time 0 is x_0. During the calculations, the optimal policy is simultaneously computed from the minimization in the right-hand side of Eq. (3.1).

The example illustrates the main advantage offered by DP. While the original inventory problem requires an optimization over the set of policies, the DP algorithm of Eq. (3.1) decomposes this problem into a sequence of minimizations carried out over the set of controls. Each of these minimizations is much simpler than the original problem.

The DP Algorithm

We now state the DP algorithm for the basic problem and show its optimality by translating in mathematical terms the heuristic argument given above for the inventory example.

Proposition 3.1: For every initial state x_0, the optimal cost $J^*(x_0)$ of the basic problem is equal to $J_0(x_0)$, where the function J_0 is given by the last step of the following algorithm, which proceeds backward in time from period $N-1$ to period 0:

$$J_N(x_N) = g_N(x_N), \qquad (3.2)$$

$$J_k(x_k) = \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \{ g_k(x_k, u_k, w_k) + J_{k+1}(f_k(x_k, u_k, w_k)) \}, \quad k = 0, 1, \ldots, N-1, \qquad (3.3)$$

where the expectation is taken with respect to the probability distribution of w_k, which depends on x_k and u_k. Furthermore, if $u_k^* = \mu_k^*(x_k)$ minimizes the right side of Eq. (3.3) for each x_k and k, the policy $\pi^* = \{\mu_0^*, \ldots, \mu_{N-1}^*\}$ is optimal.

Proof: † For any admissible policy $\pi = \{\mu_0, \mu_1, \ldots, \mu_{N-1}\}$ and each $k = 0, 1, \ldots, N-1$, denote $\pi^k = \{\mu_k, \mu_{k+1}, \ldots, \mu_{N-1}\}$. For $k = 0, 1, \ldots, N-1$, let $J_k^*(x_k)$ be the optimal cost for the $(N-k)$-stage problem that starts at state x_k and time k, and ends at time N; that is,

$$J_k^*(x_k) = \min_{\pi^k} \mathop{E}_{w_k, \ldots, w_{N-1}} \left\{ g_N(x_N) + \sum_{i=k}^{N-1} g_i\big(x_i, \mu_i(x_i), w_i\big) \right\}.$$

For $k = N$, we define $J_N^*(x_N) = g_N(x_N)$. We will show by induction that the functions J_k^* are equal to the functions J_k generated by the DP algorithm, so that for $k = 0$, we will obtain the desired result.

Indeed, we have by definition $J_N^* = J_N = g_N$. Assume that for some k and all x_{k+1}, we have $J_{k+1}^*(x_{k+1}) = J_{k+1}(x_{k+1})$. Then, since $\pi^k = (\mu_k, \pi^{k+1})$, we have for all x_k

$$\begin{aligned}
J_k^*(x_k) &= \min_{(\mu_k, \pi^{k+1})} \mathop{E}_{w_k, \ldots, w_{N-1}} \left\{ g_k\big(x_k, \mu_k(x_k), w_k\big) \right. \\
&\qquad \left. + g_N(x_N) + \sum_{i=k+1}^{N-1} g_i\big(x_i, \mu_i(x_i), w_i\big) \right\} \\
&= \min_{\mu_k} \mathop{E}_{w_k} \left\{ g_k\big(x_k, \mu_k(x_k), w_k\big) \right. \\
&\qquad \left. + \min_{\pi^k} \left[\mathop{E}_{w_{k+1}, \ldots, w_{N-1}} \left\{ g_N(x_N) + \sum_{i=k+1}^{N-1} g_i\big(x_i, \mu_i(x_i), w_i\big) \right\} \right] \right\} \\
&= \min_{\mu_k} \mathop{E}_{w_k} \left\{ g_k\big(x_k, \mu_k(x_k), w_k\big) + J_{k+1}^*\big(f_k\big(x_k, \mu_k(x_k), w_k\big)\big) \right\} \\
&= \min_{\mu_k} \mathop{E}_{w_k} \left\{ g_k\big(x_k, \mu_k(x_k), w_k\big) + J_{k+1}\big(f_k\big(x_k, \mu_k(x_k), w_k\big)\big) \right\} \\
&= \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \left\{ g_k(x_k, u_k, w_k) + J_{k+1}\big(f_k(x_k, u_k, w_k)\big) \right\} \\
&= J_k(x_k),
\end{aligned}$$

† For a strictly rigorous proof, some technical mathematical issues must be addressed; see Section 1.5. These issues do not arise if the disturbance w_k takes a finite or countable number of values and the expected values of all terms in the expression of the cost function (2.2) are well defined and finite for every admissible policy π.

completing the induction. In the second equation above, we moved the minimum over π^{k+1} inside the braced expression, using the assumption that the probability distribution of w_i, $i = k+1, \ldots, N-1$, depends only on x_i and u_i. In the third equation, we used the definition of J^*_{k+1}, and in the fourth equation we used the induction hypothesis. In the fifth equation, we converted the minimization over μ_k to a minimization over u_k, using the fact that for any function F of x and u, we have

$$\min_{\mu \in M} F\big(x, \mu(x)\big) = \min_{u \in U(x)} F(x, u),$$

where M is the set of all functions $\mu(x)$ such that $\mu(x) \in U(x)$ for all x. **Q.E.D.**

The argument of the preceding proof provides an interpretation of $J_k(x_k)$ as the optimal cost for an $(N-k)$-stage problem starting at state x_k and time k, and ending at time N. We consequently call $J_k(x_k)$ the *cost-to-go* at state x_k and time k, and refer to J_k as the *cost-to-go function* at time k.

