4. Optimization Methods

4 Optimization Methods

Optimization methods are designed to provide the 'best' values of system design and operating policy variables – values that will lead to the highest levels of system performance. These methods, combined with more detailed and accurate simulation methods, are the primary ways we have, short of actually building physical models, of estimating the likely impacts of particular water resources system designs and operating policies. This chapter introduces and illustrates some of the art of optimization model development and use in analysing water resources systems. The modelling methods introduced in this chapter are extended in subsequent chapters.

1. Introduction

All of us are optimizers. We all make decisions that maximize our welfare in some way or another. Often the welfare we are maximizing may come later in life. By optimizing, it reflects our evaluation of future benefits versus current costs or benefits forgone. In economics, the extent to which we value future benefits today is reflected by what is called a *discount rate*. While economic criteria are only a part of everything we consider when making decisions, they are often among those deemed very important. Economic evaluation methods involving discount rates can be used to consider and compare alternatives characterized by various benefits and costs that occur over time. This chapter begins with a quick and basic review of what is often called *applied engineering economics*. Many of the optimization methods used in practice incorporate concepts from engineering economics.

Engineering economic methods typically identify a set of mutually exclusive alternatives (only one alternative can be selected) and then, using various methods involving the discount rate, identify the best one. The values of the decision-variables (e.g., the design and operating policy variables) are known for each alternative. For example, consider again the tank design problem presented in the previous chapter. Alternative tank designs could be identified, and then each could be evaluated, on the basis of cost and perhaps other criteria as well. The best would be called the optimal one, at least with respect to the objective criteria used.

The optimization methods introduced in this chapter extend those engineering economics methods. Some are discrete, some are continuous. Continuous optimization methods can identify the 'best' tank design, for example, without having to identify numerous discrete, mutually exclusive alternatives. Just how that can be done will be discussed later in this chapter.

Before proceeding to a more detailed discussion of optimization, a review of some methods of dealing with time streams of economic incomes or costs (engineering economics) may be useful.

2. Comparing Time Streams of Economic Benefits and Costs

Alternative plans, p, may involve different benefits and costs over time. These different plans need to be compared somehow. Let the net benefit generated in time period t by plan p be designated simply as B_t^p. Each plan is characterized by the time stream of net benefits it generates over its planning period T_p.

$$\{B_1^p, B_2^p, B_3^p, \ldots, B_{T_p}^p\} \tag{4.1}$$

Clearly, if in any time period t the benefits exceed the costs, then $B_t^p > 0$; and if the costs exceed the benefits, $B_t^p < 0$. This section defines two ways of comparing different benefit, cost or net-benefit time streams produced by different plans perhaps having planning periods ending in different years Tp.

2.1. Interest Rates

Fundamental to the conversion of a time series of incomes and costs to a single value is the concept of the time value of money. From time to time, individuals, private corporations and governments need to borrow money to do what they want to do. The amount paid back to the lender has two components: (1) the amount borrowed and (2) an additional amount called *interest*. The interest amount is the cost of borrowing money, of having the money when it is loaned compared to when it is paid back. In the private sector the interest rate, the added fraction of the amount loaned that equals the interest, is often identified as the *marginal rate of return on capital*. Those who have money, called *capital*, can either use it themselves or they can lend it to others, including banks, and receive interest. Assuming people with capital invest their money where it yields the largest amount of interest, consistent with the risk they are willing to take, most investors should be receiving at least the prevailing interest rate as the return on their capital.

The interest rate includes a number of considerations. One is the *time value of money* (a willingness to pay something to obtain money now rather than to obtain the same amount later). Another is the *risk of losing capital* (not getting the full amount of a loan or investment returned at some future time). A third is the *risk of reduced purchasing capability* (the expected inflation over time). The greater the risks of losing capital or purchasing power, the higher the interest rate will be compared to the rate reflecting only the time value of money in a secure and inflation-free environment.

2.2. Equivalent Present Value

To compare projects or plans involving different time series of benefits and costs, it is often convenient to express these time series as a single equivalent value. One way to do this is to convert the time series to what it is worth today, its *present worth*, that is, a single value at the present time. This present worth will depend on the prevailing interest rate in each future time period. Assuming an amount V_0 is invested at the beginning of a time period, e.g., a year, in a project or a savings account earning interest at a rate r per period, then at the end of the period the value of that investment is $(1 + r)V_0$.

If one invests an amount V_0 at the beginning of period $t = 1$ and at the end of that period immediately reinvests the total amount (the original investment plus interest earned), and continues to do this for n consecutive periods at the same period interest rate r, the value, V_n, of that investment at the end of n periods would be:

$$V_n = V_0(1 + r)^n \tag{4.2}$$

The initial amount V_0 is said to be equivalent to V_n at the end of n periods. Thus the present worth or present value, V_0, of an amount of money V_n at the end of period n is:

$$V_0 = V_n/(1 + r)^n \tag{4.3}$$

Equation 4.3 is the basic compound interest discounting relation needed to determine the present value at the beginning of period 1 of net benefits V_n that accrue at the end of n time periods.

The total present value of the net benefits generated by plan p, denoted V_0^p, is the sum of the values of the net benefits V_t^p accrued at the end of each time period t times the discount factor for that period t. Assuming the interest or discount rate r in the discount factor applies for the duration of the planning period Tp,

$$V_0^p = \sum_t^{Tp} V_t^p/(1+r)^t \tag{4.4}$$

The present value of the net benefits achieved by two or more plans having the same economic planning horizons Tp can be used as an economic basis for plan selection. If the economic lives or planning horizons of projects differ, then the present value of the plans may not be an appropriate measure for comparison and plan selection. A valid comparison of alternative plans is possible if all plans cover the same planning period.

2.3. Equivalent Annual Value

If the lives of various plans differ, but the same plans will be repeated on into the future, then one need only compare the equivalent constant annual net benefits of each plan. Finding the average or equivalent annual amount V^p is done in two steps. First, one can compute the present value, V_0^p, of the time stream of net benefits, using Equation 4.4. The equivalent constant annual benefits, V^p, all discounted to the present must equal the present value, V_0^p.

$$V_0^p = \sum_t^{Tp} V_t^p/(1+r)^t = \sum_t^{Tp} V^p/(1+r)^t \qquad (4.5)$$

With a little algebra the average annual end-of-year benefits V^p of the project or plan p is:

$$V^p = V_0^p[r(1+r)^{Tp}]/[(1+r)^{Tp} - 1] \qquad (4.6)$$

The capital recovery factor CRF_n is the expression $[r(1+r)^{Tp}]/[(1+r)^{Tp} - 1]$ in Equation 4.6 that converts a fixed payment or present value V_0^p at the beginning of n time periods into an equivalent fixed periodic payment V^p at the end of each period. If the interest rate per period is r and there are n periods involved, then the capital recovery factor is:

$$CRF_n = [r(1+r)^n]/[(1+r)^n - 1] \qquad (4.7)$$

This factor is often used to compute the equivalent annual end-of-year cost of engineering structures that have a fixed initial construction cost C_0 and annual end-of-year operation, maintenance, and repair (OMR) costs. The equivalent uniform end-of-year total annual cost, TAC, equals the initial cost times the capital recovery factor plus the equivalent annual end-of-year uniform OMR costs.

$$TAC = CRF_n C_0 + OMR \qquad (4.8)$$

For private investments requiring borrowed capital, interest rates are usually established, and hence fixed, at the time of borrowing. However, benefits may be affected by changing interest rates, which are not easily predicted. It is common practice in benefit–cost analyses to assume constant interest rates over time, for lack of any better assumption.

Interest rates available to private investors or borrowers may not be the same rates that are used for analysing public investment decisions. In an economic evaluation of public-sector investments, the same relationships are used even though government agencies are not generally free to loan or borrow funds on private money markets. In the case of public-sector investments, the interest rate to be used in an economic analysis is a matter of public policy; it is the rate at which the government is willing to forego current benefits to its citizens in order to provide benefits to those living in future time periods. It can be viewed as the government's estimate of the time value of public monies or the marginal rate of return to be achieved by public investments.

These definitions and concepts of engineering economics are applicable to many of the problems faced in water resources planning and management. More detailed discussions of the application of engineering economics are contained in numerous texts on the subject.

3. Non-Linear Optimization Models and Solution Procedures

Constrained optimization is also called *mathematical programming*. Mathematical programming techniques include Lagrange multipliers, linear and non-linear programming, dynamic programming, quadratic programming, fractional programming and geometric programming, to mention a few. The applicability of each of these as well as other constrained optimization procedures is highly dependent on the mathematical structure of the model. The remainder of this chapter introduces and illustrates the application of some of the most commonly used constrained optimization techniques in water resources planning and management. These include classical constrained optimization using calculus-based Lagrange multipliers, discrete dynamic programming, and linear and non-linear programming.

Consider a river from which diversions are made to three water-consuming firms that belong to the same corporation, as illustrated in Figure 4.1. Each firm makes a product. Water is needed in the process of making that product, and is the critical resource. The three firms can be denoted by the index $j = 1$, 2 and 3 and their water allocations by x_j. Assume the problem is to determine the allocations x_j of water to each of three firms ($j = 1, 2, 3$) that maximize the total net benefits, $\sum_j NB_j(x_j)$, obtained from all three firms. The total amount of water available is constrained or limited to a quantity of Q.

Assume the net benefits, $NB_j(x_j)$, derived from water x_j allocated to each firm j, are defined by:

$$NB_1(x_1) = 6x_1 - x_1^2 \qquad (4.9)$$

$$NB_2(x_2) = 7x_2 - 1.5x_2^2 \qquad (4.10)$$

$$NB_3(x_3) = 8x_3 - 0.5x_3^2 \qquad (4.11)$$

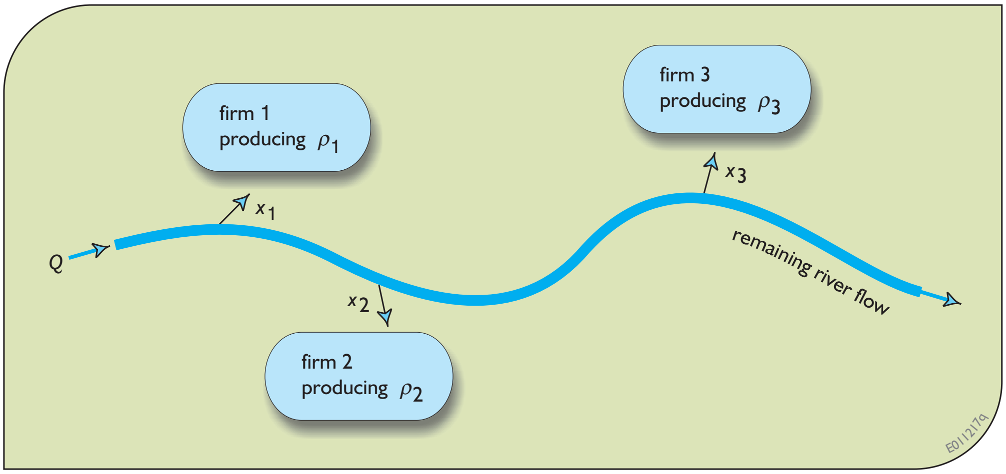


Figure 4.1. Three water-using firms obtain water from river diversions. The amounts allocated, x_j, to each firm j will depend on the amount of water available, Q, in the river.

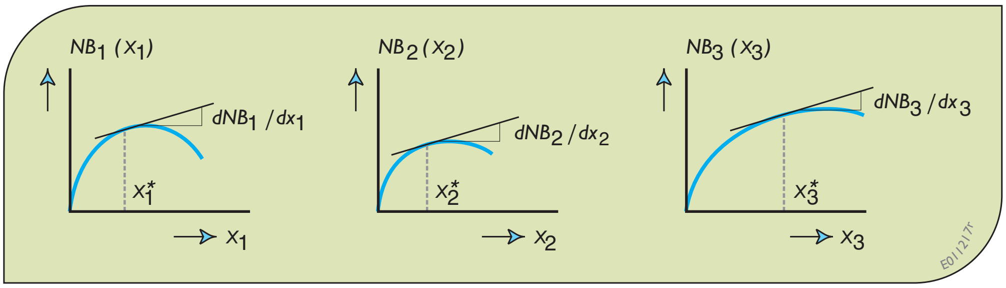


Figure 4.2. Concave net benefit functions and their slopes at allocations x_1^*, x_2^* and x_3^*.

These are concave functions exhibiting decreasing marginal net benefits with increasing allocations. These functions look like hills, as illustrated in Figure 4.2.

3.1. Solution Using Calculus

Calculus can be used to find the allocation to each firm that maximizes its own net benefit, simply by finding where the slope or derivative of the net benefit function for the firm equals zero. The derivative, $\mathrm{d}NB(x_1)/\mathrm{d}x_1$, of the net benefit function for Firm 1 is $(6 - 2x_1)$ and hence the best allocation to Firm 1 would be 6/2 or 3. The best allocations for Firms 2 and 3 are 2.33 and 8 respectively. The total amount of water desired by all firms is the sum of each firm's desired allocation, or 13.33 flow units. However, suppose only 6 units of flow are available for all three firms. Introducing this constraint renders the previous solution infeasible. In this case we want to find the allocations that maximize the total net benefit obtained from all firms subject to having only 6 flow units to allocate. Using simple calculus will not suffice.

3.2. Solution Using Hill Climbing

One approach to finding the particular allocations that maximize the total net benefit derived from all firms in this example is an incremental steepest-hill-climbing method. This method divides the total available flow Q into increments and allocates each additional increment so as to get the maximum additional net benefit from that incremental amount of water. This procedure works in this example because the functions are concave; in other words, the marginal benefits decrease as the allocation increases. This procedure is illustrated by the flow diagram in Figure 4.3.

Table 4.1 lists the results of applying the procedure shown in Figure 4.3 to the problem of a) allocating 8 and b) allocating 20 flow units to the three firms and the river. Here a river flow of at least 2 is required and is to be satisfied, when possible, before any allocations are made to the firms.

The hill-climbing method illustrated in Figure 4.3 and Table 4.1 assigns each incremental flow ΔQ to the use that yields the largest additional (marginal) net benefit. An allocation is optimal for any total flow Q when the

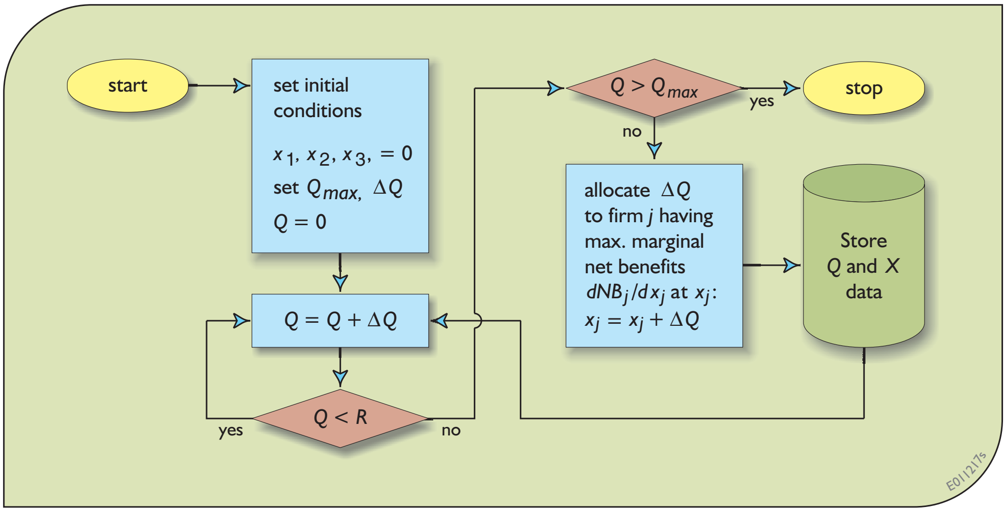


Figure 4.3. Steepest hill-climbing approach for finding allocation of a flow *Qmax* to the three firms, while meeting minimum river flow requirements *R*.

Table 4.1. Hill-climbing iterations for finding allocations that maximize total net benefit given a flow of *Qmax* and a required (minimum) streamflow of *R* = 2.

E020820g

$Q_{max} = 8$; $Q_i = 0$; $\Delta Q = 1$; river flow $R \geq \min\{Q, 2\}$

iteration i	Q_i	allocations. R, x_j: R	x_1	x_2	x_3	marginal net benefits: $6\text{-}2x_1$	$7\text{-}3x_2$	$8\text{-}x_3$	$Q_i + \Delta Q$	new allocations: R	x_1	x_2	x_3	total net benefits $\Sigma_j NB_a(x_1)_j$
1-3	0-2	0-2	0	0	0	6	7	8	3	2	0	0	1	7.5
4	3	2	0	0	1	6	7	7	4	2	0	0	2	14.0
5	4	2	0	0	2	6	7	6	5	2	0	1	2	19.5
6	5	2	0	1	2	6	4	6	6	2	0	1	3	25.0
7	6	2	0	1	3	6	4	5	7	2	1	1	3	30.0
8	7	2	1	1	3	4	4	5	8	2	1	1	4	34.5
9	8	2	1	1	4	4	4	4	–	–	–	–	–	---

$Q_{max} = 20$; $\Delta Q \longrightarrow 0$; river flow $R \geq \min\{Q, 2\}$ selected values of Q

Q	allocations. R, x_j: R	x_1	x_2	x_3	marginal net benefits: $6\text{-}2x_1$	$7\text{-}3x_2$	$8\text{-}x_3$	total net benefits $\Sigma_j NB_j(x_1)$
0-2	0-2	0	0	0	6	7	8	0.0
4	2	0	0.25	1.75	6.00	6.25	6.25	14.1
5	2	0.18	0.46	2.36	5.64	5.64	5.64	20.0
8	2	1.00	1.00	4.00	4.00	4.00	4.00	34.5
10	2	1.55	1.36	5.09	2.91	2.91	2.91	41.4
15	2	2.91	2.27	7.82	0.18	0.18	0.18	49.1
20	6.67	3.00	2.33	8.00	0	0	0	49.2

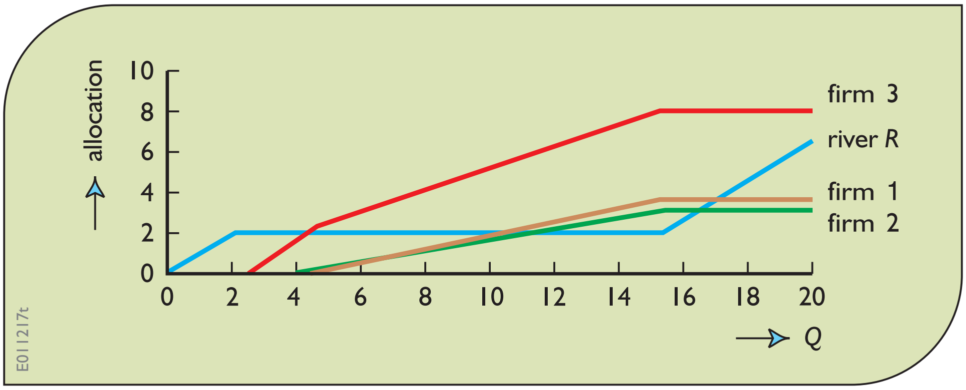


Figure 4.4. Water allocation policy that maximizes total net benefits derived from all three water-using firms.

marginal net benefits from each non-zero allocation are equal, or as close to each other as possible given the size of the increment ΔQ. In this example, with a ΔQ of 1 and Qmax of 8, it just happens that the marginal net benefits are all equal (to 4). The smaller the ΔQ, the more optimal will be the allocations in each iteration, as shown in the lower portion of Table 4.1 where ΔQ approaches 0.

Based on the allocations derived for various values of available water Q, as shown in Table 4.1, an allocation policy can be defined. For this problem the allocation policy that maximizes total net benefits is shown in Figure 4.4.

This hill-climbing approach leads to optimal allocations only if all of the net benefit functions whose sum is being maximized are concave: that is, the marginal net benefits decrease as the allocation increases. Otherwise, only a local optimum solution can be guaranteed. This is true using any calculus-based optimization procedure or algorithm.

3.3. Solution Using Lagrange Multipliers

3.3.1. Approach

As an alternative to hill-climbing methods, consider a calculus-based method involving Lagrange multipliers. To illustrate this approach, a slightly more complex water-allocation example will be used. Assume that the benefit, $B_j(x_j)$, each water-using firm receives is determined, in part, by the quantity of product it produces and the price per unit of the product that is charged. As before, these products require water and water is the limiting resource. The amount of product produced, p_j, by each firm j is dependent on the amount of water, x_j, allocated to it.