Ideally, we would like to use the DP algorithm to obtain closed-form expressions for J_k or an optimal policy. In this book, we will consider a large number of models that admit analytical solution by DP. Even if such models encompass oversimplified assumptions, they are often very useful. They may provide valuable insights about the structure of the optimal solution of more complex models, and they may form the basis for suboptimal control schemes. Furthermore, the broad collection of analytically solvable models provides helpful guidelines for modeling; when faced with a new problem it is worth trying to pattern its model after one of the principal analytically tractable models.

Unfortunately, in many practical cases an analytical solution is not possible, and one has to resort to numerical execution of the DP algorithm. This may be quite time-consuming since the minimization in the DP Eq. (3.3) must be carried out for each value of x_k. This means that the state space must be discretized in some way (if it is not already a finite set). The computational requirements are proportional to the number of discretization points, so for complex problems the computational burden may be excessive. Nonetheless, DP is the only general approach for sequential optimization under uncertainty, and even when it is computationally prohibitive, it can serve as the basis for more practical suboptimal approaches, which will be discussed in Chapter 6.

The following examples illustrate some of the analytical and computational aspects of DP.

Example 3.1

A certain material is passed through a sequence of two ovens (see Fig. 1.3.1). Denote

x_0: initial temperature of the material,

x_k, $k = 1, 2$: temperature of the material at the exit of oven k,

u_{k-1}, $k = 1, 2$: prevailing temperature in oven k.

We assume a model of the form

$$x_{k+1} = (1-a)x_k + au_k, \qquad k = 0, 1,$$

where a is a known scalar from the interval $(0, 1)$. The objective is to get the final temperature x_2 close to a given target T, while expending relatively little energy. This is expressed by a cost function of the form

$$r(x_2 - T)^2 + u_0^2 + u_1^2,$$

where $r > 0$ is a given scalar. We assume no constraints on u_k. (In reality, there are constraints, but if we can solve the unconstrained problem and verify that the solution satisfies the constraints, everything will be fine.) The problem is deterministic; that is, there is no stochastic uncertainty. However, such problems can be placed within the basic framework by introducing a fictitious disturbance taking a unique value with probability one.

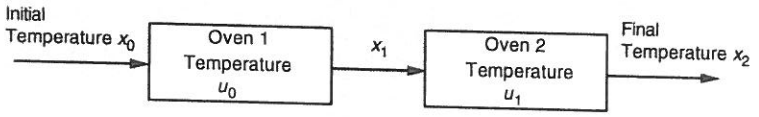


Figure 1.3.1 Problem of Example 3.1. The temperature of the material evolves according to $x_{k+1} = (1-a)x_k + au_k$, where a is some scalar with $0 < a < 1$.

We have $N = 2$ and a terminal cost $g_2(x_2) = r(x_2 - T)^2$, so the initial condition for the DP algorithm is [cf. Eq. (3.2)]

$$J_2(x_2) = r(x_2 - T)^2.$$

For the next-to-last stage, we have [cf. Eq. (3.3)]

$$\begin{aligned} J_1(x_1) &= \min_{u_1} \big[u_1^2 + J_2(x_2)\big] \\ &= \min_{u_1} \big[u_1^2 + J_2\big((1-a)x_1 + au_1\big)\big]. \end{aligned}$$

Substituting the previous form of J_2, we obtain

$$J_1(x_1) = \min_{u_1} \Big[u_1^2 + r\big((1-a)x_1 + au_1 - T\big)^2\Big]. \tag{3.4}$$

This minimization will be done by setting to zero the derivative with respect to u_1. This yields

$$0 = 2u_1 + 2ra\big((1-a)x_1 + au_1 - T\big),$$

and by collecting terms and solving for u_1, we obtain the optimal temperature for the last oven:

$$\mu_1^*(x_1) = \frac{ra\big(T-(1-a)x_1\big)}{1+ra^2}$$

Note that this is not a single control but rather a control function, a rule that tells us the optimal oven temperature $u_1 = \mu_1^*(x_1)$ for each possible state x_1.

By substituting the optimal u_1 in the expression (3.4) for J_1, we obtain

$$\begin{aligned} J_1(x_1) &= \frac{r^2a^2\big((1-a)x_1-T\big)^2}{(1+ra^2)^2} + r\left((1-a)x_1 + \frac{ra^2\big(T-(1-a)x_1\big)}{1+ra^2} - T\right)^2 \\ &= \frac{r^2a^2\big((1-a)x_1-T\big)^2}{(1+ra^2)^2} + r\left(\frac{ra^2}{1+ra^2}-1\right)^2\big((1-a)x_1-T\big)^2 \\ &= \frac{r\big((1-a)x_1-T\big)^2}{1+ra^2}. \end{aligned}$$

We now go back one stage. We have [cf. Eq. (3.3)]

$$J_0(x_0) = \min_{u_0}\big[u_0^2 + J_1(x_1)\big] = \min_{u_0}\big[u_0^2 + J_1\big((1-a)x_0 + au_0\big)\big],$$

and by substituting the expression already obtained for J_1, we have

$$J_0(x_0) = \min_{u_0}\left[u_0^2 + \frac{r\big((1-a)^2x_0 + (1-a)au_0 - T\big)^2}{1+ra^2}\right].$$

We minimize with respect to u_0 by setting the corresponding derivative to zero. We obtain

$$0 = 2u_0 + \frac{2r(1-a)a\big((1-a)^2x_0 + (1-a)au_0 - T\big)}{1+ra^2}.$$

This yields, after some calculation, the optimal temperature of the first oven:

$$\mu_0^*(x_0) = \frac{r(1-a)a\big(T-(1-a)^2x_0\big)}{1+ra^2\big(1+(1-a)^2\big)}.$$

The optimal cost is obtained by substituting this expression in the formula for J_0. This leads to a straightforward but lengthy calculation, which in the end yields the rather simple formula

$$J_0(x_0) = \frac{r\big((1-a)^2x_0 - T\big)^2}{1+ra^2\big(1+(1-a)^2\big)}.$$

This completes the solution of the problem.

One noteworthy feature in the preceding example is the facility with which we obtained an analytical solution. A little thought while tracing the steps of the algorithm will convince the reader that what simplifies the solution is the quadratic nature of the cost and the linearity of the system equation. In Section 4.1 we will see that, generally, when the system is linear and the cost is quadratic then, regardless of the number of stages N, the optimal policy is given by a closed-form expression.

Another noteworthy feature of the example is that the optimal policy remains unaffected when a zero-mean stochastic disturbance is added in the system equation. To see this, assume that the material' s temperature evolves according to

$$x_{k+1} = (1-a)x_k + au_k + w_k, \qquad k = 0, 1,$$

where w_0, w_1 are independent random variables with given distribution, zero mean

$$E\{w_0\} = E\{w_1\} = 0,$$

and finite variance. Then the equation for J_1 [cf. Eq. (3.3)] becomes

$$\begin{aligned} J_1(x_1) &= \min_{u_1}\mathop{E}_{w_1}\Big\{u_1^2 + r\big((1-a)x_1 + au_1 + w_1 - T\big)^2\Big\} \\ &= \min_{u_1}\big[u_1^2 + r\big((1-a)x_1 + au_1 - T\big)^2 \\ &\qquad + 2rE\{w_1\}\big((1-a)x_1 + au_1 - T\big) + rE\{w_1^2\}\big]. \end{aligned}$$

Since $E\{w_1\} = 0$, we obtain

$$J_1(x_1) = \min_{u_1}\Big[u_1^2 + r\big((1-a)x_1 + au_1 - T\big)^2\Big] + rE\{w_1^2\}.$$

Comparing this equation with Eq. (3.4), we see that the presence of w_1 has resulted in an additional inconsequential term, $rE\{w_1^2\}$. Therefore, the optimal policy for the last stage remains unaffected by the presence of w_1, while $J_1(x_1)$ is increased by the constant term $rE\{w_1^2\}$. It can be seen that a similar situation also holds for the first stage. In particular, the optimal cost is given by the same expression as before except for an additive constant that depends on $E\{w_0^2\}$ and $E\{w_1^2\}$.

If the optimal policy is unaffected when the disturbances are replaced by their means, we say that *certainty equivalence* holds. We will derive certainty equivalence results for several types of problems involving a linear system and a quadratic cost (see Sections 4.1, 5.2, and 5.3).

Example 3.2

To illustrate the computational aspects of DP, consider an inventory control problem that is slightly different from the one of Sections 1.1 and 1.2. In

particular, we assume that inventory u_k and the demand w_k are nonnegative integers, and that the excess demand $(w_k - x_k - u_k)$ is lost. As a result, the stock equation takes the form

$$x_{k+1} = \max(0, x_k + u_k - w_k).$$

We also assume that there is an upper bound of 2 units on the stock that can be stored, i.e. there is a constraint $x_k + u_k \leq 2$. The holding/storage cost for the kth period is given by

$$(x_k + u_k - w_k)^2,$$

implying a penalty both for excess inventory and for unmet demand at the end of the kth period. The ordering cost is 1 per unit stock ordered. Thus the cost per period is

$$g_k(x_k, u_k, w_k) = u_k + (x_k + u_k - w_k)^2.$$

The terminal cost is assumed to be 0,

$$g_N(x_N) = 0.$$

The planning horizon N is 3 periods, and the initial stock x_0 is 0. The demand w_k has the same probability distribution for all periods, given by

$$p(w_k = 0) = 0.1, \qquad p(w_k = 1) = 0.7, \qquad p(w_k = 2) = 0.2.$$

The system can also be represented in terms of the transition probabilities $p_{ij}(u)$ between the three possible states, for the different values of the control (see Fig. 1.3.2).

The starting equation for the DP algorithm is

$$J_3(x_3) = 0,$$

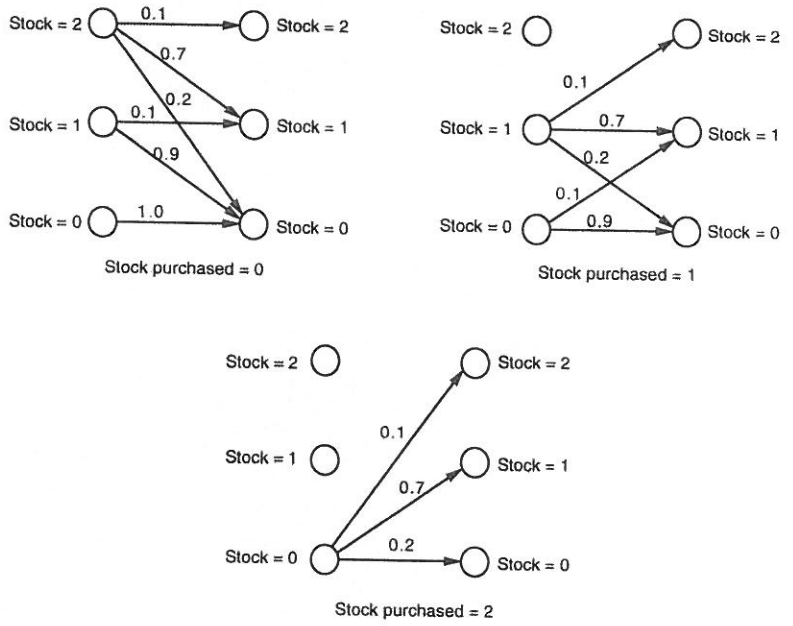
since the terminal state cost is 0 [cf. Eq. (3.2)]. The algorithm takes the form [cf. Eq. (3.3)]

$$J_k(x_k) = \min_{\substack{0 \leq u_k \leq 2 - x_k \\ u_k = 0,1,2}} E_{w_k} \left\{ u_k + (x_k + u_k - w_k)^2 + J_{k+1}\big(\max(0, x_k + u_k - w_k)\big) \right\},$$

where $k = 0, 1, 2$, and x_k, u_k, w_k can take the values 0, 1, and 2.