Let the function $P_j(x_j)$ represent the maximum amount of product, p_j, that can be produced by firm j from an allocation of water x_j. These are called production functions. They are typically concave: as x_j increases the slope, $dP_j(x_j)/dx_j$, of the production function, $P_j(x_j)$, decreases.

For this example assume the production functions for the three water-using firms are:

$$p_1 = 0.4(x_1)^{0.9} \tag{4.12}$$

$$p_2 = 0.5(x_2)^{0.8} \tag{4.13}$$

$$p_3 = 0.6(x_3)^{0.7} \tag{4.14}$$

Next consider the cost of production. Assume the associated cost of production can be expressed by the following convex functions:

$$C_1 = 3(p_1)^{1.3} \tag{4.15}$$

$$C_2 = 5(p_2)^{1.2} \tag{4.16}$$

$$C_3 = 6(p_3)^{1.15} \tag{4.17}$$

Each firm produces a unique patented product, and hence it can set and control the unit price of its product. The lower the unit price, the greater the demand and thus the more each firm can sell. Each firm has determined the relationship between the amount that can be sold and the unit price – that is, the demand functions for that product. These unit price or demand functions are shown in Figure 4.5 where the p_j's are the amounts of each product produced. The vertical axis of each graph is the unit price. To simplify the problem we are assuming linear demand functions, but this assumption is not a necessary condition.

The optimization problem is to find the water allocations, the production levels and the unit prices that together maximize the total net benefit obtained from all

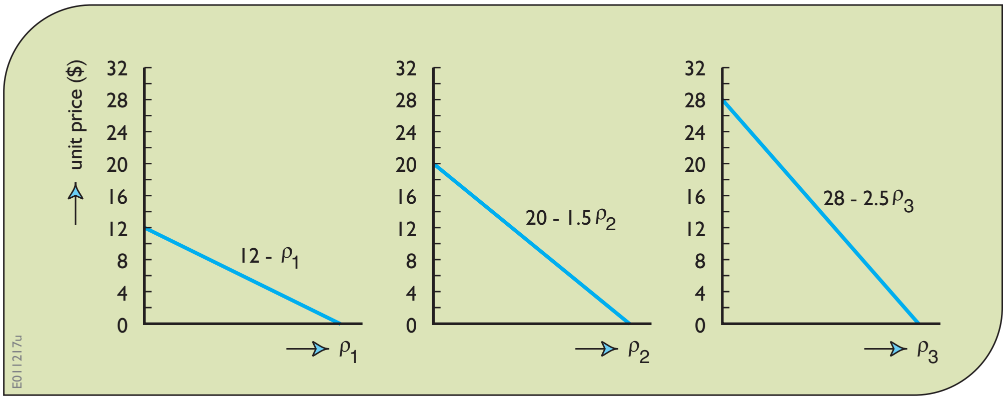


Figure 4.5. Unit prices that will guarantee the sale of the specified amounts of products p_j produced in each of the three firms.

three firms. The water allocations plus the amount that must remain in the river, R, cannot exceed the total amount of water Q available.

Constructing and solving a model of this problem for various values of Q, the total amount of water available, will define the three allocation policies as functions of Q. These policies can be displayed as a graph, as in Figure 4.4, showing the three best allocations given any value of Q. This of course assumes the firms can adjust to varying values of Q. In reality this may not be the case. (Chapter 10 examines this problem using more realistic benefit functions that reflect the degree to which firms can adapt to changing inputs over time.)

The model:

$$\text{Maximize Net\_benefit} \tag{4.18}$$

Subject to:

Definitional constraints:

$$\text{Net\_benefit} = \text{Total\_return} - \text{Total\_cost} \tag{4.19}$$

$$\text{Total\_return} = (12 - p_1)p_1 + (20 - 1.5p_2)p_2 + (28 - 2.5p_3)p_3 \tag{4.20}$$

$$\text{Total\_cost} = 3(p_1)^{1.30} + 5(p_2)^{1.20} + 6(p_3)^{1.15} \tag{4.21}$$

Production functions defining the relationship between water allocations x_j and production p_j:

$$p_1 \le 0.4(x_1)^{0.9} \tag{4.22}$$

$$p_2 \le 0.5(x_2)^{0.8} \tag{4.23}$$

$$p_3 \le 0.6(x_3)^{0.7} \tag{4.24}$$

Water allocation restriction:

$$R + x_1 + x_2 + x_3 \le Q \tag{4.25}$$

One can first solve this model for the values of each p_j that maximize the total net benefits, assuming water is not a limiting constraint. This is equivalent to finding each individual firm's maximum net benefits, assuming all the water that is needed is available. Using calculus we can equate the derivatives of the total net benefit function with respect to each p_j to 0 and solve each of the resulting three independent equations:

$$\text{Total Net\_benefit} = [(12 - p_1)p_1 + (20 - 1.5p_2)p_2 + (28 - 2.5p_3)p_3] - [3(p_1)^{1.30} + 5(p_2)^{1.20} + 6(p_3)^{1.15}] \tag{4.26}$$

Derivatives:

$$\partial(\text{Net\_benefit})/\partial p_1 = 0 = 12 - 2p_1 - 1.3(3)p_1^{0.3} \tag{4.27}$$

$$\partial(\text{Net\_benefit})/\partial p_2 = 0 = 20 - 3p_2 - 1.2(5)p_2^{0.2} \tag{4.28}$$

$$\partial(\text{Net\_benefit})/\partial p_3 = 0 = 28 - 5p_3 - 1.15(6)p_3^{0.15} \tag{4.29}$$

The result (rounded off) is $p_1 = 3.2$, $p_2 = 4.0$, and $p_3 = 3.9$ to be sold for unit prices of 8.77, 13.96, and 18.23 respectively, for a maximum net revenue of 155.75. This would require water allocations $x_1 = 10.2$, $x_2 = 13.6$ and $x_3 = 14.5$, totaling 38.3 flow units. Any amount of water less than 38.3 will restrict the allocation to, and hence the production at, one or more of the three firms.

If the total available amount of water is less than that desired, constraint Equation 4.25 can be written as an equality, since all the water available, less any that must remain in the river, R, will be allocated. If the available water supplies are less than the desired 38.3 plus the required streamflow R, then Equations 4.22 through 4.25 need to be added. These can be rewritten as equalities since they will be binding.

$$p_1 - 0.4(x_1)^{0.9} = 0 \tag{4.30}$$

$$p_2 - 0.5(x_2)^{0.8} = 0 \tag{4.31}$$

$$p_3 - 0.6(x_3)^{0.7} = 0 \tag{4.32}$$

$$R + x_1 + x_2 + x_3 - Q = 0 \tag{4.33}$$

The first three constraints, Equations 4.30, 4.31 and 4.32, are the production functions specifying the relationships between each water input x_j and product output p_j. The fourth constraint Equation 4.33 is the restriction on the total allocation of water. Since each of the four constraint equations equals zero, each can be added to the net benefit Equation 4.26 without changing its value. This is done in Equation 4.34, which is equivalent to Equation 4.26. The variable L is the value of the Lagrange form of the objective function:

$$\begin{aligned} L = {} & [(12 - p_1)p_1 + (20 - 1.5p_2)p_2 + (28 - 2.5p_3)p_3] \\ & - [3(p_1)^{1.30} + 5(p_2)^{1.20} + 6(p_3)^{1.15}] \\ & - \lambda_1[p_1 - 0.4(x_1)^{0.9}] - \lambda_2[p_2 - 0.5(x_2)^{0.8}] \\ & - \lambda_3[p_3 - 0.6(x_3)^{0.7}] - \lambda_4[R + x_1 + x_2 + x_3 - Q] \end{aligned} \tag{4.34}$$

Since each of the four constraint Equations 4.30 through 4.33 included in Equation 4.34 equals zero, each can be multiplied by a variable λ_i without changing the value of Equation 4.34. These unknown variables λ_i are called the Lagrange multipliers of constraints i. The value of each multiplier, λ_i, is the marginal value of the original objective function, Equation 4.26, with respect to a change in the value of the amount produced, p_i, or in the case of constraint, Equation 4.33, the amount of water available, Q. We will prove this shortly.

Differentiating Equation 4.34 with respect to each of the ten unknowns and setting the resulting equations to 0 yields:

$$\partial L/\partial p_1 = 0 = 12 - 2p_1 - 1.3(3)p_1^{0.3} - \lambda_1 \tag{4.35}$$

$$\partial L/\partial p_2 = 0 = 20 - 3p_2 - 1.2(5)p_2^{0.2} - \lambda_2 \tag{4.36}$$

$$\partial L/\partial p_3 = 0 = 28 - 5p_3 - 1.15(6)p_3^{0.15} - \lambda_3 \tag{4.37}$$

$$\partial L/\partial x_1 = 0 = \lambda_1 0.9(0.4)(x_1)^{-0.1} - \lambda_4 \tag{4.38}$$

$$\partial L/\partial x_2 = 0 = \lambda_2 0.8(0.5)(x_2)^{-0.2} - \lambda_4 \tag{4.39}$$

$$\partial L/\partial x_3 = 0 = \lambda_3 0.7(0.6)(x_3)^{-0.3} - \lambda_4 \tag{4.40}$$

$$\partial L/\partial \lambda_1 = 0 = p_1 - 0.4(x_1)^{0.9} \tag{4.41}$$

$$\partial L/\partial \lambda_2 = 0 = p_2 - 0.5(x_2)^{0.8} \tag{4.42}$$

$$\partial L/\partial \lambda_3 = 0 = p_3 - 0.6(x_3)^{0.7} \tag{4.43}$$

$$\partial L/\partial \lambda_4 = 0 = R + x_1 + x_2 + x_3 - Q \tag{4.44}$$

These ten equations are the conditions necessary for an optimal solution. They can be solved to obtain the values of the ten unknown variables. The solutions to these equations for various values of Q, (found in this case by using LINGO) are shown in Table 4.2. (A demo version of LINGO is available, together with its help files, at www.lindo.com.)

3.3.2. Meaning of the Lagrange Multiplier

In this example, Equation 4.34 is the objective function. It is maximized (or minimized) by equating to zero each

Table 4.2. Solutions to Equations 4.35 through 4.44.

water	allocations to firms			product productions			Lagrange multipliers			
available	1	2	3	1	2	3	marginal net benefits			
Q-R	x_1	x_2	x_3	p_1	p_2	p_3	λ_1	λ_2	λ_3	λ_4
10	1.2	3.7	5.1	0.46	1.44	1.88	8.0	9.2	11.0	2.8
20	4.2	7.3	8.5	1.46	2.45	2.68	4.7	5.5	6.6	1.5
30	7.5	10.7	11.7	2.46	3.34	3.37	2.0	2.3	2.9	0.6
38	10.1	13.5	14.4	3.20	4.00	3.89	0.1	0.1	0.1	0.0
38.3	10.2	13.6	14.5	3.22	4.02	3.91	0	0	0	0

E020820h

of its partial derivatives with respect to each unknown variable. Equation 4.34 consists of the original net benefit function plus each constraint i multiplied by a weight or multiplier λ_i. This equation is expressed in monetary units, such as dollars or euros. The added constraints are expressed in other units: either the quantity of product produced or the amount of water available. Thus the units of the weights or multipliers λ_i associated with these constraints are expressed in monetary units per constraint units. In this example the multipliers λ_1, λ_2 and λ_3 represent the change in the total net benefit value of the objective function (Equation 4.26) per unit change in the products p_1, p_2 and p_3 produced. The multiplier λ_4 represents the change in the total net benefit per unit change in the water available for allocation, $Q - R$.

Note in Table 4.2 that as the quantity of available water increases, the marginal net benefits decrease. This is reflected in the values of each of the multipliers, λ_i. In other words, the net revenue derived from a quantity of product produced at each of the three firms, and from the quantity of water available, are concave functions of those quantities, as illustrated in Figure 4.2.

To review the general Lagrange multiplier approach and derive the definition of the multipliers, consider the general constrained optimization problem containing n decision-variables x_j and m constraint equations i.

$$\text{Maximize (or minimize) } F(\boldsymbol{X}) \tag{4.45}$$

subject to constraints

$$g_i(\boldsymbol{X}) = b_i \quad i = 1, 2, 3, \ldots, m \tag{4.46}$$

where $\boldsymbol{X}$ is the vector of all x_j. The Lagrange function $L(\boldsymbol{X}, \boldsymbol{\lambda})$ is formed by combining Equations 4.46, each equalling zero, with the objective function of Equation 4.45.

$$L(\boldsymbol{X}, \boldsymbol{\lambda}) = F(\boldsymbol{X}) - \sum_i \lambda_i (g_i(\boldsymbol{X}) - b_i) \tag{4.47}$$

Solutions of the equations

$$\partial L/\partial x_j = 0 \quad \text{for all decision-variables } j$$

and

$$\partial L/\partial \lambda_i = 0 \quad \text{for all constraints } i \tag{4.48}$$

are possible local optima.

There is no guarantee that a global optimum solution will be found using calculus-based methods such as this one. Boundary conditions need to be checked. Furthermore, since there is no difference in the Lagrange multipliers procedure for finding a minimum or a maximum solution, one needs to check whether in fact a maximum or minimum is being obtained. In this example, since each net benefit function is concave, a maximum will result.

The meaning of the values of the multipliers λ_i at the optimum solution can be derived by manipulation of Equation 4.48. Taking the partial derivative of the Lagrange function, Equation 4.47, with respect to an unknown variable x_j and setting it to zero results in:

$$\partial L/\partial x_j = 0 = \partial F/\partial x_j - \sum_i \lambda_i \partial(g_i(\boldsymbol{X}))/\partial x_j \tag{4.49}$$

Multiplying each term by ∂x_j yields

$$\partial F = \sum_i \lambda_i \partial(g_i(\boldsymbol{X})) \tag{4.50}$$

Dividing each term by ∂b_k associated with a particular constraint, say k, defines the meaning of λ_k.

$$\partial F/\partial b_k = \sum_i \lambda_i \partial(g_i(\boldsymbol{X}))/\partial b_k = \lambda_k \tag{4.51}$$

Equation 4.51 follows from the fact that $\partial(g_i(\boldsymbol{X}))/\partial b_k = 0$ for constraints $i \neq k$ and $\partial(g_i(\boldsymbol{X}))/\partial b_k = 1$ for the constraint $i = k$. The latter is true since $b_i = g_i(\boldsymbol{X})$ and thus $\partial(g_i(\boldsymbol{X})) = \partial b_i$.

Thus from Equation 4.51, each multiplier λ_k is the marginal change in the original objective function $F(\boldsymbol{X})$ with respect to a change in the constant b_k associated with the constraint k. For non-linear problems it is the slope of the objective function plotted against the value of b_k.

Readers can work out a similar proof if a slack or surplus variable, S_i, is included in inequality constraints to make them equations. For a less-than-or-equal constraint $g_i(\boldsymbol{X}) \leq b_i$ a squared slack variable S_i^2 can be added to the left-hand side to make it an equation $g_i(\boldsymbol{X}) + S_i^2 = b_i$. For a greater-than-or-equal constraint $g_i(\boldsymbol{X}) \geq b_i$ a squared surplus variable S_i^2 can be subtracted from the left hand side to make it an equation $g_i(\boldsymbol{X}) - S_i^2 = b_i$. These slack or surplus variables are squared to ensure they are non-negative, and also to make them appear in the differential equations.

$$\partial L/\partial S_i = 0 = -2S_i\lambda_i = S_i\lambda_i \tag{4.52}$$

Equation 4.52 shows that either the slack or surplus variable, S_i, or the multiplier, λ_i, will always be zero. If the value of the slack or surplus variable S_i is non-zero, the constraint is redundant. The optimal solution will not be affected by the constraint. Small changes in the values, b_i, of redundant constraints will not change the optimal value of the objective function $F(X)$. Conversely, if the constraint is binding, the value of the slack or surplus variable S_i will be zero. The multiplier λ_i can be non-zero if the value of the function $F(X)$ is sensitive to the constraint value b_i.

The solution of the set of partial differential Equations 4.52 often involves a trial-and-error process, equating to zero a λ_i or a S_i for each inequality constraint and solving the remaining equations, if possible. This tedious procedure, along with the need to check boundary solutions when non-negativity conditions are imposed, detracts from the utility of classical Lagrange multiplier methods for solving all but relatively simple water resources planning problems.

4. Dynamic Programming

The water allocation problems in the previous section considered a net-benefit function for each water-using firm. In those examples they were continuous differentiable functions, a convenient attribute if methods based on calculus (such as hill-climbing or Lagrange multipliers) are to be used to find the best solution. In many practical situations these functions may not be so continuous, or so conveniently concave for maximization or convex for minimization, making calculus-based methods for their solution difficult.

A possible solution method for constrained optimization problems containing continuous and/or discontinuous functions of any shape is called *discrete dynamic programming*. Each decision-variable value can assume one of a set of discrete values. For continuous valued objective functions, the solution derived from discrete dynamic programming may therefore be only an approximation of the best one. For all practical purposes this is not a significant limitation, especially if the intervals between the discrete values of the decision-variables are not too large and if simulation modelling is used to refine the solutions identified using dynamic programming.

Dynamic programming is an approach that divides the original optimization problem, with all of its variables, into a set of smaller optimization problems, each of which needs to be solved before the overall optimum solution to the original problem can be identified. The water supply allocation problem, for example, needs to be solved for a range of water supplies available to each firm. Once this is done the particular allocations that maximize the total net benefit can be determined.

4.1. Dynamic Programming Networks and Recursive Equations

A network of nodes and links can represent each discrete dynamic programming problem. Dynamic programming methods find the best way to get to any node in that network. The nodes represent possible discrete states that can exist and the links represent the decisions one could make to get from one state to another. Figure 4.6 illustrates a portion of such a network for the three-firm allocation problem shown in Figure 4.1. In this case the total amount of water available, $Q - R$, to all three firms is 10.

Thus, dynamic programming models involve states, stages and decisions. The relationships among states, stages and decisions are represented by networks, such as that shown in Figure 4.6. The states of the system are the nodes and the values of the states are the numbers in the nodes. Each node value in this example is the quantity of water available to allocate to all remaining firms, that is, to all connected links to the right of the node. These state variable values typically represent some existing condition either before making, or after having made, a decision. The stages of the system are the separate columns of linked nodes. The links in this example represent possible allocation decisions for each of the three different firms. Each stage is a separate firm.

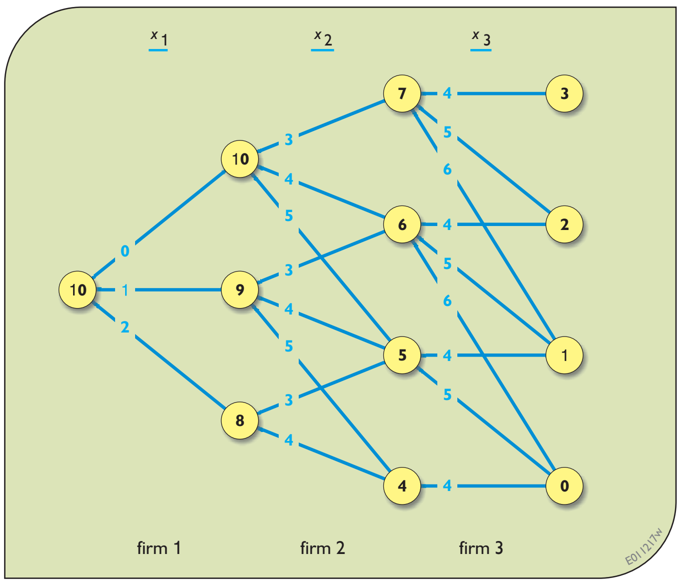


Figure 4.6. A network representing some of the possible integer allocations of water to three water-consuming firms j. The circles or nodes represent the discrete quantities of water available, and the links represent feasible allocation decisions x_j.

Each link connects two nodes, the left node value indicating the state of a system before a decision is made, and the right node value indicating the state of a system after a decision is made. In this case the state of the system is the amount of water available to allocate to the remaining firms.

In the example shown in Figure 4.6, the state and decision-variables are represented by integer values – an admittedly fairly coarse discretization. The total amount of water available, in addition to the amount that must remain in the river, is 10. Note from the first row of Table 4.2 the exact allocation solution is $x_1 = 1.2$, $x_2 = 3.7$, and $x_3 = 5.1$. Normally we wouldn't know this solution before solving for it using dynamic programming, but since we do we can reduce the complexity of the dynamic programming network so that the repetitive process of finding the best solution is clearer. Thus assume the range of x_1 is limited to integer values from 0 to 2, the range of x_2 is from 3 to 5, and the range of x_3 is from 4 to 6. These range limits are imposed here just to reduce the size of the network. In this case, these assumptions will not affect or constrain the optimal solution. If we did not make these assumptions the network would have, after the first column of one node, three columns of 11 nodes, one representing each integer value from 0 to 10. Finer (non-integer) discretizations would involve even more nodes and connecting links.

The links of Figure 4.6 represent the water allocations. Note that the link allocations, the numbers on the links, cannot exceed the amount of water available, that is, the number in the left node. The number in the right node is the quantity of water remaining after an allocation has been made. The value in the right node, state S_{j+1}, at the beginning of stage $j+1$, is equal to the value in the left node, S_j, less the amount of water, x_j, allocated to firm j as indicated on the link. Hence, beginning with a quantity of water $Q - R$ that can be allocated to all three firms, after allocating x_1 to Firm 1 what remains is S_2:

$$Q - R - x_1 = S_2 \tag{4.53}$$

Allocating x_2 to Firm 2, leaves S_3.

$$S_2 - x_2 = S_3 \tag{4.54}$$

Finally, allocating x_3 to Firm 3 leaves S_4.

$$S_3 - x_3 = S_4 \tag{4.55}$$

Figure 4.6 shows the different values of each of these states, S_j, and decision-variables x_j beginning with a quantity $Q - R = 10$. Our task is to find the best path through the network, beginning at the left-most node having a state value of 10. To do this we need to know the net benefits we will get associated with all the links (representing the allocation decisions we could make) at each node (state) for each firm (stage).

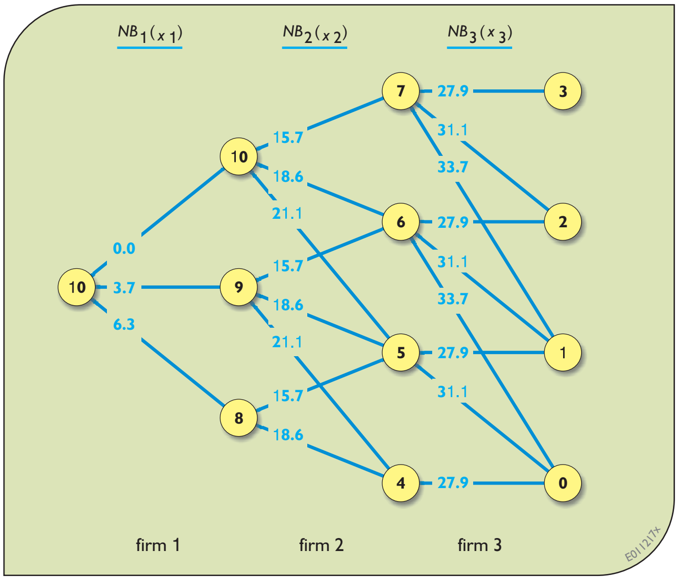


Figure 4.7. Network representing integer value allocations of water to three water-consuming firms. The circles or nodes represent the discrete quantities of water available, and the links represent feasible allocation decisions. The numbers on the links indicate the net benefits obtained from these particular allocation decisions.

Figure 4.7 shows the same network as in Figure 4.6; however the numbers on the links represent the net benefits obtained from the associated water allocations. For the three firms $j = 1$, 2 and 3, the net benefits, $NB_j(x_j)$, associated with allocations x_j are:

$$NB_1(x_1) = \text{maximum}(12 - p_1)p_1 - 3(p_1)^{1.30} \quad \text{where } p_1 \leq 0.4(x_1)^{0.9} \tag{4.56}$$

$$NB_2(x_2) = \text{maximum}(20 - 1.5p_2)p_2 - 5(p_2)^{1.20} \quad \text{where } p_2 \leq 0.5(x_2)^{0.8} \tag{4.57}$$

$$NB_3(x_3) = \text{maximum}(28 - 2.5p_3)p_3 - 6(p_3)^{1.15} \quad \text{where } p_3 \leq 0.6(x_3)^{0.7} \tag{4.58}$$

respectively.

The discrete dynamic programming algorithm or procedure is a systematic way to find the best path through this network, or any other suitable network. What makes a network suitable for dynamic programming is the fact that all the nodes can be lined up in a sequence of columns and each link connects a node in one column to another node in the next column of nodes. No link passes over or through any other column(s) of nodes. Links also do not connect nodes in the same column. In addition, the contribution to the overall objective value (in this case, the total net benefits) associated with each discrete decision (link) in any stage or for any firm is strictly a function of the allocation of water to the firm. It is not dependent on the allocation decisions associated with other stages (firms) in the network.

The main challenge in using discrete dynamic programming to solve an optimization problem is to structure the problem so that it fits this dynamic programming network format. Perhaps surprisingly, many water resources planning and management problems do. But it takes practice to become good at converting optimization problems to networks of states, stages and decisions suitable for solution by discrete dynamic programming algorithms.

In this problem the overall objective is to:

$$\text{Maximize} \sum_{j=1}^{3} NB_j(x_j) \tag{4.59}$$

where $NB_j(x_j)$ is the net benefit associated with an allocation of x_j to firm j. Equations 4.56, 4.57 and 4.58 define these net benefit functions. As before, the index j represents the particular firm, and each firm is a stage for this problem. Note that the index or subscript used in the objective function often represents an object (like a water-using firm) at a place in space or in a time period. These places or time periods are called the *stages* of a dynamic programming problem. Our task is to find the best path from one stage to the next: in other words, the best allocation decisions for all three firms.

Dynamic programming is called a multi-stage decision-making process. Instead of deciding all three allocations in one single optimization procedure, like Lagrange multipliers, the dynamic programming procedure divides the problem up into many optimization problems, one for each possible discrete state (e.g., amount of water available) in each stage (e.g., for each firm). Given a particular state S_j and stage j – that is, a particular node in the network – what decision (link) x_j will result in the maximum total net benefits, designated as $F_j(S_j)$, given this state S_j for this and all remaining stages or firms j, $j+1$, $j+2 \ldots$? This question must be answered for each node in the network before one can find the overall best set of decisions for each stage: in other words, the best allocations to each firm (represented by the best path through the network) in this example.

Dynamic programming networks can be solved in two ways – beginning at the most right column of nodes or states and moving from right to left, called the *backward-moving* (but forward-looking) algorithm, or beginning at the left most node and moving from left to right, called the *forward-moving* (but backward-looking) algorithm. Both methods will find the best path through the network. In some problems, however, only the backward-moving algorithm produces a useful solution. This is especially relevant when the stages are time periods. We often want to know what we should do next given a particular state we are in, not what we should have just done to get to the particular state we are in. We cannot alter past decisions, but we can, and indeed must, make future decisions. We will revisit this issue when we get to reservoir operation where the stages are time periods.

4.2. Backward-Moving Solution Procedure

Consider the network in Figure 4.7. Again, the nodes represent the discrete states – the water available to allocate to all remaining users. The links represent particular discrete allocation decisions. The numbers on the links are the net benefits obtained from those allocations. We want to proceed through the node-link network from the state of 10 at the beginning of the first stage to the end of the network in such a way as to maximize total net benefits. But without looking at all combinations of successive allocations we cannot do this beginning at a state of 10. However, we can find the best solution if we assume we have already made the first two allocations and are at any of the nodes or states at the beginning of the final, third, stage with only one allocation decision remaining. Clearly at each node representing the water available to allocate to the third firm, the best decision is to pick the allocation (link) having the largest net benefits.

Denoting $F_3(S_3)$ as the maximum net benefits we can achieve from the remaining amount of water S_3, then for each discrete value of S_3 we can find the x_3 that maximizes $F_3(S_3)$. Those shown in Figure 4.7 include:

$$F_3(7) = \text{Maximum}\{NB_3(x_3)\}$$
$$x_3 \le 7, \text{ the total flow available.}$$

$$4 \le x_3 \le 6, \text{ the allowable range of allocation decisions.}$$

$$= \text{Maximum}\{27.9, 31.1, 33.7\} = 33.7 \quad \text{when } x_3 = 6 \tag{4.60}$$

$$F_3(6) = \text{Maximum}\{NB_3(x_3)\}$$
$$x_3 \le 6$$
$$4 \le x_3 \le 6$$
$$= \text{Maximum}\{27.9, 31.1, 33.7\} = 33.7 \quad \text{when } x_3 = 6 \tag{4.61}$$

$$F_3(5) = \text{Maximum}\{NB_3(x_3)\}$$
$$x_3 \le 5$$
$$4 \le x_3 \le 6$$
$$= \text{Maximum}\{27.9, 31.1\} = 31.1 \quad \text{when } x_3 = 5 \tag{4.62}$$

$$F_3(4) = \text{Maximum}\{NB_3(x_3)\}$$
$$x_3 \le 4$$
$$4 \le x_3 \le 6$$
$$= \text{Maximum}\{27.9\} = 27.9 \quad \text{when } x_3 = 4 \tag{4.63}$$

These computations are shown on the network in Figure 4.8. Note that there are no benefits to be obtained after the third allocation, so the decision to be made for each node or state prior to allocating water to Firm 3 is simply that which maximizes the net benefits derived from that last (third) allocation. In Figure 4.8 the links representing the decisions or allocations that result in the largest net benefits are shown with arrows.

Having computed the maximum net benefits, $F_3(S_3)$, associated with each initial state S_3 for Stage 3, we can now move backward (to the left) to the discrete states S_2 at the beginning of the second stage. Again, these states represent the quantity of water available to allocate to Firms 2 and 3. Denote $F_2(S_2)$ as the maximum total net benefits obtained from the two remaining allocations x_2 and x_3 given the quantity S_2 water available. The best x_2 depends not only on the net benefits obtained from the allocation x_2 but also on the maximum net benefits obtainable after that, namely the just-calculated $F_3(S_3)$ associated with the state S_3 that results from the initial state S_2 and a decision x_2. As defined in Equation 4.54, this final state S_3 in Stage 2 obviously equals $S_2 - x_2$. Hence for those nodes at the beginning of Stage 2 shown in Figure 4.8:

$$F_2(10) = \text{Maximum}\{NB_2(x_2) + F_3(S_3 = 10 - x_2)\} \tag{4.64}$$

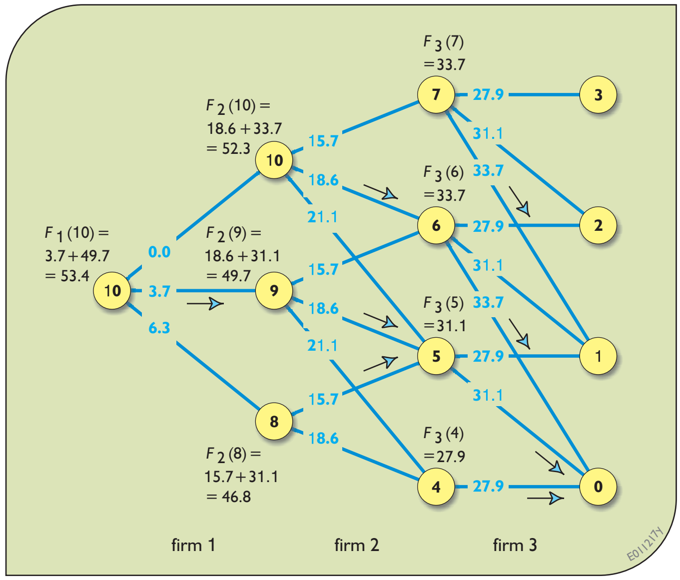


Figure 4.8. Using the backward-moving dynamic programming method for finding the maximum remaining net benefits, $F_j(S_j)$, and optimal allocations (denoted by the arrows on the links) for each state in Stage 3, then for each state in Stage 2 and finally for the initial state in Stage 1 to obtain the optimum allocation policy and maximum total net benefits, $F_1(10)$. The minimum flow to remain in the river, R, is in addition to the ten units available for allocation and is not shown in this network.

$$x_2 \le 10$$
$$3 \le x_2 \le 5$$
$$= \text{Maximum}\{15.7 + 33.7, 18.6 + 33.7, 21.1 + 31.1\} = 52.3 \quad \text{when } x_2 = 4$$

$$F_2(9) = \text{Maximum}\{NB_2(x_2) + F_3(S_3 = 9 - x_2)\} \tag{4.65}$$
$$x_2 \le 9$$
$$3 \le x_2 \le 5$$
$$= \text{Maximum}\{15.7 + 33.7, 18.6 + 31.1, 21.1 + 27.9\} = 49.7 \quad \text{when } x_2 = 4$$

$$F_2(8) = \text{Maximum}\{NB_2(x_2) + F_3(S_3 = 8 - x_2)\} \tag{4.66}$$
$$x_2 \le 8$$
$$3 \le x_2 \le 5$$
$$= \text{Maximum}\{15.7 + 31.1, 18.6 + 27.9\} = 46.8 \quad \text{when } x_2 = 3$$

These maximum net benefit functions, $F_2(S_2)$, could be calculated for the remaining discrete states from 7 to 0.

Having computed the maximum net benefits obtainable for each discrete state at the beginning of Stage 2, that is, all the $F_2(S_2)$ values, we can move backward or left to the beginning of Stage 1. For this problem there is only one state, the state of 10 we are actually in before making any allocations to any of the firms. In this case, the maximum net benefits, $F_1(10)$, we can obtain from all three allocations given 10 units of water available, is

$$F_1(10) = \text{Maximum}\{NB_1(x_1) + F_2(S_2 = 10 - x_1)\} \quad (4.67)$$
$$x_1 \leq 10$$
$$0 \leq x_1 \leq 2$$
$$= \text{Maximum}\{0 + 52.3,\ 3.7 + 49.7,\ 6.3 + 46.8\} = 53.4 \quad \text{when } x_1 = 1$$

Equation 4.67 is another way of expressing Equation 4.59. Their values are the same. It is the maximum net benefits obtainable from allocating the available 10 units of water. From Equation 4.67 we know that we will get a maximum of 53.4 net benefits if we allocate 1 unit of water to Firm 1. This leaves 9 units of water to allocate to the two remaining firms. This is our optimal state at the beginning of Stage 2. Given a state of 9 at the beginning of Stage 2, we see from Equation 4.65 that we should allocate 4 units of water to Firm 2. This leaves 5 units of water for Firm 3. Given a state of 5 at the beginning of Stage 3, Equation 4.62 tells us we should allocate all 5 units to Firm 3. All this is illustrated in Figure 4.8.

Compare this discrete solution with the continuous one defined by Lagrange multipliers as shown in Table 4.2. The exact solution, to the nearest tenth, is 1.2, 3.7, and 5.1 for x_1, x_2 and x_3 respectively. The solution just derived from discrete dynamic programming that assumed only integer values is 1, 4, and 5 respectively.

To summarize, a dynamic programming model was developed for the following problem:

$$\text{Maximize Net\_benefit} \quad (4.68)$$

Subject to:

$$\text{Net\_benefit} = \text{Total\_return} - \text{Total\_cost} \quad (4.69)$$

$$\text{Total\_return} = (12 - p_1)p_1 + (20 - 1.5p_2)p_2 + (28 - 2.5p_3)p_3 \quad (4.70)$$

$$\text{Total\_cost} = 3(p_1)^{1.30} + 5(p_2)^{1.20} + 6(p_3)^{1.15} \quad (4.71)$$

$$p_1 \leq 0.4(x_1)^{0.9} \quad (4.72)$$

$$p_2 \leq 0.5(x_2)^{0.8} \quad (4.73)$$

$$p_3 \leq 0.6(x_3)^{0.7} \quad (4.74)$$

$$x_1 + x_2 + x_3 \leq 10 \quad (4.75)$$

The discrete dynamic programming version of this problem required discrete states S_j representing the amount of water available to allocate to firms $j, j+1, \ldots$. It required discrete allocations x_j. Next it required the calculation of the maximum net benefits, $F_j(S_j)$, that could be obtained from all firms j, beginning with Firm 3, and proceeding backwards as indicated in Equations 4.76 to 4.78.

$$F_3(S_3) = \text{maximum}\{NB_3(x_3)\} \text{ over all } x_3 \leq S_3, \quad \text{for all discrete } S_3 \text{ values between 0 and 10} \quad (4.76)$$

$$F_2(S_2) = \text{maximum}\{NB_2(x_2) + F_3(S_3)\} \text{ over all } x_2 \leq S_2 \text{ and } S_3 = S_2 - x_2, \quad 0 \leq S_2 \leq 10 \quad (4.77)$$

$$F_1(S_1) = \text{maximum}\{NB_1(x_1) + F_2(S_2)\} \text{ over all } x_1 \leq S_1 \text{ and } S_2 = S_1 - x_1 \text{ and } S_1 = 10 \quad (4.78)$$

To solve for $F_1(S_1)$ and each optimal allocation x_j we must first solve for all values of $F_3(S_3)$. Once these are known we can solve for all values of $F_2(S_2)$. Given these $F_2(S_2)$ values, we can solve for $F_1(S_1)$. Equations 4.76 need to be solved before Equations 4.77 can be solved, and Equations 4.77 need to be solved before Equations 4.78 can be solved. They need not be solved simultaneously, and they cannot be solved in reverse order. These three equations are called recursive equations. They are defined for the backward-moving dynamic programming solution procedure.

There is a correspondence between the non-linear optimization model defined by Equations 4.68 to 4.75 and the dynamic programming model defined by the recursive Equations 4.76, 4.77 and 4.78. Note that $F_3(S_3)$ in Equation 4.76 is the same as:

$$F_3(S_3) = \text{Maximum } NB_3(x_3) \quad (4.79)$$

Subject to:

$$x_3 \leq S_3 \quad (4.80)$$

where $NB_3(x_3)$ is defined in Equation 4.58.

Similarly, $F_2(S_2)$ in Equation 4.62 is the same as:

$$F_2(S_2) = \text{Maximum } NB_2(x_2) + NB_3(x_3) \quad (4.81)$$

Subject to:

$$x_2 + x_3 \leq S_2 \quad (4.82)$$

where $NB_2(x_2)$ and $NB_3(x_3)$ are defined in Equations 4.57 and 4.58.

Finally, $F_1(S_1)$ in Equation 4.63 is the same as:

$$F_1(S_1) = \text{Maximum } NB_1(x_1) + NB_2(x_2) + NB_3(x_3) \quad (4.83)$$

Subject to:

$$x_1 + x_2 + x_3 \leq S_1 = 10 \quad (4.84)$$

where $NB_1(x_1)$, $NB_2(x_2)$ and $NB_3(x_3)$ are defined in Equations 4.56, 4.57 and 4.58.

Alternatively, $F_3(S_3)$ in Equation 4.76 is the same as:

$$F_3(S_3) = \text{Maximum}(28 - 2.5p_3)p_3 - 6(p_3)^{1.15} \quad (4.85)$$

Subject to:

$$p_3 \leq 0.6(x_3)^{0.7} \quad (4.86)$$

$$x_3 \leq S_3 \quad (4.87)$$

Similarly, $F_2(S_2)$ in Equation 4.77 is the same as:

$$F_2(S_2) = \text{Maximum}(20 - 1.5p_2)p_2 + (28 - 2.5p_3) \times p_3 - 5(p_2)^{1.20} - 6(p_3)^{1.15} \quad (4.88)$$

Subject to:

$$p_2 \leq 0.5(x_2)^{0.8} \quad (4.89)$$

$$p_3 \leq 0.6(x_3)^{0.7} \quad (4.90)$$

$$x_2 + x_3 \leq S_2 \quad (4.91)$$

Finally, $F_1(S_1)$ in Equation 4.78 is the same as:

$$F_1(S_1) = \text{Maximum}(12 - p_1)p_1 + (20 - 1.5p_2)p_2 + (28 - 2.5p_3)p_3 - [3(p_1)^{1.30} + 5(p_2)^{1.20} + 6(p_3)^{1.15}] \quad (4.92)$$

Subject to:

$$p_1 \leq 0.4(x_1)^{0.9} \quad (4.93)$$

$$p_2 \leq 0.5(x_2)^{0.8} \quad (4.94)$$

$$p_3 \leq 0.6(x_3)^{0.7} \quad (4.95)$$

$$x_1 + x_2 + x_3 \leq S_1 = 10 \quad (4.96)$$

The transition function of dynamic programming defines the relationship between two successive states S_j and S_{j+1} and the decision x_j. In the above example these transition functions are defined by Equations 4.53, 4.54 and 4.55, or, in general terms for all firms j, by:

$$S_{j+1} = S_j - x_j \quad (4.97)$$

4.3. Forward-Moving Solution Procedure

We have just described the backward-moving dynamic programming algorithm. In that approach at each node (state) in each stage we calculated the best value of the objective function that can be obtained from all further or remaining decisions. Alternatively one can proceed forward, that is, from left to right, through a dynamic programming network. For the forward-moving algorithm at each node we need to calculate the best value of the objective function that could be obtained from all past decisions leading to that node or state. In other words, we need to find how best to get to each state S_{j+1} at the end of each stage j.