Period 2: We compute $J_2(x_2)$ for each of the three possible states. We have

$$\begin{aligned} J_2(0) &= \min_{u_2 = 0,1,2} E_{w_2} \left\{ u_2 + (u_2 - w_2)^2 \right\} \\ &= \min_{u_2 = 0,1,2} \left[u_2 + 0.1(u_2)^2 + 0.7(u_2 - 1)^2 + 0.2(u_2 - 2)^2 \right]. \end{aligned}$$



Stock	Stage 0 Cost-to-go	Stage 0 Optimal stock to purchase	Stage 1 Cost-to-go	Stage 1 Optimal stock to purchase	Stage 2 Cost-to-go	Stage 2 Optimal stock to purchase
0	3.67	1	2.5	1	1.3	1
1	2.67	0	1.2	0	0.3	0
2	2.608	0	1.68	0	1.1	0

Figure 1.3.2 System and DP results for Example 3.2. The transition probability diagrams for the different values of stock purchased (control) are shown. The numbers next to the arcs are the transition probabilities. The control $u = 1$ is not available at state 2 because of the limitation $x_k + u_k \leq 2$. Similarly, the control $u = 2$ is not available at states 1 and 2. The results of the DP algorithm are given in the table.

We calculate the expectation of the right side for each of the three possible values of u_2:

$$\begin{aligned} u_2 = 0 &: E\{\cdot\} = 0.7 \cdot 1 + 0.2 \cdot 4 = 1.5, \\ u_2 = 1 &: E\{\cdot\} = 1 + 0.1 \cdot 1 + 0.2 \cdot 1 = 1.3, \\ u_2 = 2 &: E\{\cdot\} = 2 + 0.1 \cdot 4 + 0.7 \cdot 1 = 3.1 \end{aligned}$$

Hence we have, by selecting the minimizing u_2,

$$J_2(0) = 1.3, \qquad \mu_2^*(0) = 1.$$

For $x_2 = 1$, we have

$$\begin{aligned} J_2(1) &= \min_{u_2=0,1} \mathop{E}_{w_2}\{u_2 + (1 + u_2 - w_2)^2\} \\ &= \min_{u_2=0,1}\big[u_2 + 0.1(1 + u_2)^2 + 0.7(u_2)^2 + 0.2(u_2 - 1)^2\big]. \end{aligned}$$

$$u_2 = 0 : E\{\cdot\} = 0.1 \cdot 1 + 0.2 \cdot 1 = 0.3,$$
$$u_2 = 1 : E\{\cdot\} = 1 + 0.1 \cdot 4 + 0.7 \cdot 1 = 2.1.$$

Hence

$$J_2(1) = 0.3, \qquad \mu_2^*(1) = 0.$$

For $x_2 = 2$, the only admissible control is $u_2 = 0$, so we have

$$J_2(2) = \mathop{E}_{w_2}\{(2 - w_2)^2\} = 0.1 \cdot 4 + 0.7 \cdot 1 = 1.1,$$

$$J_2(2) = 1.1, \qquad \mu_2^*(2) = 0.$$

Period 1: Again we compute $J_1(x_1)$ for each of the three possible states $x_2 = 0, 1, 2$, using the values $J_2(0)$, $J_2(1)$, $J_2(2)$ obtained in the previous period. For $x_1 = 0$, we have

$$J_1(0) = \min_{u_1=0,1,2} \mathop{E}_{w_1}\{u_1 + (u_1 - w_1)^2 + J_2\big(\max(0, u_1 - w_1)\big)\},$$

$$u_1 = 0 : E\{\cdot\} = 0.1 \cdot J_2(0) + 0.7\big(1 + J_2(0)\big) + 0.2\big(4 + J_2(0)\big) = 2.8,$$
$$u_1 = 1 : E\{\cdot\} = 1 + 0.1\big(1 + J_2(1)\big) + 0.7 \cdot J_2(0) + 0.2\big(1 + J_2(0)\big) = 2.5,$$
$$u_1 = 2 : E\{\cdot\} = 2 + 0.1\big(4 + J_2(2)\big) + 0.7\big(1 + J_2(1)\big) + 0.2 \cdot J_2(0) = 3.68,$$

$$J_1(0) = 2.5, \qquad \mu_1^*(0) = 1.$$

For $x_1 = 1$, we have

$$J_1(1) = \min_{u_1=0,1} \mathop{E}_{w_1}\{u_1 + (1 + u_1 - w_1)^2 + J_2\big(\max(0, 1 + u_1 - w_1)\big)\},$$

$$u_1 = 0 : E\{\cdot\} = 0.1\big(1 + J_2(1)\big) + 0.7 \cdot J_2(0) + 0.2\big(1 + J_2(0)\big) = 1.2,$$
$$u_1 = 1 : E\{\cdot\} = 1 + 0.1\big(4 + J_2(2)\big) + 0.7\big(1 + J_2(1)\big) + 0.2 \cdot J_2(0) = 2.68,$$