Returning to the allocation example, define $f_j(S_{j+1})$ as the maximum net benefits from the allocation of water to firms 1, 2, … , j, given a state S_{j+1} after having made those allocations. For this example we begin the forward-moving, but backward-looking, process by selecting each of the ending states in the first stage $j = 1$ and finding the best way to have arrived at (or to have achieved) those ending states. Since in this example there is only one way to get to each of those states, as shown in Figure 4.6 or 4.7, the allocation decisions are obvious.

$$f_1(S_2) = \text{maximum}\{NB_1(x_1)\}$$
$$x_1 = 10 - S_2 \quad (4.98)$$

Hence, $f_1(S_2)$ is simply $NB_1(10 - S_2)$. Once the values for all $f_1(S_2)$ are known for all discrete S_2 between 0 and 10, move forward (to the right) to the end of Stage 2 and find the best allocations to have made given each final state S_3.

$$f_2(S_3) = \text{maximum}\{NB_2(x_2) + f_1(S_2)\}$$
$$0 \leq x_2 \leq 10 - S_3$$
$$S_2 = S_3 + x_2 \quad (4.99)$$

Once the values of all $f_2(S_3)$ are known for all discrete states S_3 between 0 and 10, move forward to Stage 3 and find the best allocations to have made given each final state S_4.

$$f_3(S_4) = \text{maximum}\{NB_3(x_3) + f_2(S_3)\}$$

for all discrete S_4 between 0 and 10.

$$0 \leq x_3 \leq 10 - S_4$$
$$S_3 = S_4 + x_3 \quad (4.100)$$

Figure 4.9 illustrates a portion of the network represented by Equations 4.98 through 4.100, and the $f_j(S_{j+1})$ values.

From Figure 4.9, note the highest total net benefits are obtained by ending with 0 remaining water at the end of Stage 3. The arrow tells us that if we are to get to that state optimally, we should allocate 5 units of water to Firm 3. Thus we must begin Stage 3, or end Stage 2, with $10 - 5 = 5$ units of water. To get to this state at the end of Stage 2 we should allocate 4 units of water to Firm 2.

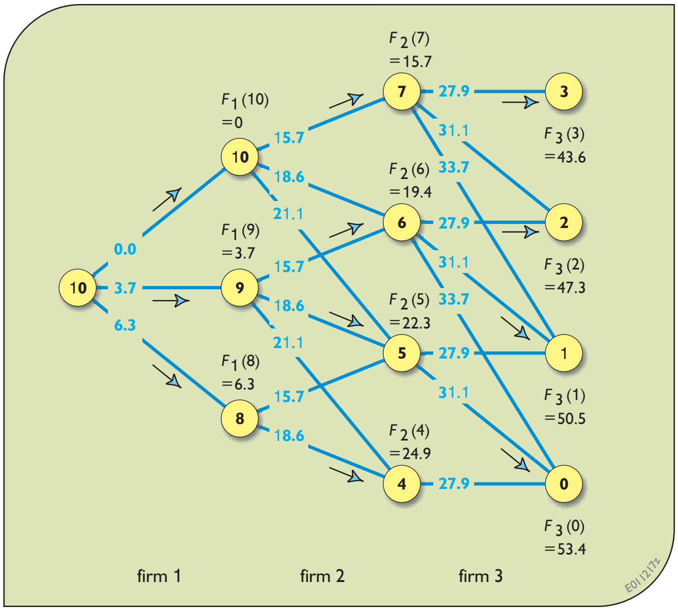


Figure 4.9. Using the forward-moving dynamic programming method for finding the maximum accumulated net benefits, $F_j(S_{j+1})$, and optimal allocations (denoted by the arrows on the links) that should have been made to reach each ending state, beginning with the ending states in Stage 1, then for each ending state in Stage 2 and finally for the ending states in Stage 3.

The arrow also tells us we should have had 9 units of water available at the end of Stage 1. Given this state of 9 at the end of Stage 1, the arrow tells us we should allocate 1 unit of water to Firm 1. This is the same allocation policy as obtained using the backward-moving algorithm.

4.4. Numerical Solutions

The application of discrete dynamic programming to most practical problems will require writing some software. There are no general dynamic programming computer programs available that will solve all dynamic programming problems. Thus any user of dynamic programming will need to write a computer program to solve a particular problem. Most computer programs written for solving specific dynamic programming problems create and store the solutions of the recursive equations in tables. Each stage is a separate table, as shown in Tables 4.3, 4.4 and 4.5 for this example water allocation problem. These tables apply to only a part of the entire problem, namely that part of the network shown in Figures 4.8 and 4.9. The backward solution procedure is used.

Table 4.3 contains the solutions of Equations 4.60 to 4.63 for the third stage. Table 4.4 contains the solutions of Equations 4.64 to 4.66 for the second stage. Table 4.5 contains the solution of Equation 4.67 for the first stage.

From Table 4.5 we see that, given 10 units of water available, we will obtain 53.4 net benefits and to get this we should allocate 1 unit to Firm 1. This leaves 9 units of water for the remaining two allocations. From Table 4.4 we see that for a state of 9 units of water available we should allocate 4 units to Firm 2. This leaves 5 units. From Table 4.3 for a state of 5 units of water available we see we should allocate all 5 of them to Firm 3.

Performing these calculations for various discrete total amounts of water available, say from 0 to 38 in this example, will define an allocation policy (such as the one shown in Figure 4.6 for a different allocation problem) for situations when the total amount of water is less than that desired by all the firms. This policy can then be simulated using alternative time series of available amounts of water, such as streamflows, to obtain estimates of the time series (or statistical measures of those time series) of net benefits obtained by each firm, assuming the allocation policy is followed over time.

Table 4.3. Computing the values of $F_3(S_3)$ and optimal allocations x_3 for all states S_3 in Stage 3.

E020820j

remaining net benefits $NB_3(S_3)$

	decisions: x_3				
state S_3	4	5	6	$F_3(S_3)$	x_3
7	27.9	31.1	33.7	33.7	6
6	27.9	31.1	33.7	33.7	6
5	27.9	31.1	---	31.1	5
4	27.9	---	---	27.9	4

remaining net benefits $NB_2(S_2) + F_3(S_3 = S_2 - x_2)$					
	decisions: x_2				
state S_2	3	4	5	$F_2(S_2)$	x_2
10	15.7 + 33.7	18.6 + 33.7	21.1 + 31.1	52.3	4
9	15.7 + 33.7	18.6 + 31.1	21.1 + 27.9	49.7	4
8	15.7 + 31.1	18.6 + 27.9	---	46.8	3

E020820k

Table 4.4. Computing the values of $F_2(S_2)$ and optimal allocations x_2 for all states S_2 in Stage 2.

remaining net benefits $NB_1(S_1) + F_2(S_2 = S_1 - x_1)$					
	decisions: x_2				
state S_2	0	1	2	$F_2(S_2)$	x_2
10	0 + 52.3	3.7 + 49.7	6.3 + 46.8	53.4	1

E020820m

Table 4.5. Computing the values of $F_1(S_1)$ and optimal allocations x_1 for all states S_1 in Stage 1.

4.5. Dimensionality

One of the limitations of dynamic programming is handling multiple state variables. In our water allocation example we had only one state variable: the total amount of water available. We could have enlarged this problem to include other types of resources the firms require to make their products. Each of these state variables would need to be discretized. If, for example, only m discrete values of each state variable are considered, for n different state variables (e.g., types of resources) there are m^n different combinations of state variable values to consider at each stage. As the number of state variables increases, the number of discrete combinations of state variable values increases exponentially. This is called dynamic programming's '*curse of dimensionality*'. It has motivated many researchers to search for ways of reducing the number of possible discrete states required to find an optimal solution to large multi-state problems.

4.6. Principle of Optimality

The solution of dynamic programming models or networks is based on a principal of optimality (Bellman, 1957). The backward-moving solution algorithm is based on the principal that no matter what the state and stage (i.e., the particular node you are at), an optimal policy is one that proceeds forward from that node or state and stage optimally. The forward-moving solution algorithm is based on the principal that no matter what the state and stage (i.e., the particular node you are at), an optimal policy is one that has arrived at that node or state and stage in an optimal manner.

This 'principle of optimality' is a very simple concept but requires the formulation of a set of recursive equations at each stage. It also requires that either in the last stage ($j = J$) for a backward-moving algorithm, or in the first stage ($j = 1$) for a forward-moving algorithm, the future value functions, $F_{J+1}(S_{J+1})$, associated with the ending state variable values, or past value functions, $f_0(S_1)$, associated with the beginning state variable values, respectively, all equal some known value. Usually that value is 0 but not always. This condition is required in order to begin the process of solving each successive recursive equation.

4.7. Additional Applications

Among the common dynamic programming applications in water resources planning are water allocations to multiple uses, infrastructure capacity expansion, and reservoir operation. The previous three-user water allocation problem (Figure 4.1) illustrates the first type of application. The other two applications are presented below.

4.7.1. Capacity Expansion

How much infrastructure should be built, when and why? Consider a municipality that must plan for the future expansion of its water supply system or some component of that system, such as a reservoir, aqueduct, or treatment plant. The capacity needed at the end of each future period t has been estimated to be D_t. The cost, $C_t(s_t, x_t)$, of adding capacity x_t in each period t is a function of that added capacity as well as of the existing capacity s_t at the beginning of the period. The planning problem is to find that time sequence of capacity expansions that minimizes the present value of total future costs while meeting the predicted capacity demand requirements. This is the usual capacity-expansion problem.

This problem can be written as an optimization model: The objective is to minimize the present value of the total cost of capacity expansion.

$$\text{Minimize } \sum_t C_t(s_t, x_t) \tag{4.101}$$

where $C_t(s_t, x_t)$ is the present value of the cost of capacity expansion x_t in period t given an initial capacity of s_t.

The constraints of this model define the minimum required final capacity in each period t, or equivalently the next period's initial capacity, s_{t+1}, as a function of the known existing capacity s_1 and each expansion x_t up through period t.

$$s_{t+1} = s_t + \sum_{\tau=1}^{t} x_\tau \quad \text{for } t = 1, 2, \ldots, T \tag{4.102}$$

Alternatively these equations may be expressed by a series of continuity relationships:

$$s_{t+1} = s_t + x_t \quad \text{for } t = 1, 2, \ldots, T \tag{4.103}$$

In this problem, the constraints must also ensure that the actual capacity s_{t+1} at the end of each future period t is no less than the capacity required D_t at the end of that period.

$$s_{t+1} \geq D_t \quad \text{for } t = 1, 2, \ldots, T \tag{4.104}$$

There may also be constraints on the possible expansions in each period defined by a set Ω_t of feasible capacity additions in each period t:

$$x_t \in \Omega_t \tag{4.105}$$

Figure 4.10 illustrates this type of capacity-expansion problem. The question is how much capacity to add and when. It is a significant problem for several reasons. One is that the cost functions $C_t(s_t, x_t)$ typically exhibit

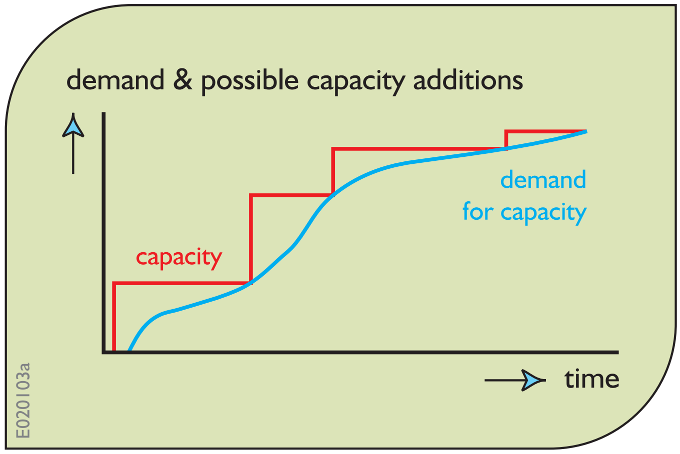


Figure 4.10. A demand projection (solid line) and a possible capacity-expansion schedule for meeting that projected demand over time.

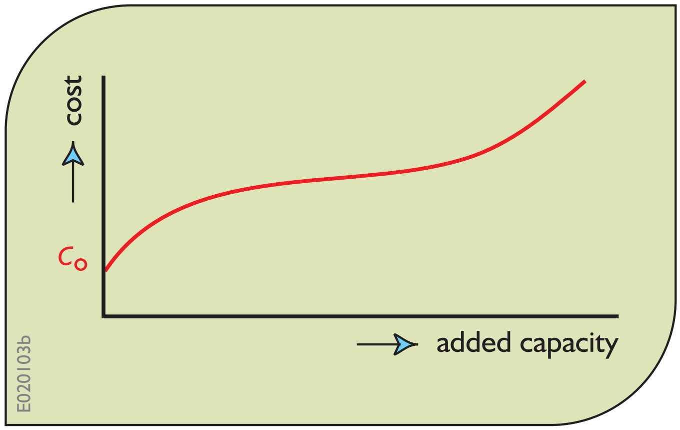


Figure 4.11. Typical cost function for additional capacity given an existing capacity. The cost function shows the fixed costs, C_0, required if additional capacity is to be added, and the economies of scale associated with the concave portion of the cost function.

fixed costs and economies of scale, as illustrated in Figure 4.11. Each time any capacity is added there are fixed as well as variable costs incurred. Fixed and variable costs that show economies of scale (decreasing average costs associated with increasing capacity additions) motivate the addition of excess capacity, capacity not needed immediately but expected to be needed eventually to meet an increased demand for additional capacity in the future.

The problem is also important because any estimates made today of future demands, costs and interest rates are likely to be wrong. The future is uncertain. Its uncertainties increase the further we look into the future. Capacity-expansion planners need to consider the future if their plans are to be cost-effective. Just how far into the future do they need to look? And what about the uncertainty in all future estimates? These questions will be addressed after showing how the problem can be solved for any fixed-planning horizon and estimates of future demands, interest rates and costs.

The constrained optimization model defined by Equations 4.101 to 4.105 can be restructured as a multistage decision-making process and solved using either a forward or backward-moving discrete dynamic programming solution procedure. The stages of the model will be the time periods t. The states will be either the capacity s_{t+1} at the end of a stage or period t if a forward-moving solution procedure is adopted, or the capacity s_t, at the beginning of a stage or period t if a backward-moving solution procedure is used.

A network of possible discrete capacity states and decisions can be superimposed onto the demand projection of Figure 4.10, as shown in Figure 4.12.

The solid blue circles in Figure 4.12 represent possible discrete states, S_t, of the system, the amounts of additional capacity existing at the end of each period $t-1$ or equivalently at the beginning of period t.

Consider first a forward-moving dynamic programming algorithm. To implement this, define $f_t(s_{t+1})$ as the minimum cost of achieving a capacity s_{t+1}, at the end of period t. Since at the beginning of the first period $t = 1$, the accumulated least cost is 0, $f_0(s_1) = 0$.

Hence, for each final discrete state s_2 in stage $t = 1$ ranging from D_1 to the maximum demand D_T, define

$$f_1(s_2) = \min\{C_1(s_1, x_1)\} \text{ in which the discrete } x_1 = s_2 \text{ and } s_1 = 0 \quad (4.106)$$

Moving to stage $t = 2$, for the final discrete states s_3 ranging from D_2 to D_T,

$$f_2(s_3) = \min\{C_2(s_2, x_2) + f_1(s_2)\} \text{ over all discrete } x_2 \text{ between 0 and } s_3 - D_1$$

$$\text{where } s_2 = s_3 - x_2 \quad (4.107)$$

Moving to stage $t = 3$, for the final discrete states s_4 ranging from D_3 to D_T,

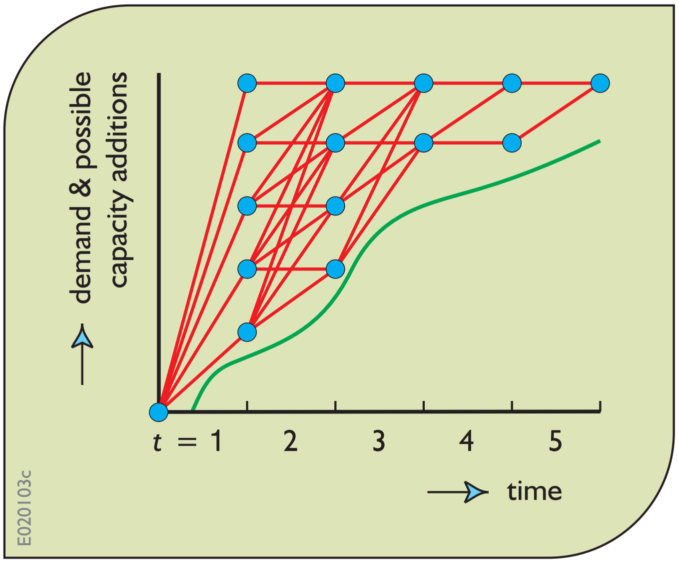


Figure 4.12. Network of discrete capacity-expansion decisions (links) that meet the projected demand.

$$f_3(s_4) = \min\{C_3(s_3, x_3) + f_2(s_3)\} \text{ over all discrete } x_3 \text{ between 0 and } s_4 - D_2$$

$$\text{where } s_3 = s_4 - x_3 \quad (4.108)$$

In general for all stages t between the first and last:

$$f_t(s_{t+1}) = \min\{C_t(s_t, x_t) + f_{t-1}(s_t)\} \text{ over all discrete } x_t \text{ between 0 and } s_{t+1} - D_{t-1}$$

$$\text{where } s_t = s_{t+1} - x_t \quad (4.109)$$

For the last stage $t = T$ and for the final discrete state $s_{T+1} = D_T$,

$$f_T(s_{T+1}) = \min\{C_T(s_T, x_T) + f_{T-1}(s_T)\} \text{ over all discrete } x_T \text{ between 0 and } D_T - D_{T-1}$$

$$\text{where } s_T = s_{T+1} - x_T \quad (4.110)$$

The value of $f_T(s_{T+1})$ is the minimum present value of the total cost of meeting the demand for T time periods. To identify the sequence of capacity-expansion decisions that results in this minimum present value of the total cost requires backtracking to collect the set of best decisions x_t for all stages t. A numerical example will illustrate this.

A numerical example

Consider the five-period capacity-expansion problem shown in Figure 4.12. Figure 4.13 is the same network

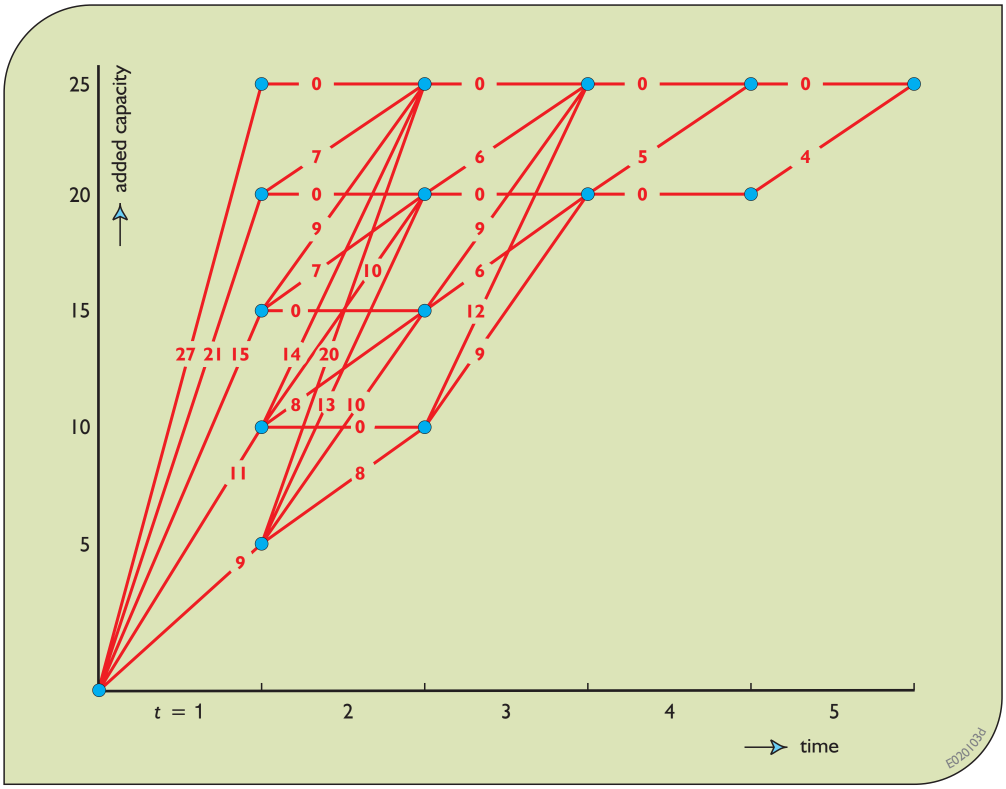


Figure 4.13. A discrete capacity-expansion network showing the present value of the expansion costs associated with each feasible expansion decision. Finding the best path through the network can be done using forward or backward-moving discrete dynamic programming.

with the present value of the expansion costs on each link. The values of the states, the existing capacities, represented by the nodes, are shown on the left vertical axis. The capacity-expansion problem is solved on Figure 4.14 using the forward-moving algorithm.

From the forward-moving solution to the dynamic programming problem shown in Figure 4.14, the present value of the cost of the optimal capacity-expansion schedule is 23 units of money. Backtracking (moving left against the arrows) from the farthest right node, this schedule adds 10 units of capacity in period $t = 1$, and 15 units of capacity in period $t = 3$.

Next consider the backward-moving algorithm applied to this capacity-expansion problem. The general recursive equation for a backward-moving solution is

$$F_t(s_t) = \text{minimum}\{C_t(s_t, x_t) + F_{t+1}(s_{t+1})\} \text{ over all discrete } x_t \text{ from } D_t - s_t \text{ to } D_T - s_t$$

$$\text{for all discrete states } s_t \text{ from } D_{t-1} \text{ to } D_T \qquad (4.111)$$

where $F_{T+1}(D_T) = 0$ and as before each cost function is the discounted cost.

Once again, as shown in Figure 4.15, the minimum total present value cost is 23 if 10 units of additional capacity are added in period $t = 1$ and 15 in period $t = 3$.

Now we look to the question of the uncertainty of future demands, D_t, discounted costs, $C_t(s_t, x_t)$, as well as to the fact that the planning horizon T is only 5 time periods. Of importance is just how these uncertainties and finite planning horizon affect our decisions. While the model gives us a time series of future capacity-expansion decisions for the next 5 time periods, what is important to decision-makers is what additional capacity to add now, not what capacity to add in future periods. Does the uncertainty of future demands and costs and the 5-period planning horizon affect this first decision, x_1? This is the question to address. If the answer is no, then one can place some confidence in the value of x_1. If yes, then more study may be warranted to determine which demand and cost scenario to assume, or, if applicable, how far into the future to extend the planning horizon.

Future capacity-expansion decisions in Periods 2, 3 and so on can be based on updated information and analyses carried out closer to the time those decisions are to be made. At those times, the forecast demands and

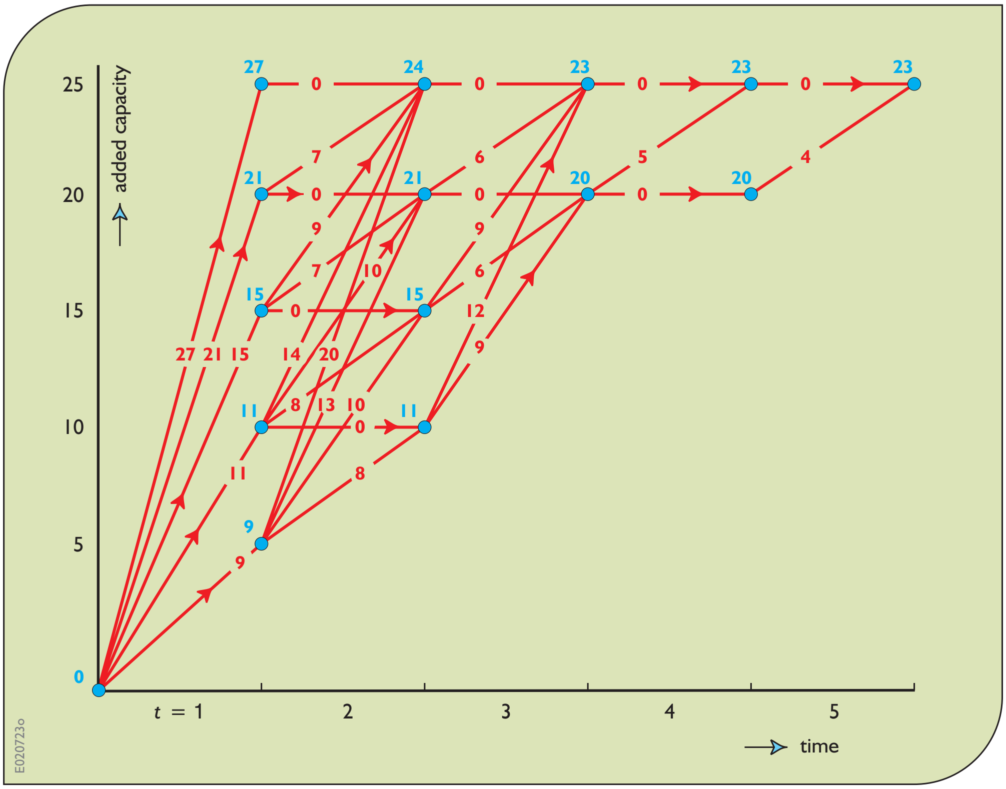


Figure 4.14. A capacity-expansion example, showing the results of a forward-moving dynamic programming algorithm. The numbers next to the nodes are the minimum cost to have reached that particular state at the end of the particular time period *t*.

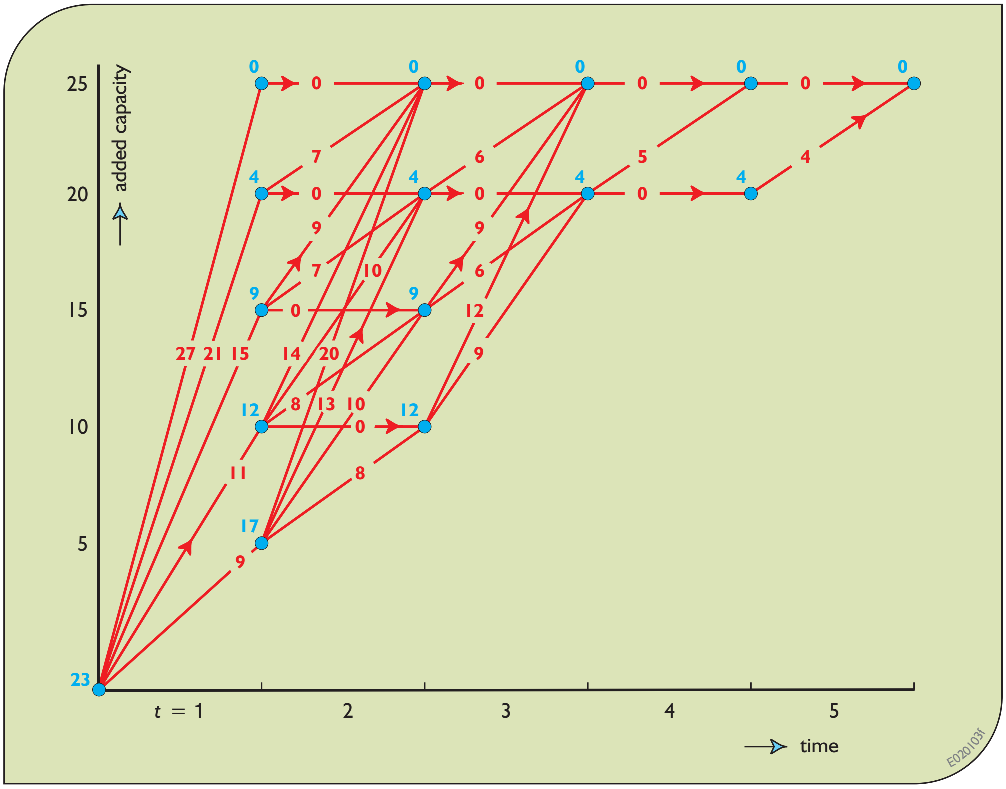


Figure 4.15. A capacity-expansion example, showing the results of a backward-moving dynamic programming algorithm. The numbers next to the nodes are the minimum remaining cost to have the particular capacity required at the end of the planning horizon given the existing capacity of the state.

economic cost estimates can be updated and the planning horizon extended, as necessary, to a period that again does not affect the immediate decision. Note that in the example problem shown in Figures 4.14 and 4.15, the use of 4 periods instead of 5 would have resulted in the same first-period decision. There is no need to extend the analysis to 6 or more periods.

To summarize: What is important to decision-makers now is what additional capacity to add now. While the current period's capacity addition should be based on the best estimates of future costs, interest rates and demands, once a solution is obtained for the capacity expansion required for this and all future periods up to some distant time horizon, one can then ignore all but that first decision, x_1: that is, what to add now. Then just before the beginning of the second period, the forecasting and analysis can be redone with updated data to obtain an updated solution for what capacity to add in Period 2, and so on into the future. Thus, these sequential decision-making dynamic programming models can be designed to be used in a sequential decision-making process.

4.7.2. Reservoir Operation

Reservoir operators need to know how much water to release and when. Reservoirs designed to meet demands for water supplies, recreation, hydropower, the environment and/or flood control need to be operated in ways that meet those demands in the most reliable and effective manner. Since future inflows or storage volumes are uncertain, the challenge, of course, is to determine the best reservoir release or discharge for a variety of possible inflows and storage conditions.

Reservoir release policies are often defined in the form of what are called 'rule curves'. Figure 4.16 illustrates a rule curve for a single reservoir on the Columbia River in the northwestern United States. It combines components of two basic types of release rules. In both of these, the year is divided into various within-year time periods. There is a specified release for each value of storage in each within-year time period. Usually higher storage zones are associated with higher reservoir releases. If the actual storage is relatively low, then less water is

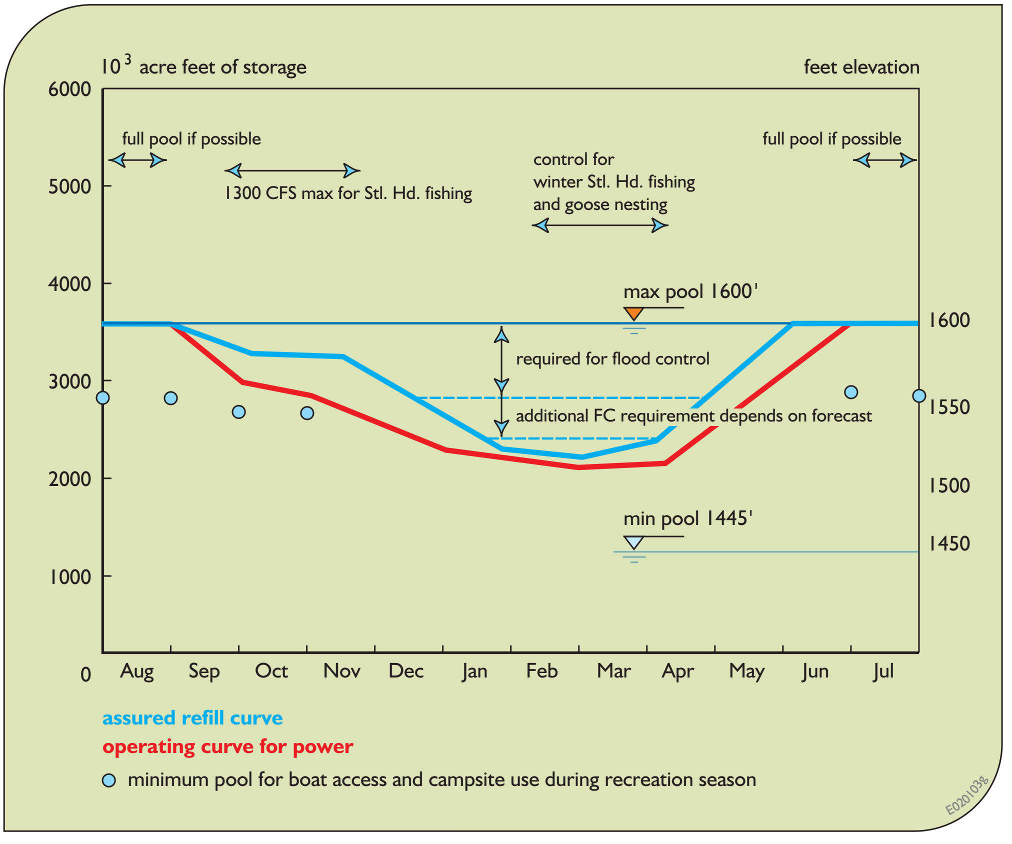


Figure 4.16. An example reservoir rule curve specifying the storage targets and some of the release constraints, given the particular current storage volume and time of year. The release constraints also include the minimum and maximum release rates and the maximum downstream channel rate of flow and depth changes that can occur in each month.

usually released so as to hedge against a continuing water shortage or drought.

Release rules may also specify the desired storage level for the time of year. The operator is to release water as necessary to achieve these target storage levels. Maximum and minimum release constraints might also be specified that may affect how quickly the target storage levels can be met. Some rule curves define multiple target storage levels depending on hydrological (e.g., snow pack) conditions in the upstream watershed, or on the forecast climate conditions as affected by ENSO cycles, solar geomagnetic activity, ocean currents and the like. (There is further discussion of this topic in Appendix C).

Reservoir release rule curves for a year, such as that shown in Figure 4.16, define a policy that does not vary from one year to the next. The actual releases will vary, however, depending on the inflows and storage volumes that actually occur. The releases are often specified independently of future inflow forecasts. They are typically based only on existing storage volumes and within-year periods.

Release rules are typically derived from trial and error simulations. To begin these simulations it is useful to have at least an approximate idea of the expected impact of different alternative policies on various system performance measures or objectives. Policy objectives could be the maximization of expected annual net benefits from downstream releases, reservoir storage volumes, hydroelectric energy and flood control, or the minimization of deviations from particular release, storage volume, hydroelectric energy or flood flow targets or target ranges. Discrete dynamic programming can be used to obtain initial estimates of reservoir operating policies that meet these and other objectives. The results of discrete dynamic programming can be expressed in the form shown in Figure 4.16.

A numerical example

Consider, as a simple example, a reservoir having an active storage capacity of 20 million cubic metres, or for that matter any specified volume units. The active storage volume in the reservoir can vary between 0 and 20. To use discrete dynamic programming, this range of possible storage volumes must be divided into a set of discrete values. These will be the discrete state variable values. In this example let the range of storage volumes be divided into intervals of 5 storage volume units. Hence, the initial storage volume, S_t, can assume values of 0, 5, 10, 15 and 20 for all periods t.

For each period t, let Q_t be the mean inflow, $L_t(S_t, S_{t+1})$ the evaporation and seepage losses that depend on the storage volume in the reservoir, and R_t the release or discharge from the reservoir. Each variable is expressed as volume units for the period t.

Storage volume continuity requires that in each period t the initial active storage volume, S_t, plus the inflow, Q_t, less the losses, $L_t(S_t, S_{t+1})$, and release, R_t, equals the final storage, or equivalently the initial storage, S_{t+1}, in the following period $t + 1$. Hence:

$$S_t + Q_t - R_t - L_t(S_t, S_{t+1}) = S_{t+1} \quad \text{for each period } t. \tag{4.112}$$

To satisfy the requirement (imposed for convenience in this example) that each storage volume variable be a discrete value ranging from 0 to 20 in units of 5, the releases, R_t, must be such that when $Q_t - R_t - L_t(S_t, S_{t+1})$ is added to S_t the resulting value of S_{t+1} is one of the 5 discrete numbers between 0 and 20.

Assume four within-year periods t in each year (kept small for this illustrative example). In these four seasons assume the mean inflows, Q_t, are 24, 12, 6 and 18 respectively. Table 4.6 defines the evaporation and seepage losses based on different discrete combinations of initial and final storage volumes for each within-year period t.

Rounding these losses to the nearest integer value, Table 4.7 shows the net releases associated with initial and final storage volumes. They are computed using Equation 4.112.

The information in Table 4.7 allows us to draw a network representing each of the discrete storage volume states (the nodes), and each of the feasible releases (the links). This network for the four seasons t in the year is illustrated in Figure 4.17.

This reservoir-operating problem is a multistage decision-making problem. As Figure 4.17 illustrates, at the beginning of any season t, the storage volume can be in any of the five discrete states. Given that state, a release decision is to be made. This release will depend on the state: the initial storage volume and the mean inflow, as well as the losses that may be estimated based on the

period	initial storage	final storage 0	5	10	15	20
period $t = 1$	0	0	0.1	0.3	0.4	0.5
	5	0.1	0.2	0.4	0.6	0.8
	10	0.3	0.4	0.6	0.8	1.0
	15	0.4	0.6	0.8	1.0	1.2
	20	0.6	0.8	1.0	1.2	1.4
period $t = 2$	0	0	0.5	0.7	0.8	1.0
	5	0.5	0.7	0.8	1.0	1.2
	10	0.7	0.8	1.0	1.2	1.4
	15	0.8	1.0	1.2	1.4	1.6
	20	1.0	1.2	1.4	1.6	1.8
period $t = 3$	0	0	0.7	0.9	1.0	1.2
	5	0.7	0.9	1.0	1.2	1.4
	10	0.9	1.0	1.2	1.4	1.6
	15	1.0	1.2	1.4	1.6	1.8
	20	1.2	1.4	1.6	1.8	2.0
period $t = 4$	0	0	0.1	0.2	0.3	0.4
	5	0.1	0.2	0.3	0.4	0.5
	10	0.2	0.3	0.4	0.5	0.6
	15	0.3	0.4	0.5	0.6	0.7
	20	0.4	0.5	0.6	0.7	0.8

losses

E020820n

Table 4.6. Evaporation and seepage losses based on initial and final storage volumes for example reservoir operating problem.

inflow / period	initial storage	final storage 0	5	10	15	20
inflow $Q_1 = 24$, period $t = 1$	0	24	19	14	8	3
	5	29	24	18	13	8
	10	34	29	23	18	13
	15	39	33	28	23	18
	20	43	38	33	28	23
inflow $Q_2 = 12$, period $t = 2$	0	12	7	1	–	–
	5	17	11	6	1	–
	10	21	[illegible]	11	6	1
	15	26	21	16	11	5
	20	31	26	21	15	10
inflow $Q_3 = 6$, period $t = 3$	0	6	0	–	–	–
	5	10	5	0	–	–
	10	15	10	5	0	–
	15	20	15	10	4	–
	20	25	20	14	9	4
inflow $Q_4 = 18$, period $t = 4$	0	18	13	8	3	–
	5	23	18	13	8	3
	10	28	23	18	13	7
	15	33	28	23	17	12
	20	38	33	27	22	17

releases

E020820o

Table 4.7. Discrete releases associated with initial and final storage volumes.

initial and final storage volumes, as in Table 4.6. The release will also depend on what is to be accomplished – that is, the objectives to be satisfied.

For this example, assume there are various targets that water users would like to achieve. Downstream water users want reservoir operators to meet their flow targets. Individuals who use the lake for recreation want the reservoir operators to meet storage volume or storage level targets. Finally, individuals living on the downstream floodplain want the reservoir operators to provide storage capacity for flood protection. Table 4.8 identifies these different targets that are to be met, if possible, for the duration of each season t.