$$J_1(1) = 1.2, \qquad \mu_1^*(1) = 0.$$

For $x_1 = 2$, the only admissible control is $u_1 = 0$, so we have

$$\begin{aligned} J_1(2) &= \mathop{E}_{w_1}\{(2 - w_1)^2 + J_2\big(\max(0, 2 - w_1)\big)\} \\ &= 0.1\big(4 + J_2(2)\big) + 0.7\big(1 + J_2(1)\big) + 0.2 \cdot J_2(0) \\ &= 1.68, \end{aligned}$$

$$J_1(2) = 1.68, \qquad \mu_1^*(2) = 0.$$

Period 0: Here we need to compute only $J_0(0)$ since the initial state is known to be 0. We have

$$J_0(0) = \min_{u_0=0,1,2} \mathop{E}_{w_0}\{u_0 + (u_0 - w_0)^2 + J_1\big(\max(0, u_0 - w_0)\big)\},$$

$$u_0 = 0 : E\{\cdot\} = 0.1 \cdot J_1(0) + 0.7\big(1 + J_1(0)\big) + 0.2\big(4 + J_1(0)\big) = 4.0,$$
$$u_0 = 1 : E\{\cdot\} = 1 + 0.1\big(1 + J_1(1)\big) + 0.7 \cdot J_1(0) + 0.2\big(1 + J_1(0)\big) = 3.67,$$
$$u_0 = 2 : E\{\cdot\} = 2 + 0.1\big(4 + J_1(2)\big) + 0.7\big(1 + J_1(1)\big) + 0.2 \cdot J_1(0) = 5.108,$$

$$J_0(0) = 3.67, \qquad \mu_0^*(0) = 1.$$

If the initial state were not known a priori, we would have to compute in a similar manner $J_0(1)$ and $J_0(2)$, as well as the minimizing u_0. The reader may verify (Exercise 1.2) that these calculations yield

$$J_0(1) = 2.67, \qquad \mu_0^*(1) = 0,$$

$$J_0(2) = 2.608, \qquad \mu_0^*(2) = 0.$$

Thus the optimal ordering policy for each period is to order one unit if the current stock is zero and order nothing otherwise. The results of the DP algorithm are given in tabular form in Fig. 1.3.1.

Example 3.3 (Optimizing a Chess Match Strategy)

Consider the chess match example considered in Section 1.1. There, a player can select timid play (probabilities p_d and $1 - p_d$ for a draw or loss, respectively) or bold play (probabilities p_w and $1 - p_w$ for a win or loss, respectively) in each game of the match. We want to formulate a DP algorithm for finding the policy that maximizes the player's probability of winning the match. Note that here we are dealing with a maximization problem. We can convert the problem to a minimization problem by changing the sign of the cost function, but a simpler alternative, which we will generally adopt, is to replace the minimization in the DP algorithm with maximization.

Let us consider the general case of an N-game match, and let the state be the *net score*, that is, the difference between the points of the player minus the points of the opponent (so a state of 0 corresponds to an even score). The optimal cost-to-go function at the start of the kth game is given by the dynamic programming recursion

$$\begin{aligned} J_k(x_k) = \max\big[&p_d J_{k+1}(x_k) + (1 - p_d) J_{k+1}(x_k - 1), \\ &p_w J_{k+1}(x_k + 1) + (1 - p_w) J_{k+1}(x_k - 1)\big]. \end{aligned} \tag{3.5}$$

The maximum above is taken over the two possible decisions:

(a) Timid play, which keeps the score at x_k with probability p_d, and changes x_k to $x_k - 1$ with probability $1 - p_d$.

(b) Bold play, which changes x_k to $x_k + 1$ or to $x_k - 1$ with probabilities p_w or $(1 - p_w)$, respectively.

It is optimal to play bold when

$$p_w J_{k+1}(x_k + 1) + (1 - p_w) J_{k+1}(x_k - 1) \geq p_d J_{k+1}(x_k) + (1 - p_d) J_{k+1}(x_k - 1)$$

or equivalently, if

$$\frac{p_w}{p_d} \geq \frac{J_{k+1}(x_k) - J_{k+1}(x_k - 1)}{J_{k+1}(x_k + 1) - J_{k+1}(x_k - 1)}. \tag{3.6}$$

The dynamic programming recursion is started with

$$J_N(x_N) = \begin{cases} 1 & \text{if } x_N > 0, \\ p_w & \text{if } x_N = 0, \\ 0 & \text{if } x_N < 0. \end{cases} \tag{3.7}$$

We have $J_N(0) = p_w$ because when the score is even after N games ($x_N = 0$), it is optimal to play bold in the first game of sudden death.

By executing the DP algorithm (3.5) starting with the terminal condition (3.7), and using the criterion (3.6) for optimality of bold play, we find the following, assuming that $p_d > p_w$:

$$\begin{aligned} J_{N-1}(x_{N-1}) &= 1 \text{ for } x_{N-1} > 1; \quad \text{optimal play: either} \\ J_{N-1}(1) &= \max[p_d + (1 - p_d)p_w,\ p_w + (1 - p_w)p_w] \\ &= p_d + (1 - p_d)p_w; \quad \text{optimal play: timid} \\ J_{N-1}(0) &= p_w; \quad \text{optimal play: bold} \\ J_{N-1}(-1) &= p_w^2; \quad \text{optimal play: bold} \\ J_{N-1}(x_{N-1}) &= 0 \text{ for } x_{N-1} < -1; \quad \text{optimal play: either.} \end{aligned}$$

Also, given $J_{N-1}(x_{N-1})$, and Eqs. (3.5) and (3.6) we obtain

$$\begin{aligned} J_{N-2}(0) &= \max\left[p_d p_w + (1 - p_d)p_w^2,\ p_w\left(p_d + (1 - p_d)p_w\right) + (1 - p_w)p_w^2\right] \\ &= p_w\left(p_w + (p_w + p_d)(1 - p_w)\right) \end{aligned}$$

and that if the score is even with 2 games remaining, it is optimal to play bold. Thus for a 2-game match, the optimal policy for both periods is to play timid if and only if the player is ahead in the score. The region of pairs (p_w, p_d) for which the player has a better than 50-50 chance to win a 2-game match is

$$R_2 = \left\{(p_w, p_d) \mid J_0(0) = p_w\left(p_w + (p_w + p_d)(1 - p_w)\right) > 1/2\right\},$$

and, as noted in the preceding section, it includes points where $p_w < 1/2$.

Example 3.4 (Finite-State Systems)

We mentioned earlier (cf. the examples in Section 1.1) that systems with a finite number of states can be represented either in terms of a discrete-time system equation or in terms of the probabilities of transition between the states. Let us work out the DP algorithm corresponding to the latter case. We will assume for the sake of the following discussion that the problem is stationary (i.e., the transition probabilities, the cost per stage, and the control constraint sets do not change from one stage to the next). Then, if

$$p_{ij}(u) = P\{x_{k+1} = j \mid x_k = i, u_k = u\}$$

are the transition probabilities, we can alternatively represent the system by the system equation (cf. the discussion of the previous section)

$$x_{k+1} = w_k,$$

where the probability distribution of the disturbance w_k is

$$P\{w_k = j \mid x_k = i, u_k = u\} = p_{ij}(u).$$

Using this system equation and denoting by $g(i, u)$ the expected cost per stage at state i when control u is applied, the DP algorithm can be rewritten as

$$J_k(i) = \min_{u \in U(i)} \left[g(i, u) + E\left\{J_{k+1}(w_k)\right\}\right]$$

or equivalently (in view of the distribution of w_k given previously)

$$J_k(i) = \min_{u \in U(i)} \left[g(i, u) + \sum_j p_{ij}(u) J_{k+1}(j)\right].$$

As an illustration, in the machine replacement example of Section 1.1, this algorithm takes the form

$$J_N(i) = 0, \qquad i = 1, \ldots, n,$$

$$J_k(i) = \min\left[R + g(1) + J_{k+1}(1),\ g(i) + \sum_{j=i}^{n} p_{ij} J_{k+1}(j)\right].$$

The two expressions in the above minimization correspond to the two available decisions (replace or not replace the machine).

In the queueing example of Section 1.1, the DP algorithm takes the form

$$J_N(i) = R(i), \qquad i = 0, 1, \ldots, n,$$

$$J_k(i) = \min\left[r(i) + c_f + \sum_{j=0}^{n} p_{ij}(u_f) J_{k+1}(j),\ r(i) + c_s + \sum_{j=0}^{n} p_{ij}(u_s) J_{k+1}(j)\right].$$

The two expressions in the above minimization correspond to the two possible decisions (fast and slow service).

1.4 STATE AUGMENTATION

We now discuss how to deal with situations where some of the assumptions of the basic problem are violated. Generally, in such cases the problem can be reformulated into the basic problem format. This process is called *state augmentation* because it typically involves the enlargement of the state space. The general guideline in state augmentation is to *include in the enlarged state at time k all the information that is known to the controller at time k and can be used with advantage in selecting u_k*. Unfortunately, state augmentation often comes at a price: the reformulated problem may have very complex state and/or control spaces. We provide some examples.

Time Lags

In many applications the system state x_{k+1} depends not only on the preceding state x_k and control u_k but also on earlier states and controls. In other words, states and controls influence future states with some time lag. Such situations can be handled by state augmentation; the state is expanded to include an appropriate number of earlier states and controls.

For simplicity, assume that there is at most a single period time lag in the state and control; that is, the system equation has the form

$$x_{k+1} = f_k(x_k, x_{k-1}, u_k, u_{k-1}, w_k), \qquad k = 1, 2, \ldots, N-1, \tag{4.1}$$

$$x_1 = f_0(x_0, u_0, w_0).$$

Time lags of more than one period can be handled similarly.

If we introduce additional state variables y_k and s_k, and we make the identifications $y_k = x_{k-1}$, $s_k = u_{k-1}$, the system equation (4.1) yields

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \\ s_{k+1} \end{pmatrix} = \begin{pmatrix} f_k(x_k, y_k, u_k, s_k, w_k) \\ x_k \\ u_k \end{pmatrix}. \tag{4.2}$$

By defining $\tilde{x}_k = (x_k, y_k, s_k)$ as the new state, we have

$$\tilde{x}_{k+1} = \tilde{f}_k(\tilde{x}_k, u_k, w_k),$$

where the system function $\tilde{f}_k$ is defined from Eq. (4.2). By using the preceding equation as the system equation and by expressing the cost function in terms of the new state, the problem is reduced to the basic problem without time lags. Naturally, the control u_k should now depend on the new state $\tilde{x}_k$, or equivalently a policy should consist of functions μ_k of the

current state x_k, as well as the preceding state x_{k-1} and the preceding control u_{k-1}.