Clearly, it will not be possible to meet all these storage volume and release targets in all four seasons, given inflows of 24, 12, 6 and 18, respectively. Hence, the objective in this example will be to do the best one can: to minimize a weighted sum of squared deviations from each of these targets. The weights reflect the relative importance of meeting each target in each season t. Target deviations are squared to reflect the fact that the marginal

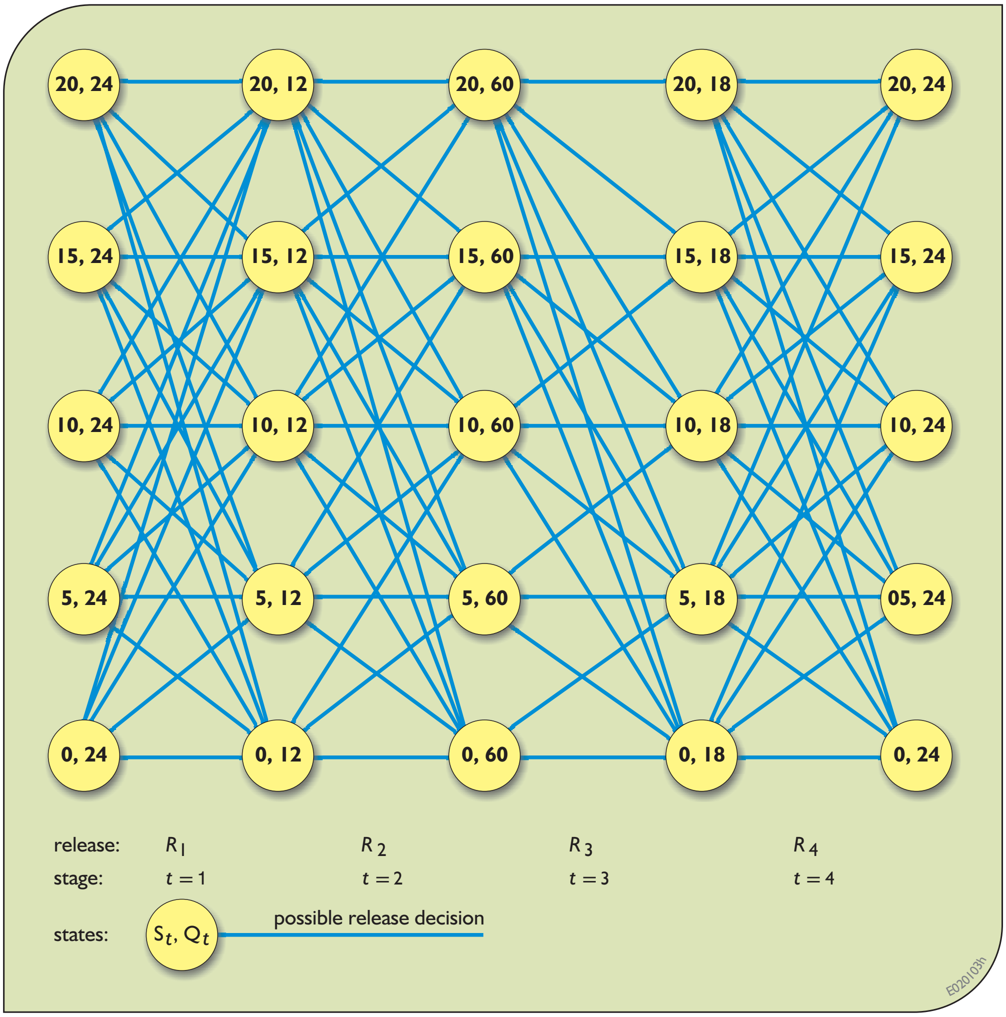


Figure 4.17. Network representation of the four-season reservoir release problem. Given any initial storage volume S_t at the beginning of a season t, and an expected inflow of Q_t during season t, the links indicate the possible release decisions corresponding to those in Table 4.7.

E020820p

period or season t	storage targets $TS^R{}_t$, $TS^F{}_t$		release target TR_t
1	15	flood control	≥ 10
2	20	recreation	≥ 15
3	20	recreation	≥ 20
4			≥ 15

Table 4.8. Storage volume and release targets for the example reservoir operation problem.

'losses' associated with deviations increase with increasing deviations. Small deviations are not as serious as larger deviations, and it is better to have numerous small deviations rather than a single larger one.

During the recreation season (Periods 2 and 3), deviations below or above the recreation storage lake volume targets are damaging. During the flood season (Period 1), any storage volume in excess of the flood control storage targets of 15 reduces the flood storage capacity. Deviations below that flood control target are not penalized. Flood control and recreation storage targets during each season t apply throughout the season, thus they apply to the initial storage S_t as well as to the final storage S_{t+1}.

The objective is to minimize the sum of total weighted squared deviations, TSD_t, over all seasons t from now on into the future:

$$\text{Minimize} \sum_t TSD_t \tag{4.113}$$

where

$$TSD_t = ws_t[(TS_t^R - S_t)^2 + (TS_t^R - S_{t+1})^2] + wfs_t[(ES_t)^2 + (ES_{t+1})^2] + wr_t[DR_t^2] \tag{4.114}$$

In the above equation, when $t = 4$, the last period of the year, $t + 1 = 1$, is the first period in the following year. Each ES_t is the storage volume in excess of the flood storage target volume, TS_t^F. Each DR_t is the difference between the actual release, R_t, and the target release TR_t, when the release is less than the target. The excess storage, ES_t, at the beginning of each season t can be defined by the constraint:

$$S_t \leq TS_t^F + ES_t \tag{4.115}$$

for periods $t = 1$ and 2, and the deficit release, DR_t, during period t can be defined by the constraint:

$$R_t \geq TR_t - DR_t \tag{4.116}$$

This constraint applies for all periods t.

The first component of the right side of Equation 4.114 defines the weighted squared deviations from a recreation storage target, TS_t^R, at the beginning and end of season t. In this example the recreation season is during Periods 2 and 3. The weights ws_t associated with the recreation component of the objective are 1 in Periods 2 and 3. In Periods 1 and 4 the weights ws_t are 0.

The second component of Equation 4.114 is for flood control. It defines the weighted squared deviations associated with storage volumes in excess of the flood control target volume, TS_t^F, at the beginning and end of the flood season, period $t = 1$. In this example, the weights wfs_t are 1 for Period 1 and 0 for Periods 2, 3 and 4. Note the conflict between flood control and recreation at the end of Period 1 or equivalently at the beginning of Period 2.

Finally, the last component of Equation 4.114 defines the weighted squared deficit deviations from a release target, TR_t, In this example all release weights wr_t equal 1.

Associated with each link in Figure 4.17 is the release, R_t, as defined in Table 4.7. Also associated with each link is the sum of weighted squared deviations, TSD_t, that result from the particular initial and final storage volumes and the storage volume and release targets identified in Table 4.8. They are computed using Equation 4.114, with the releases defined in Table 4.7 and targets defined in Table 4.8, for each feasible combination of initial and final storage volumes, S_t and S_{t+1}, for each of the four seasons in a year. These computed weighted squared deviations for each link are shown in Table 4.9.

The goal in this example problem is to find the path through a multi-year network – each year of which is as shown in Figure 4.17 – that minimizes the sum of the squared deviations associated with each of the path's links. Again, each link's weighted squared deviations are given in Table 4.9. Of interest is the best path into the future from any of the nodes or states (discrete storage volumes) that the system could be in at the beginning of any season t.

These paths can be found using the backward-moving solution procedure of discrete dynamic programming. This procedure begins at any arbitrarily selected time period or season when the reservoir presumably produces no further benefits to anyone (and it doesn't matter when that time is – just pick any time) and proceeds backward, from right to left one stage (i.e., one time period) at a time, towards the present. At each node (representing a discrete storage volume S_t and inflow Q_t), we can calculate the release or final storage volume in that period that minimizes the remaining sum of weighted squared deviations for all remaining seasons. Denote this minimum sum of weighted squared deviations for all n remaining seasons t as $F_t^n(S_t, Q_t)$. This value is dependent on the state (S_t, Q_t), and stage, t, and the number n of remaining seasons. It is not a function of the decision R_t or S_{t+1}.

This minimum sum of weighted squared deviations for all n remaining seasons t is equal to:

$$F_t^n(S_t, Q_t) = \min \sum_t^n TSD_t(S_t, R_t, S_{t+1}) \text{ over all feasible values of } R_t \tag{4.117}$$

initial storage	final storage 0	5	10	15	20
$TR_1 \geq 10,\ TS^F_t \leq 15$, period t = 1					
0	0	0	0	4	74
5	0	0	0	0	29
10	0	0	0	0	25
15	0	0	0	0	25
20	25	25	25	25	50
$TS_2 = 20,\ TR_2 \geq 15$, period t = 2					
0	809	689	696	---	---
5	625	466	406	446	---
10	500	325	216	206	296
15	425	250	125	66	125
20	400	225	100	25	25
$TS_3 = 20,\ TR_3 \geq 20$, period t = 3					
0	996	1025	---	---	---
5	725	675	725	---	---
10	525	425	425	525	---
15	425	275	225	306	---
20	400	225	136	146	256
$TR_4 \geq 15$, period t = 4					
0	0	4	49	144	---
5	0	0	4	49	144
10	0	0	0	4	64
15	0	0	0	0	9
20	0	0	0	0	0

TSD

E020820q

Table 4.9. Total sum of squared deviations, TSD_t, associated with initial and final storage volumes. These are calculated using Equations 4.114 through 4.116.

where

$$S_{t+1} = S_t + Q_t - R_t - L_t(S_t, S_{t+1}) \tag{4.118}$$

and

$$S_t \leq K, \text{ the capacity of the reservoir} \tag{4.119}$$

The policy we want to derive is called a *steady-state policy*. Such a policy assumes the reservoir will be operating for a relatively long time with the same objectives. We can find this steady-state policy by first assuming that at some time all future benefits, losses or penalties, $F^0_t(S_t, Q_t)$, will be 0. We can begin in that season t and work backwards towards the present, moving left through the network one season t at a time. We can continue for multiple years until the annual policy begins repeating itself each year. In other words, when the optimal R_t associated with a particular state (S_t, Q_t) is the same in two or more successive years, and this applies for all states (S_t, Q_t) in each season t, a steady-state policy has probably been obtained (a more definitive test of whether or not a steady-state policy has been reached will be discussed later). A steady-state policy will occur if the inflows, Q_t, and objectives, $TSD_t(S_t, R_t, S_{t+1})$, remain the same from year to year. This steady-state policy is independent of the assumption that the operation will end at some point.

To find the steady-state operating policy for this example problem, assume the operation ends in some distant year at the end of Season 4 (the right-hand side nodes in Figure 4.17). At the end of this season the number of remaining seasons, n, equals 0. The values of the remaining minimum sums of weighted squared deviations, $F^0_1(S_1, Q_1)$ associated with each state (S_1, Q_1), i.e., each node, equal 0. Now we can begin the process of finding the best releases R_t in each successive season, moving backward to the beginning of stage $t = 4$, then stage $t = 3$, then to $t = 2$, and then to $t = 1$, and then to $t = 4$ of the preceding year, and so on, each move to the left increasing the number n remaining seasons by one.

At each stage, or season, we can compute the release R_t or equivalently the final storage volume S_{t+1}, that minimizes

$$F^n_t(S_t, Q_t) = \text{Minimum}\{TSD_t(S_t, R_t, S_{t+1}) + F^{n-1}_{t+1}(S_{t+1}, Q_{t+1})\} \quad \text{for all } 0 \leq S_t \leq 20 \tag{4.120}$$

The decision-variable can be either the release, R_t, or the final storage volume, S_{t+1}. If the decision-variable is the release, then the constraints on that release R_t are:

$$R_t \leq S_t + Q_t - L_t(S_t, S_{t+1}) \tag{4.121}$$

$$R_t \geq S_t + Q_t - L_t(S_t, S_{t+1}) - 20 \text{ (the capacity)} \tag{4.122}$$

and

$$S_{t+1} = S_t + Q_t - R_t - L_t(S_t, S_{t+1}) \tag{4.123}$$

If the decision-variable is the final storage volume, the constraints on that final storage volume S_{t+1} are:

$$0 \leq S_{t+1} \leq 20 \tag{4.124}$$

$$S_{t+1} \leq S_t + Q_t - L_t(S_t, S_{t+1}) \tag{4.125}$$

and

$$R_t = S_t + Q_t - S_{t+1} - L_t(S_t, S_{t+1}) \tag{4.126}$$

Note that if the decision-variable is S_{t+1} in season t, this decision becomes the state variable in season $t + 1$. In both cases, the storage volumes in each season are limited to discrete values 0, 5, 10, 15 and 20.

Tables 4.10 through 4.19 show the values obtained from solving the recursive equations for 10 successive seasons or stages (2.5 years). Each table represents a stage or season t, beginning with Table 4.10 at $t = 4$ and the number of remaining seasons $n = 1$. The data in each table are obtained from Tables 4.7 and 4.9. The last two columns of each table represent the best release and final storage volume decision(s) associated with the state (initial storage volume and inflow).

Note that the policy defining the release or final storage for each discrete initial storage volume in season $t = 3$ in Table 4.12 is the same as in Table 4.16, and similarly for season $t = 4$ in Tables 4.13 and 4.17, and for season $t = 1$ in Tables 4.14 and 4.18, and finally for season $t = 2$ in Tables 4.15 and 4.19. The policy differs over each state, and over each different season, but not from year to year for any specified state and season. We have reached a steady-state policy. If we kept on computing the release and final storage policies for preceding seasons, we would

Table 4.10. Calculation of minimum squared deviations associated with various discrete storage states in season $t = 4$ with only $n = 1$ season remaining for reservoir operation.

period $t = 4$ $n = 1$ $Q_4 = 18$

$F_4^1(S_4, Q_4) = \min TSD_4$

initial storage S_4	TSD_4 final storage S_1: 0	5	10	15	20	F_4^1	R_4	S_1
0	0	4	49	144	-	0	18	0
5	0	0	4	49	144	0	18-23	0-5
10	0	0	0	4	64	0	18-28	0-10
15	0	0	0	0	9	0	17-33	0-15
20	0	0	0	0	0	0	17-38	0-20

E020827a

Table 4.11. Calculation of minimum squared deviations associated with various discrete storage states in season $t = 3$ with $n = 2$ seasons remaining for reservoir operation.

period $t = 3$ $n = 2$ $Q_3 = 6$

$F_3^2(S_3, Q_3) = \min \{TSD_3(S_3, R_3, S_4) + F_4^1(S_4, Q_4)\}$

initial storage S_3	$TSD_3 + F_4^1(S_4, Q_4)$ final storage S_4: 0	5	10	15	20	F_3^2	R_3	S_4
0	996	1025	-	-	-	996	6	0
5	725	675	725	-	-	675	5	5
10	525	425	425	525	-	425	5-10	5-10
15	425	275	225	306	-	225	10	10
20	400	225	136	146	256	136	14	10

E020827b

period t = 2 $n = 3$ $Q_2 = 12$

$F_2^3(S_2, Q_2) = \min\{TSD_2(S_2, R_2, S_3) + F_3^2(S_3, Q_3)\}$

initial storage S_2	$TSD_2 + F_3^2(S_3, Q_3)$ final storage S_3: 0	5	10	15	20	F_2^3	R_2	S_3
0	809+996	689+675	696+425	-	-	1121	1	10
5	625+996	466+675	406+425	446+225	-	671	1	15
10	500+996	325+675	216+425	206+225	296+136	431	6	15
15	425+996	250+675	125+425	66+225	125+136	261	5	20
20	400+996	225+675	100+425	25+225	25+136	161	10	20

E020827c

Table 4.12. Calculation of minimum squared deviations associated with various discrete storage states in season $t = 2$ with $n = 3$ seasons remaining for reservoir operation.

period t = 1 $n = 4$ $Q_1 = 24$

$F_1^4(S_1, Q_1) = \min\{TSD_1(S_1, R_1, S_2) + F_2^3(S_2, Q_2)\}$

initial storage S_1	$TSD_1 + F_2^3(S_2, Q_2)$ final storage S_2: 0	5	10	15	20	F_1^4	R_1	S_2
0	0+1121	0+671	0+431	4+261	74+161	235	3	20
5	0+1121	0+671	0+431	0+261	29+161	190	8	20
10	0+1121	0+671	0+431	0+261	25+161	186	13	20
15	0+1121	0+671	0+431	0+261	25+161	186	18	20
20	25+1121	25+671	25+431	25+261	50+161	211	23	20

E020827d

Table 4.13. Calculation of minimum squared deviations associated with various discrete storage states in season $t = 1$ with $n = 4$ seasons remaining for reservoir operation.

period t = 4 $n = 5$ $Q_4 = 18$

$F_4^5(S_4, Q_4) = \min\{TSD_4(S_4, R_4, S_1) + F_1^4(S_1, Q_1)\}$

initial storage S_4	$TSD_4 + F_1^4(S_1, Q_1)$ final storage S_1: 0	5	10	15	20	F_4^5	R_4	S_1
0	0+235	4+190	49+186	144+186	-	194	13	5
5	0+235	0+190	4+186	49+186	144+211	190	13-18	5-10
10	0+235	0+190	0+186	4+186	64+211	186	18	10
15	0+235	0+190	0+186	0+186	9+211	186	17-23	10-15
20	0+235	0+190	0+186	0+186	0+211	186	22-27	10-15

E020827e

Table 4.14. Calculation of minimum squared deviations associated with various discrete storage states in season $t = 4$ with $n = 5$ seasons remaining for reservoir operation.

Table 4.15. Calculation of minimum squared deviations associated with various discrete storage states in season $t = 3$ with $n = 6$ seasons remaining for reservoir operation.

period t = 3 $\quad n = 6 \quad Q_3 = 6$

$F_3^6(S_3, Q_3) = \min\{TSD_3(S_3, R_3, S_4) + F_4^5(S_4, Q_4)\}$

initial storage S_3	$TSD_3 + F_4^5(S_4, Q_4)$ final storage S_4: 0	5	10	15	20	F_3^6	R_3	S_4
0	996+194	1025+190	-	-	-	1190	6	0
5	725+194	675+190	725+186	-	-	865	5	5
10	525+194	425+190	425+186	525+186	-	611	5	10
15	425+194	275+190	225+186	306+186	-	411	10	10
20	400+194	225+190	136+186	146+186	256+186	322	14	10

E020827f

Table 4.16. Calculation of minimum squared deviations associated with various discrete storage states in season $t = 2$ with $n = 7$ seasons remaining for reservoir operation.

period t = 2 $\quad n = 7 \quad Q_2 = 12$

$F_2^7(S_2, Q_2) = \min\{TSD_2(S_2, R_2, S_3) + F_3^6(S_3, Q_3)\}$

initial storage S_2	$TSD_2 + F_3^6(S_3, Q_3)$ final storage S_3: 0	5	10	15	20	F_2^7	R_2	S_3
0	809+1190	689+865	696+611	-	-	1307	1	10
5	625+1190	466+865	406+611	446+411	-	857	1	15
10	500+1190	325+865	216+611	206+411	296+322	617	6	15
15	425+1190	250+865	125+611	66+411	125+322	447	5	20
20	400+1190	225+865	100+611	25+411	25+322	347	10	20

E020827g

Table 4.17. Calculation of minimum squared deviations associated with various discrete storage states in season $t = 1$ with $n = 8$ seasons remaining for reservoir operation.

period t = 1 $\quad n = 8 \quad Q_2 = 24$

$F_1^8(S_1, Q_1) = \min\{TSD_1(S_1, R_1, S_2) + F_2^7(S_2, Q_2)\}$

initial storage S_1	$TSD_1 + F_2^7(S_2, Q_2)$ final storage S_2: 0	5	10	15	20	F_1^8	R_1	S_2
0	0+1307	0+857	0+617	4+447	74+347	421	3	20
5	0+1307	0+857	0+617	0+447	29+347	376	8	20
10	0+1307	0+857	0+617	0+447	25+347	372	13	20
15	0+1307	0+857	0+617	0+447	25+347	372	18	20
20	25+1307	25+857	25+617	25+447	50+347	397	23	20

E020827h

Table 4.18. Calculation of minimum squared deviations associated with various discrete storage states in season $t = 4$ with $n = 9$ seasons remaining for reservoir operation.

period $t = 4$ $n = 9$ $Q_4 = 18$

$F_4^9(S_4, Q_4) = \min \{TSD_4(S_4, R_4, S_1) + F_1^8(S_1, Q_1)\}$

initial storage S_4	$TSD_4 + F_1^8(S_1, Q_1)$ final storage S_1					F_4^9	R_4	S_1
	0	5	10	15	20			
0	0+421	4+376	49+372	144+372	-	380	13	5
5	0+421	0+376	4+372	49+372	144+397	376	13-18	5-10
10	0+421	0+376	0+372	4+372	64+397	372	18	10
15	0+421	0+376	0+372	0+372	9+397	372	17-23	10-15
20	0+421	0+376	0+372	0+372	0+397	372	22-27	10-15

Table 4.19. Calculation of minimum squared deviations associated with various discrete storage states in season $t = 3$ with $n = 10$ seasons remaining for reservoir operation.

period $t = 3$ $n = 10$ $Q_3 = 6$

$F_3^{10}(S_3, Q_3) = \min \{TSD_3(S_3, R_3, S_4) + F_4^9(S_4, Q_4)\}$

initial storage S_3	$TSD_3 + F_4^9(S_4, Q_4)$ final storage S_4					F_3^{10}	R_3	S_4
	0	5	10	15	20			
0	996+380	1025+376	-	-	-	1376	6	0
5	725+380	675+376	725+372	-	-	1051	5	5
10	525+380	425+376	425+372	525+372	-	797	5	10
15	425+380	275+376	225+372	306+372	-	597	10	10
20	400+380	225+376	136+372	146+372	256+372	508	14	10

get the same policy as that found for the same season in the following year. The policy is dependent on the state – the initial storage volume in this case – and on the season t, but not on the year. This policy as defined in Tables 4.16–4.19 is summarized in Table 4.20.