When the DP algorithm for the reformulated problem is translated in terms of the variables of the original problem, it takes the form

$$J_N(x_N) = g_N(x_N),$$

$$\begin{aligned} &J_{N-1}(x_{N-1}, x_{N-2}, u_{N-2}) \\ &\quad = \min_{u_{N-1} \in U_{N-1}(x_{N-1})} \mathop{E}_{w_{N-1}} \Big\{ g_{N-1}(x_{N-1}, u_{N-1}, w_{N-1}) \\ &\qquad + J_N\big(f_{N-1}(x_{N-1}, x_{N-2}, u_{N-1}, u_{N-2}, w_{N-1})\big) \Big\}, \end{aligned}$$

$$\begin{aligned} J_k(x_k, x_{k-1}, u_{k-1}) = \min_{u_k \in U_k(x_k)} \mathop{E}_{w_k} \Big\{ & g_k(x_k, u_k, w_k) \\ + J_{k+1}\big(f_k(x_k, x_{k-1}, u_k, u_{k-1}, w_k), x_k, u_k\big) \Big\}, \quad & k = 1, \ldots, N-2, \end{aligned}$$

$$J_0(x_0) = \min_{u_0 \in U_0(x_0)} \mathop{E}_{w_0} \big\{ g_0(x_0, u_0, w_0) + J_1\big(f_0(x_0, u_0, w_0), x_0, u_0\big) \big\}.$$

Similar reformulations are possible when time lags appear in the cost; for example, in the case where the cost is of the form

$$E\left\{ g_N(x_N, x_{N-1}) + g_0(x_0, u_0, w_0) + \sum_{k=1}^{N-1} g_k(x_k, x_{k-1}, u_k, w_k) \right\}.$$

The extreme case of time lags in the cost arises in the nonadditive form

$$E\big\{ g_N(x_N, x_{N-1}, \ldots, x_0, u_{N-1}, \ldots, u_0, w_{N-1}, \ldots, w_0) \big\}.$$

Then, the problem can be reduced to the basic problem format, by taking as augmented state

$$\tilde{x}_k = \big(x_k, x_{k-1}, \ldots, x_0, u_{k-1}, \ldots, u_0, w_{k-1}, \ldots, w_0\big)$$

and $E\{g_N(\tilde{x}_N)\}$ as reformulated cost. Policies consist of functions μ_k of the present and past states $x_k, \ldots, x_0$, the past controls $u_{k-1}, \ldots, u_0$, and the past disturbances $w_{k-1}, \ldots, w_0$. Naturally, we must assume that the past disturbances are known to the controller. Otherwise, we are faced with a problem where the state is imprecisely known to the controller. Such problems are known as problems with imperfect state information and will be discussed in Chapter 5.

Correlated Disturbances

We turn now to the case where the disturbances w_k are correlated over time. A common situation that can be handled efficiently by state augmentation arises when the process $w_0, \ldots, w_{N-1}$ can be represented as the output of a linear system driven by independent random variables. As an example, suppose that by using statistical methods, we determine that the evolution of w_k can be modeled by an equation of the form

$$w_k = \lambda w_{k-1} + \xi_k,$$

where λ is a given scalar and $\{\xi_k\}$ is a sequence of independent random vectors with given distribution. Then we can introduce an additional state variable

$$y_k = w_{k-1}$$

and obtain a new system equation

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} f_k(x_k, u_k, \lambda y_k + \xi_k) \\ \lambda y_k + \xi_k \end{pmatrix},$$

where the new state is the pair $\tilde{x}_k = (x_k, y_k)$ and the new disturbance is the vector ξ_k.

More generally, suppose that w_k can be modeled by

$$w_k = C_k y_{k+1},$$

where

$$y_{k+1} = A_k y_k + \xi_k, \qquad k = 0, \ldots, N-1,$$

A_k, C_k are known matrices of appropriate dimension, and ξ_k are independent random vectors with given distribution (see Fig. 1.4.1). By viewing y_k as an additional state variable, we obtain the new system equation

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} f_k\big(x_k, u_k, C_k(A_k y_k + \xi_k)\big) \\ A_k y_k + \xi_k \end{pmatrix}.$$

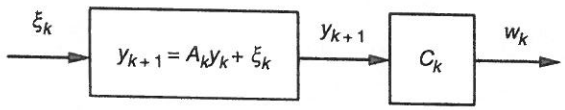


Figure 1.4.1 Representing correlated disturbances as the output of a linear system driven by independent random vectors.

Note that in order to have perfect state information, the controller must be able to observe y_k. Unfortunately, this is true only in the minority of practical cases; for example when C_k is the identity matrix and w_{k-1} is observed before u_k is applied. In the case of perfect state information, the DP algorithm takes the form

$$J_N(x_N, y_N) = g_N(x_N),$$

$$J_k(x_k, y_k) = \min_{u_k \in U_k(x_k)} E_{\xi_k}\big\{g_k\big(x_k, u_k, C_k(A_k y_k + \xi_k)\big) + J_{k+1}\big(f_k\big(x_k, u_k, C_k(A_k y_k + \xi_k)\big), A_k y_k + \xi_k\big)\big\}.$$

Forecasts

Finally, consider the case where at time k the controller has access to a forecast y_k that results in a reassessment of the probability distribution of w_k and possibly of future disturbances. For example, y_k may be an exact prediction of w_k or an exact prediction that the probability distribution of w_k is a specific one out of a finite collection of distributions. Forecasts of interest in practice are, for example, probabilistic predictions on the state of the weather, the interest rate for money, and the demand for inventory.

Generally, forecasts can be handled by state augmentation although the reformulation into the basic problem format may be quite complex. We will treat here only a simple special case.