This policy can be defined as a rule curve, as shown in Figure 4.18. It provides a first approximation of a reservoir release rule curve that one can improve upon using simulation.

Table 4.20 and Figure 4.18 define a policy that can be implemented for any initial storage volume condition at the beginning of any season t. This can be simulated under different flow patterns to determine just how well it satisfies the overall objective of minimizing the weighted sum of squared deviations from desired, but conflicting, storage and release targets. There are other performance criteria that may also be evaluated using simulation, such as measures of reliability, resilience, and vulnerability (Chapter 10).

Assuming the inflows that were used to derive this policy actually occurred each year, we can simulate the derived sequential steady-state policy to find the storage volumes and releases that would occur in each period, year after year, once a repetitive steady-state condition were reached. This is done in Table 4.21 for an arbitrary initial storage volume of 20 in season $t = 1$. You can try other initial conditions to verify that it requires only two years at most to reach a repetitive steady-state policy.

As shown in Table 4.21, if the inflows were repetitive and the optimal policy was followed, the initial storage

Table 4.20. The discrete steady-state reservoir operating policy as computed for this example problem in Tables 4.16 to 4.19.

initial storage S	release season t 1	2	3	4	final storage volume season t 1	2	3	4
0	3	1	6	13	20	10	0	5
5	8	1	5	13-18	20	15	5	5-10
10	13	6	5	18	15	10	10	10
15	18	5	10	17-23	20	20	10	10-15
20	23	10	14	22-27	20	20	10	10-15

E020827n

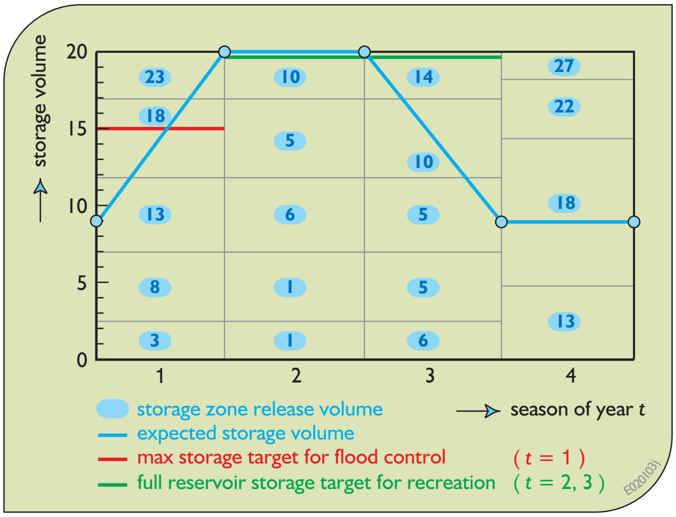


Figure 4.18. Reservoir rule curve based on policy defined in Table 4.20. Each season is divided into storage volume zones. The releases associated with each storage volume zone are specified. Also shown are the storage volumes that would result if in each year the actual inflows equalled the inflows used to derive this rule curve.

volumes and releases would begin to repeat themselves once a steady-state condition has been reached. Once reached, the storage volumes and releases will be the same each year (since the inflows are the same). These storage volumes are denoted as a blue line on the rule curve shown in Figure 4.18. The annual total squared deviations will also be the same each year. As seen in Table 4.21, this annual minimum weighted sum of squared deviations for this example equals 186. This is what would be observed if the inflows assumed for this analysis repeated themselves.

Note from Tables 4.12 – 4.15 and 4.16 – 4.19 that once the steady-state sequential policy has been reached for any specified storage volume, S_t, and season t, the annual difference of the accumulated minimum sum of squared deviations, $F_t^n(S_t, Q_t)$, equals a constant, namely the annual value of the objective function. In this case that constant is 186.

$$F_t^{n+4}(S_t, Q_t) - F_t^n(S_t, Q_t) = 186 \quad \text{for all } S_t, Q_t \text{ and } t. \tag{4.127}$$

This condition indicates a steady-state policy has been achieved.

This policy in Table 4.21 applies only for the assumed inflows in each season. It does not define what to do if the initial storage volumes or inflows differ from those for which the policy is defined. Initial storage volumes and inflows can and will vary from those specified in the solution of any deterministic model. One fact is certain: no matter what inflows are assumed in any model, the actual inflows will always differ. Hence, a policy as defined in Table 4.20 and Figure 4.18 is much more useful than that in Table 4.21. But as for any output from a relatively simple optimization model, this policy should be simulated, evaluated and further refined in an effort to identify the policy that best meets the operating policy objectives.

4.8. General Comments on Dynamic Programming

Before ending this discussion of dynamic programming methods and its applicability for analysing water resources planning problems, we should examine a major

year	season t	S_t +	Q_t −	R_t −	L_t =	S_{t+1}	TSD_t
1	1	20	24	23	1	20	
1	2	20	12	10	2	20	
1	3	20	6	14	2	10	
1	4	10	18	18	0	10	

year	season t	S_t +	Q_t −	R_t −	L_t =	S_{t+1}	TSD_t
2	1	10	24	14	0	20	25
2	2	20	12	10	2	20	25
2	3	20	6	14	2	10	136
2	4	10	18	18	0	10	0

year	season t	S_t +	Q_t −	R_t −	L_t =	S_{t+1}	TSD_t
3	1	10	etc ...	repeating ...			
		total sum of squared deviations, **TSD**, =					186

E020827p

Table 4.21. A simulation of the derived operating policy in Table 4.20. The storage volumes and releases in each period *t* will repeat themselves each year, after the first year. The annual total squared deviations, TSD_t for the specific initial and final storage volumes and release conditions are obtained from Table 4.9.

assumption that has been made in each of the applications presented. The first is that the net benefits or costs or other objective values resulting from each decision at each stage of the problem are dependent only on the state variable values in each stage. They are independent of decisions made at other stages. If the returns at any stage are dependent on the decisions made at other stages in a way not captured by the state variables. For example, if the release and storage targets or the reservoir capacity were unknown in the previous reservoir operating policy problem, then dynamic programming, with some exceptions, becomes much more difficult to apply. Dynamic programming models can be applied to design problems, such as the capacity-expansion problem or a reservoir storage capacity–yield relationship as will be discussed later, or to operating problems, such as the water allocation and reservoir operation problems, but rarely to problems having both unknown design and operating policy decision-variables. While there are some tricks that may allow dynamic programming to be used to find the best solutions to both design and operating problems encountered in water resources planning and management studies, other optimization methods, perhaps combined with dynamic programming where appropriate, are often more useful.

5. Linear Programming

If the objective function and constraints of an optimization model are all linear, there are many readily available computer programs one can use to find their solutions. These programs are very powerful, and unlike many other optimization methods, they can be applied successfully to very large optimization problems. Many water resources problems contain many variables and constraints, too many to be easily solved using non-linear or dynamic programming methods. Linear programming procedures or algorithms for solving linear optimization models are often the most efficient ways to find solutions to such problems.

Because of the availability of computer programs that can solve linear programming problems, linear programming is arguably the most popular and commonly applied optimization algorithm. It is used to identify and evaluate alternative plans, designs and management policies in agriculture, business, commerce, education, engineering, finance, the civil and military branches of government, and many other fields.

Many models of complex water resources systems are, or can be made, linear. Many are also very large. The number of variables and constraints simply defining mass balances and capacity limitations alone at many river basin sites and for numerous time periods can become so big as to preclude the practical use of most other optimization methods. Because of the power and availability of computer programs that can solve large linear programming problems, a variety of methods have been developed to approximate non-linear (especially separable) functions with linear ones just so linear programming can be used to solve various otherwise non-linear problems. Some of these methods will be described shortly.

In spite of its power and popularity, for most real-world water resources planning and management problems, linear programming, like the other optimization methods already discussed in this chapter, is best viewed as a preliminary screening tool. Its value is more for reducing the number of alternatives for further more detailed simulations than for finding the best decision. This is not just because approximation methods may have been used to convert non-linear functions to linear ones, but more likely because it is difficult to incorporate all the complexity of the system and all the objectives considered important to all stakeholders into the linear model. Nevertheless, linear programming, like other optimization methods, can provide initial designs and operating policy information that simulation models require before they can simulate those designs and operating policies.

Equations 4.45 and 4.46 define the general structure of any constrained optimization problem. If the objective function $F(\boldsymbol{X})$ of the vector $\boldsymbol{X}$ of decision-variables x_j is linear and if all the constraints $g_i(\boldsymbol{X})$ in Equation 4.46 are linear, then the model becomes a linear programming model. The general structure of a linear programming model is:

$$\text{Maximize or minimize } \sum_{j}^{n} P_j \, x_j \tag{4.128}$$

$$\text{Subject to: } \sum_{j}^{n} a_{ij} \, x_j \leq b_i \quad \text{for } i = 1, 2, 3, \ldots, m \tag{4.129}$$

$$x_j \geq 0 \quad \text{for all } j = 1, 2, 3, \ldots, n. \tag{4.130}$$

If any model fits this general form, where the constraints can be any combination of equalities (=) and inequalities (≥ or ≤), then a large variety of linear programming computer programs can be used to find the 'optimal' values of all the unknown decision-variables x_j. With some exceptions, variable non-negativity is enforced within the solution algorithms of most commercial linear programming programs, eliminating the need to have to specify these conditions in any particular application.

Users of linear programming need to know how to construct linear models and how to use the computer programs that are available for solving them. They do not have to understand all the mathematical details of the solution procedure incorporated in the linear programming codes. But users of linear programming computer programs should understand what the solution procedure does and what the computer program output means. To begin this discussion of these topics, consider some simple examples of linear programming models.

5.1. Reservoir Storage Capacity–Yield Models

Linear programming can be used to define storage capacity–yield functions for a single or multiple reservoirs. A storage capacity–yield function defines the maximum constant 'dependable' reservoir release or yield that will be available, at a given level of reliability, during each period of operation, as a function of the active storage volume capacity. The yield from any reservoir or group of reservoirs will depend on the active storage capacity of each reservoir and the water that flows into each reservoir, its inflow. Figure 4.19 illustrates two typical storage–yield functions for a single reservoir.

To describe what a yield is and how it can be increased, consider a sequence of 5 annual flows, say 2, 4, 1, 5 and 3, at a site in an unregulated stream. Based on this admittedly very limited record of flows, the minimum (historically) 'dependable' annual flow yield of the stream at that site is 1, the minimum observed flow. Assuming the flow record is representative of what future flows might be, a discharge of 1 can be 'guaranteed' in each of non-zero the five time-periods of record. (In reality, that or any non-zero yield will have a reliability less than 1, as will be considered in Chapter 11).

If a reservoir having an active storage capacity of 1 is built, it could store 1 volume unit of flow when the flow is equal to or greater than 2, and then release it along with the natural flow when the natural flow is 1, increasing the

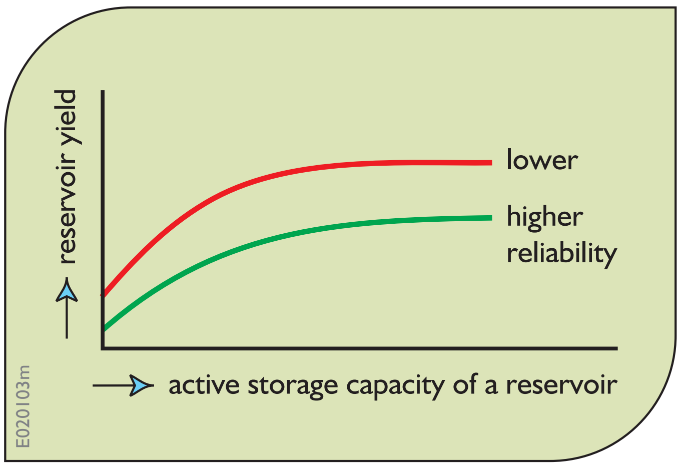


Figure 4.19. Two storage–yield functions for a single reservoir defining the maximum minimum dependable release. These functions can be defined for varying levels of yield reliability.

minimum dependable flow to 2 units in each year. Storing 2 units when the flow is 5, releasing 1 and the natural flow when that natural flow is 2, and storing 1 when the flow is 4, and then releasing the stored 2 units along with the natural flow when the natural flow is 1, will permit a yield of 3 in each time period with 2 units of active capacity. This is the maximum annual yield that is possible at this site, again based on these five annual inflows and their sequence. The maximum annual yield cannot exceed the mean annual flow, which in this example is 3. Hence, the storage capacity–yield function equals 1 when the active capacity is 0, 2 when the active capacity is 1, and 3 when the active capacity is 2. The annual yield remains at 3 for any active storage capacity in excess of 2.

This storage–yield function is dependent not only on the natural unregulated annual flows but also on their sequence. For example if the sequence of the same 5 annual flows were 5, 2, 1, 3, 4, the needed active storage capacity is 3 instead of 2 volume units as before to obtain a dependable flow or yield of 3 volume units. In spite of these limitations of storage capacity–yield functions, historical records are still typically used to derive them. (Ways of augmenting the historical flow record are discussed in Chapter 7.)

There are many methods available for deriving storage–yield functions. One very versatile method, especially for multiple reservoir systems, uses linear programming. Others are discussed in Chapter 11.

To illustrate a storage capacity–yield model, consider a single reservoir that must provide a minimum release or yield Y in each period t. Assume a record of known (historical or synthetic) streamflows at the reservoir site is available. The problem is to find the maximum uniform yield Y obtainable from a given active storage capacity. The objective is to

$$\text{maximize } Y \tag{4.131}$$

This maximum yield is constrained by the water available in each period, and by the reservoir capacity. Two sets of constraints are needed to define the relationships among the inflows, the reservoir storage volumes, the yields, any excess release, and the reservoir capacity. The first set of continuity equations equate the unknown final reservoir storage volume S_{t+1} in period t to the unknown initial reservoir storage volume S_t plus the known inflow Q_t, minus the unknown yield Y and excess release, R_t, if any, in period t. (Losses are being ignored in this example.)

$$S_t + Q_t - Y - R_t = S_{t+1}$$
$$\text{for each period } t = 1, 2, 3, \dots, T. \quad T+1 = 1 \tag{4.132}$$

If, as indicated in Equation 4.132, one assumes that Period 1 follows the last Period T, it is not necessary to specify the value of the initial storage volume S_1 and/or final storage volume S_{T+1}. They are set equal to each other and that variable value remains unknown. The resulting 'steady-state' solution is based on the inflow sequence that is assumed to repeat itself.

The second set of required constraints ensures that the reservoir storage volumes S_t at the beginning of each period t are no greater than the active reservoir capacity K.

$$S_t \le K \quad t = 1, 2, 3, \dots, T \tag{4.133}$$

To derive a storage–yield function, the model defined by Equations 4.131, 4.132 and 4.133 must be solved for various assumed values of capacity K. Only the inflow values Q_t and reservoir active storage capacity K are known. All other storage, release and yield variables are unknown. Linear programming will be able to find their optimal values. Clearly, the upper bound on the yield regardless of reservoir capacity will equal the mean inflow (less any losses if they were included).

Alternatively, one can solve a number of linear programming models that minimize an unknown storage capacity K needed to achieve various specified yields Y. The resulting storage–yield function will be same; the minimum capacity needed to achieve a specified yield will be the same as the maximum yield obtainable from the corresponding specified capacity K. However, the specified yield Y cannot exceed the mean inflow. If it does, there will be no feasible solution to the linear programming model.

Box 4.1 illustrates an example storage–yield model and its solution to find the storage–yield function. For this problem, and others in this chapter, the program LINGO (www.lindo.com) is used.

Before moving to another application of linear programming, consider how this storage–yield problem, Equations 4.131–4.133, can be formulated as a discrete dynamic programming model. The use of discrete dynamic programming is clearly not the most efficient way to define a storage–yield function but the problem of

Box 4.1. Example storage capacity–yield model and its solution from LINGO

E020903a

```
! Reservoir Storage-Yield Model:
Define St as the initial active res. storage, period t,
Y as the reliable yield in each period t,
Rt as the excess release from the res., period t,
Qt as the known inflow volume to the res., period t
K as the reservoir active storage volume capacity.
;
Max = Y ; !Applies to Model 1. Must be omitted for Model 2;
Min = K ; !Applies to Model 2. Must be omitted for Model 1;
!
Subject to:
Mass balance constraints for each of 5 periods t.
;
S1 + Q1 - Y - R1 = S2;
S2 + Q2 - Y - R2 = S3;
S3 + Q3 - Y - R3 = S4;
S4 + Q4 - Y - R4 = S5;
S5 + Q5 - Y - R5 = S1; ! assumes a steady -state condition;
!
Capacity constraints on storage volumes.
;
S1 < K; S2 < K; S3 < K; S4 < K; S5 < K;
Data:
Q1 = 10; Q2 = 5; Q3 = 30; Q4 = 20; Q5 = 15;
!Note mean = 16;
K = ? ;  ! Use for Model 1 only. Allows user to enter
any value of K during model run.;
Y = ? ;  ! Use for Model 2 only. Allows user to enter
any value of Y during model run.
Enddata
```

model solutions for specified values of *K* in model 1 or values of *Y* in model 2

K	*Y*	*S*1	*S*2	*S*3	*S*4	*S*5	*R*1	*R*2	*R*3	*R*4	*R*5
0	5	0	0	0	0	0	5	0	25	15	10
5	10	5	5	0	5	5	0	0	15	10	5
10	12.5	10	7.5	0	2.5	10	0	0	15	0	2.5
15	15	10	10	0	15	15	0	0	0	5	0
18	16	17	11	0	14	18	0	0	0	0	0

finding a storage–yield function provides a good exercise in dynamic programming. The dynamic programming network has the same form as shown in Figure 4.19, where each node is a discrete storage and inflow state, and the links represent releases. Let $F_t^n(S_t)$ be the maximum yield obtained given a storage volume of S_t in period t of a year with n periods remaining of reservoir operation. For initial conditions, assume all values of $F_t^0(S_t)$ for some final period t with no more periods n remaining equal a large number that exceeds the mean annual inflow. Then for the set of feasible discrete total releases R_t:

$$F_t^n(S_t) = \max\{\min[R_t, F_{t+1}^{n-1}(S_{t+1})]\} \quad (4.134)$$

This applies for all discrete storage volumes S_t and for all within-year periods t and remaining periods n. The constraints on the decision-variables R_t are:

$$R_t \leq S_t + Q_t,$$
$$R_t \geq S_t + Q_t - K, \quad \text{and}$$
$$S_{t+1} = S_t + Q_t - R_t \tag{4.135}$$

These recursive Equations 4.134 together with constraint Equations 4.135 can be solved, beginning with $n = 1$ and then for successive values of seasons t and remaining periods n, until a steady-state solution is obtained, that is, until

$$F_t^n(S_t) - F_t^{n-T}(S_t) = 0 \quad \text{for all values of } S_t \text{ and periods } t. \tag{4.136}$$

The steady-state yields $F_t(S_t)$ will depend on the storage volumes S_t. High initial storage volumes will result in higher yields than will lower ones. The highest yield will be that associated with the highest storage volumes and it will equal the same value obtained from either of the two linear programming models.