Assume that at the beginning of each period k, the controller receives an accurate prediction that the next disturbance w_k will be selected according to a particular probability distribution out of a given collection of distributions $\{Q_1, \ldots, Q_m\}$; that is, if the forecast is i, then w_k is selected according to Q_i. The a priori probability that the forecast will be i is denoted by p_i and is given.

As an example, suppose that in our earlier inventory example the demand w_k is determined according to one of three distributions Q_1, Q_2, and Q_3, corresponding to "small," "medium," and "large" demand. Each of the three types of demand occurs with a given probability at each time period, independently of the values of demand at previous time periods. However, the inventory manager, prior to ordering u_k, gets to know through a forecast the type of demand that will occur. (Note that it is the probability distribution of demand that becomes known through the forecast, not the demand itself.)

The forecasting process can be represented by means of the equation

$$y_{k+1} = \xi_k,$$

where y_{k+1} can take the values $1, \ldots, m$, corresponding to the m possible forecasts, and ξ_k is a random variable taking the value i with probability

p_i. The interpretation here is that when ξ_k takes the value i, then w_{k+1} will occur according to the distribution Q_i.

By combining the system equation with the forecast equation $y_{k+1} = \xi_k$, we obtain an augmented system given by

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} f_k(x_k, u_k, w_k) \\ \xi_k \end{pmatrix}.$$

The new state is $\tilde{x}_k = (x_k, y_k)$, and because the forecast y_k is known at time k, perfect state information prevails. The new disturbance is $\tilde{w}_k = (w_k, \xi_k)$, and its probability distribution is determined by the distributions Q_i and the probabilities p_i, and depends explicitly on $\tilde{x}_k$ (via y_k) but not on the prior disturbances. Thus, by suitable reformulation of the cost, the problem can be cast into the basic problem format. Note that the control applied depends on both the current state and the current forecast.

The DP algorithm takes the form

$$J_N(x_N, y_N) = g_N(x_N), \tag{4.3}$$

$$J_k(x_k, y_k) = \min_{u_k \in U_k(x_k)} \underset{w_k}{E} \Big\{ g_k(x_k, u_k, w_k) + \sum_{i=1}^{m} p_i J_{k+1}\big(f_k(x_k, u_k, w_k), i\big) \,\Big|\, y_k \Big\}, \tag{4.4}$$

where y_k may take the values $1, \ldots, m$, and the expectation over w_k is taken with respect to the distribution Q_{y_k}.

There is a nice simplification of the above algorithm that allows DP to be executed over a smaller space. In particular, define

$$\hat{J}_k(x_k) = \sum_{i=1}^{m} p_i J_k(x_k, i), \qquad k = 0, 1, \ldots, N-1,$$

and

$$\hat{J}_N(x_N) = g_N(x_N).$$

Then from Eq. (4.4), we obtain the algorithm

$$\hat{J}_k(x_k) = \sum_{i=1}^{m} p_i \min_{u_k \in U_k(x_k)} \underset{w_k}{E} \Big\{ g_k(x_k, u_k, w_k) + \hat{J}_{k+1}\big(f_k(x_k, u_k, w_k)\big) \,\Big|\, y_k = i \Big\},$$

which is executed over the space of x_k rather than x_k *and* y_k. This simplification arises also in other contexts where the state has a component that cannot be affected by the choice of control (see Exercise 1.22).

It should be clear that the preceding formulation admits several extensions; one example is the case where forecasts can be influenced by the control action and involve several future disturbances. However, the price for these extensions is increased complexity of the corresponding DP algorithm.

1.5 SOME MATHEMATICAL ISSUES

Let us now discuss some technical issues relating to the basic problem formulation. The reader who is not mathematically inclined need not be concerned about these issues and can skip this section without loss of continuity.

Once an admissible policy $\{\mu_0, \ldots, \mu_{N-1}\}$ is adopted, the following sequence of events is envisioned at the typical stage k:

1. The controller observes x_k and applies $u_k = \mu_k(x_k)$.
2. The disturbance w_k is generated according to the given distribution $P_k(\cdot \mid x_k, \mu_k(x_k))$.
3. The cost $g_k\big(x_k, \mu_k(x_k), w_k\big)$ is incurred and added to previous costs.
4. The next state x_{k+1} is generated according to the system equation

$$x_{k+1} = f_k\big(x_k, \mu_k(x_k), w_k\big).$$

If this is the last stage ($k = N-1$), the terminal cost $g_N(x_N)$ is added to previous costs and the process terminates. Otherwise, k is incremented, and the same sequence of events is repeated for the next stage.

For each stage, the above process is well-defined and is couched in precise probabilistic terms. Matters are, however, complicated by the need to view the cost as a well-defined random variable with well-defined expected value. The framework of probability theory requires that for each policy we define an underlying probability space, that is, a set Ω, a collection of events in Ω, and a probability measure on these events. In addition, the cost must be a well-defined random variable on this space in the sense of Appendix C (a measurable function from the probability space into the real line in the terminology of measure-theoretic probability theory). For this to be true, additional (measurability) assumptions on the functions f_k, g_k, and μ_k may be required, and it may be necessary to introduce additional structure on the spaces S_k, C_k, and D_k. Furthermore, these assumptions may restrict the class of admissible policies, since the functions μ_k may be constrained to satisfy additional (measurability) requirements.

Thus, unless these additional assumptions and structure are specified, the basic problem is formulated inadequately. Unfortunately, a rigorous formulation for general state, control, and disturbance spaces is well beyond the mathematical framework of this introductory book and will not be undertaken here. Nonetheless, it turns out that these difficulties are mainly technical and do not substantially affect the basic results to be obtained. For this reason, we find it convenient to proceed with informal derivations and arguments; this is consistent with most of the literature on the subject.