5.2. A Water Quality Management Problem

Some linear programming modelling and solution techniques can be demonstrated using the simple water quality management example shown in Figure 4.20. In addition, this example can serve to illustrate how models can help identify just what data are needed and how accurate they must be for the decisions that are being considered.

The stream shown in Figure 4.20 receives wastewater effluent from two point sources located at Sites 1 and 2. Without some wastewater treatment at these sites, the concentration of some pollutant, P_j mg/l, at sites $j = 2$ and 3, will continue to exceed the maximum desired concentration P_j^{max}. The problem is to find the level of wastewater treatment (waste removed) at sites $i = 1$ and 2 that will achieve the desired concentrations at sites $j = 2$ and 3 at a minimum total cost.

This is the classic water quality management problem that is frequently found in the literature, although least-cost objectives have not been applied in practice. There are valid reasons for this that we will review later. Nevertheless, this particular problem can serve to illustrate the development of some linear models for determining data needs, estimating the values of model parameters, and for finding, in this case, cost-effective treatment efficiencies. This problem can also serve to illustrate graphically the general mathematical procedures used for solving linear programming problems.

The first step is to develop a model that predicts the pollutant concentrations in the stream as a function of the pollutants discharged into it. To do this we need some notation. Define P_j to be the pollutant concentration in the stream at site j. The total mass per unit time of the pollutant (M/T) in the stream at site j will be its concentration P_j (M/L^3) times the streamflow Q_j (L^3/T). For example if the concentration is in units of mg/l and

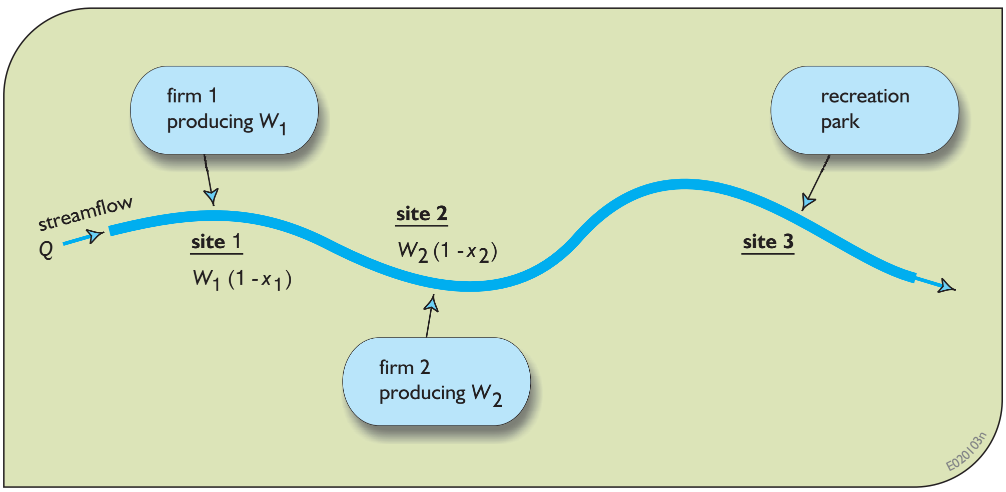


Figure 4.20. A stream pollution problem that requires finding the waste removal efficiencies (x_1, x_2) of wastewater treatment at Sites 1 and 2 that meet the stream quality standards at Sites 2 and 3 at minimum total cost. W_1 and W_2 are the amounts of pollutant prior to treatment at Sites 1 and 2.

the flow is in terms of m^3/s, and mass is to be expressed as kg/day:

$$\text{Mass at site } j \text{ (kg/day)} = P_j \text{ (mg/l) } Q_j \text{ (m}^3\text{/s) (}10^3 \text{ litres/m}^3\text{) (kg/}10^6 \text{ mg) (86400 sec./day)} = 86.4\, P_j Q_j \tag{4.137}$$

Each unit of a degradable pollutant mass in the stream at Site 1 in this example will decrease as it travels downstream to Site 2. Similarly each unit of the pollutant mass in the stream at Site 2 will decrease as it travels downstream to Site 3. The fraction a_{12} of the mass at Site 1 that reaches Site 2 is often assumed to be:

$$a_{12} = \exp(-\mathrm{k}t_{12}) \tag{4.138}$$

where k is a rate constant (1/time unit) that depends on the pollutant and the temperature, and t_{12} is the time (number of time units) it takes a particle of pollutant to flow from Site 1 to Site 2. A similar expression, a_{23}, applies for the fraction of pollutant mass at Site 2 that reaches Site 3. The actual concentration at the downstream end of a reach will depend on the streamflow at that site as well as on the initial pollutant mass, the time of travel and rate constant k.

In this example problem, the fraction of pollutant mass at Site 1 that reaches Site 3 is the product of the transfer coefficients a_{12} and a_{23}:

$$a_{13} = a_{12}a_{23} \tag{4.139}$$

In general, for any site k between sites i and j:

$$a_{ij} = a_{ik}a_{kj} \tag{4.140}$$

Knowing the a_{ij} values for any pollutant and the time of flow t_{ij} permits the determination of the rate constant k for that pollutant and reach, or contiguous series of reaches, from sites i to j, using Equation 4.138. If the value of k is 0, the pollutant is called a conservative pollutant; salt is an example of this. Only increased dilution by less polluted water will reduce its concentration.

For the purposes of determining wastewater treatment efficiencies or other capital investments in infrastructure designed to control the pollutant concentrations in the stream, some 'design' streamflow conditions have to be established. Usually the design streamflow condition is set at some very low flow value (e.g., the lowest monthly average flow expected once in twenty years, or the minimum seven-day average flow expected once in ten years). Low design flows are based on the assumption that pollutant concentrations will be higher in low-flow conditions than in higher-flow conditions because of less dilution. While low-flow conditions may not provide as much dilution, they do result in longer travel times, and hence greater reductions in pollutant masses between water quality monitoring sites. Hence the pollutant concentrations may well be greater at some downstream site when the flow conditions are higher than those of the low-flow design value.

In any event, given particular design streamflow and temperature conditions, our first job is to determine the values of these dimensionless transfer coefficients a_{ij}. They will be independent of the amount of waste discharged into the stream. To determine both a_{12} and a_{23} in this example problem (Figure 4.20) requires a number of pollutant concentration measurements at Sites 1, 2 and 3 during design streamflow conditions. These measurements of pollutant concentrations must be made just downstream of the wastewater effluent discharge at Site 1, just upstream and downstream of the wastewater effluent discharge at Site 2, and at Site 3.

Assuming no change in streamflow and no extra pollutant entering the reach that begins just downstream of Site 1 and ends just upstream of Site 2, the pollutant concentration P_2 just upstream of Site 2 will equal the concentration just downstream of Site 1, P_1, times the transfer coefficient a_{12}:

$$P_2 = P_1a_{12} \tag{4.141}$$

Under similar equal flow conditions in the following reach beginning just downstream from Site 2 and extending to Site 3, the pollutant concentration P_3 will equal the concentration just downstream of Site 2, P_2^+, times the transfer coefficient a_{23}.

If the streamflows Q_i and Q_j at sites i and j differ, then the downstream pollutant concentration P_j resulting from an upstream concentration of P_i will be:

$$P_j = P_ia_{ij}(Q_i/Q_j) \tag{4.142}$$

5.2.1. Model Calibration

Sample measurements are needed to estimate the values of each reach's pollutant transport coefficients a_{ij}. Assume five pairs of sample pollutant concentration measurements have been taken in the two stream reaches during design flow conditions. For this example, also assume that the design streamflow just downstream of Site 1 and just upstream of Site 2 are the same and equal to 12 m^3/s. The concentration samples taken just downstream from Site 1 and just upstream of Site 2 during this design flow condition can be used to solve Equation 4.142 after adding error terms. More than one sample is needed to allow for measurement errors and other random effects such as those from wind, incomplete mixing or varying wasteload discharges within a day.

Denote the concentrations of each pair of sample measurements s in the first reach (just downstream of Site 1 and just upstream of Site 2) as SP_{1s} and SP_{2s} and their combined error as E_s. Equation 4.142 becomes

$$SP_{2s} + E_s = SP_{1s}\, a_{12}\, (Q_1/Q_2) \tag{4.143}$$

The problem is to find the best estimates of the unknown a_{12}. One way to do this is to define 'best' as those values of a_{12} and all E_s that minimize the sum of the absolute values of all the error terms E_s. This objective could be written

$$\text{Minimize} \sum_s |E_s| \tag{4.144}$$

The set of Equations 4.143 and 4.144 is an optimization model. If the absolute value signs can be removed, these equations together with constraints that require all unknown variables to be non-negative would form a linear programming model.

The absolute value signs in Equation 4.144 can be removed by writing each error term as the difference between two non-negative variables, $PE_s - NE_s$. Thus for each sample pair s:

$$E_s = PE_s - NE_s \tag{4.145}$$

If any E_s is negative, PE_s will be 0 and $-NE_s$ will equal E_s. The actual value of NE_s is non-negative. If E_s is positive, it will equal PE_s, and NE_s will be 0. The objective function, Equation 4.154, that minimizes the sum of absolute value of error terms, can now be written as one that minimizes the sum of the positive and negative components of E_s:

$$\text{Minimize} \sum_s (PE_s + NE_s) \tag{4.146}$$

Equations 4.143 and 4.145, together with objective function 4.146 and a set of measurements, SP_{1s} and SP_{2s}, upstream and downstream of the reach between Sites 1 and 2 define a linear programming model that can be solved to find the transfer coefficient a_{12}. An example illustrating the solution of this model for the stream reach between Site 1 and just upstream of Site 2 is presented in Box 4.2. Again, the program LINGO (www.lindo.com) is used to solve the models.

Box 4.3 contains the model and solution for the reach beginning just downstream of Site 2 to Site 3. In this reach the design streamflow is 12.5 m^3/s due to the addition of wastewater flow at Site 2.

In this example, based on the solutions for a_{ij} and flows given in Boxes 4.2 and 4.3, $a_{12} = 0.25$, $a_{23} = 0.60$, and thus from Equation 4.140, $a_{12}\, a_{23} = a_{13} = 0.15$.

5.2.2. Management Model

Now that these parameter values a_{ij} are known, a water quality management model can be developed. The water quality management problem, illustrated in Figure 4.20, involves finding the fractions x_i of waste removal at sites $i = 1$ and 2 that meet the stream quality standards at the downstream Sites 2 and 3 at a minimum total cost.

The pollutant concentration, P_2, just upstream of Site 2 that results from the pollutant concentration at Site 1 equals the total mass of pollutant at Site 1 times the fraction of that mass remaining at Site 2, a_{12}, divided by the streamflow just upstream of Site 2, Q_2. The total mass of pollutant at Site 1 at the wastewater discharge point is the sum of the mass just upstream of the discharge site, P_1Q_1, plus the mass discharged into the stream, $W_1(1 - x_1)$, at Site 1. The parameter W_1 is the total amount of pollutant entering the treatment plant at Site 1. Similarly for Site 2. Hence the concentration of pollutant just upstream of Site 2 is:

$$P_2 = [P_1Q_1 + W_1(1-x_1)]a_{12}/Q_2 \tag{4.147}$$

The terms P_1 and Q_1 are the pollutant concentration (M/L^3) and streamflow (L^3/T) just upstream of the wastewater discharge outfall at Site 1. Their product is the mass of pollutant at that site per unit time period (M/T).

The pollutant concentration, P_3, at Site 3 that results from the pollutant concentration at Site 2 equals the total mass of pollutant at Site 2 times the fraction a_{23}. The total mass of pollutant at Site 2 at the wastewater discharge point is the sum of what is just upstream of the discharge site, P_2Q_2, plus what is discharged into the stream, $W_2(1-x_2)$. Hence the concentration of pollutant at Site 3 is:

$$P_3 = [P_2Q_2 + W_2(1-x_2)]a_{23}/Q_3 \tag{4.148}$$

Box 4.2. Calibration of water quality model transfer coefficient parameter a_{12}

E020903b

```
! Calibration of Water Quality Model parameter a12.

Define variables:
SP1(k) = sample pollutant concentration just downstream of site 1 (mg/l).
SP2(k) = sample pollutant concentration just upstream of site 2 (mg/l).
PE(k) = positive error in pollutant conc. sample just upstream of site 2 (mg.l).
NE(k) = negative error in pollutant conc. sample just upstream of site 2 (mg.l).
Qi = streamflow at site i (i=1, 2), (m3/s).
a12 = pollutant transfer coefficient for stream reach between sites 1 and 2. ;
Sets:
Sample / 1..5 / : PE, NE, SP1, SP2 ;
Endsets
!
Objective: Minimize total sum of positive and negative errors.
;
Min = @sum( Sample: PE + NE )
;
! Subject to constraint for each sample k:
;
@For (Sample: a12 * SP1 = ( SP2 + PE - NE )* (Q2/Q1));

Data:
SP1 = 232, 256, 220, 192, 204;
SP2 =  55,  67,  53,  50,  51;
Q1 = 12; !Flow downstream of site 1; Q2 = 12; !Flow upstream of site 2;
Enddata


Solution: a12 = 0.25; Total sum of absolute values of deviations = 10.0
```

Equations 4.147 and 4.148 will become the predictive portion of the water quality management model. The remaining parts of the model include the restrictions on the pollutant concentrations at Sites 2 and 3, and limits on the range of values that each waste removal efficiency x_i can assume.

$$P_j \leq P_j^{\max} \quad \text{for } j = 2 \text{ and } 3 \tag{4.149}$$

$$0 \leq x_i \leq 1.0 \quad \text{for } i = 1 \text{ and } 2. \tag{4.150}$$

Finally, the objective is to minimize the total cost of meeting the stream quality standards $P_2^{\max}$ and $P_3^{\max}$ specified in Equations 4.149. Letting $C_i(x_i)$ represent the wastewater treatment cost function for site i, the objective can be written:

$$\text{Minimize } C_1(x_1) + C_2(x_2) \tag{4.151}$$

The complete optimization model consists of Equations 4.147 through 4.151. There are four unknown decision variables, x_1, x_2, P_2, and P_3. All variables are assumed to be non-negative.

Some of the constraints of this optimization model can be combined to remove the two unknown concentration values, P_2 and P_3. Combining Equations 4.147 and 4.149, the concentration just upstream of Site 2 must be no greater than $P_2^{\max}$:

$$[P_1Q_1 + W_1(1-x_1)]a_{12}/Q_2 \leq P_2^{\max} \tag{4.152}$$

Combining Equations 4.148 and 4.149, and using the fraction a_{13} (see Equation 4.139) to predict the contribution of the pollutant concentration at Site 1 on the pollutant concentration at Site 3:

$$\{[P_1Q_1 + W_1(1-x_1)]a_{13} + [W_2(1-x_2)]a_{23}\}/Q_3 \leq P_3^{\max} \tag{4.153}$$

Box 4.3. Calibration of water quality model transfer coefficient parameter a_{23}

E020903c

```
! Calibration of Water Quality Model parameter a23.

Define variables:
SP2(k) = sample pollutant concentration just downstream of site 2 (mg/l).
SP3(k) = sample pollutant concentration at site 3. (mg/l).
PE(k) = positive error in pollutant conc. sample at site 3  (mg/l).
NE(k) = negative error in pollutant conc. sample at site 3   (mg/l).
Qi = streamflow at site i  (i=2, 3), (m3/s).
a23 = pollutant transfer coefficient for stream reach between sites 2 and 3.
;
Sets:
Sample / 1..5 / : PE, NE, SP2, SP3   ;
Endsets
!
Objective:   Minimize total sum of positive and negative errors.
;
Min = @sum( Sample:  PE + NE  )
;
!  Subject to constraint for each sample k:
;
@For (Sample:  a23 * SP2  = (SP3 + PE – NE )* (Q3/Q2) );

Data:
SP2 = 158,  180,  140,   150,   135;
SP3 =   96,  107,   82,    92,     81;
Q2 = 13;  !Flow just downstream of site 2; Q3 = 13; !Flow at site 3;
Enddata


Solution: a23 = 0.60; Total sum of absolute values of deviations = 6.2
```

Equation 4.153 assumes that each pollutant discharged into the stream can be tracked downstream, independent of the other pollutants in the stream. Alternatively, Equation 4.148 computes the sum of all the pollutants found at Site 2 and then uses that total mass to compute the concentration at Site 3. Both modelling approaches give the same results if the parameter values and cost functions are the same.

To illustrate the solution of either of these models, assume the values of the parameters are as listed in Table 4.22. Rewriting the water quality management model defined by Equations 4.150 to 4.153 and substituting the parameter values in place of the parameters, and recalling that kg/day $= 86.4$ (mg/l)(m^3/s):

$$\text{Minimize } C_1(x_1) + C_2(x_2) \tag{4.154}$$

Subject to:

Water quality constraint at Site 2:

$$[P_1Q_1 + W_1(1-x_1)]a_{12}/Q_2 \leq P_2^{\max} \tag{4.155}$$

$$[(32)(10) + 250000(1-x_1)/86.4]0.25/12 \leq 20$$

that when simplified is: $x_1 \geq 0.78$

Water quality constraint at Site 3:

$$[P_1Q_1 + W_1(1-x_1)]a_{13} + [W_2(1-x_2)]a_{23}\}/Q_3 \leq P_3^{\max} \tag{4.156}$$

$$\{[(32)(10) + 250000(1-x_1)/86.4]0.15 + [80000(1-x_2)/86.4]0.60\}/13 \leq 20$$

that when simplified is: $x_1 + 1.28x_2 \geq 1.79$

Restrictions on fractions of waste removal:

$$0 \leq x_i \leq 1.0 \quad \text{for sites } i = 1 \text{ and } 2 \tag{4.157}$$

Table 4.22. Parameter values selected for the water quality management problem illustrated in Figure 4.20.

	parameter	unit	value	remark
flow	Q_1	m^3/s	10	flow just upstream of site 1
	Q_2	m^3/s	12	flow just upstream of site 2
	Q_3	m^3/s	13	flow at park
waste	W_1	kg/day	250,000	pollutant mass produced at site 1
	W_2	kg/day	80,000	pollutant mass produced at site 2
pollutant conc.	P_1	mg/l	32	concentration just upstream of site 1
	P_2	mg/l	20	maximum allowable concentration upstream of 2
	P_3	mg/l	20	maximum allowable concentration at site 3
decay fraction	a_{12}	--	0.25	fraction of site 1 pollutant mass at site 2
	a_{13}	--	0.15	fraction of site 1 pollutant mass at site 3
	a_{23}	--	0.60	fraction of site 2 pollutant mass at site 2

E020827r

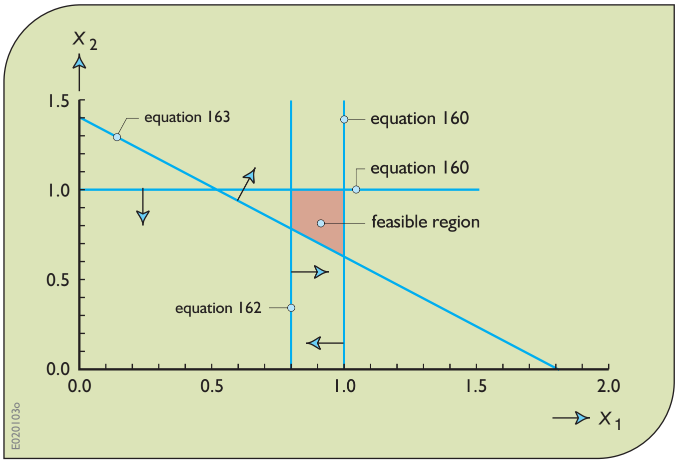


Figure 4.21. Plot of the constraints of water quality management model identifying those values of the unknown (decision) variables x_1 and x_2 that satisfy all the constraints.

The feasible combinations of x_1 and x_2 can be shown on a graph, as in Figure 4.21. This graph is a plot of each constraint, showing the boundaries of the region of combinations of x_1 and x_2 that satisfy all the constraints. This red shaded region is called the feasible region.

To find the least-cost solution we need the cost functions $C_1(x_1)$ and $C_2(x_2)$ in Equations 4.151 or 4.154. Suppose these functions are not known. Can we determine the least-cost solution without knowing these costs? Models like the one just developed can be used to determine just how accurate these cost functions (or any model parameters) need to be for the decisions being considered.

While the actual cost functions are not known, their general form can be assumed, as shown in Figure 4.22. Since the wasteloads produced at Site 1 are substantially greater than those produced at Site 2, and given similar site, transport, labour, and material cost conditions, it seems reasonable to assume that the cost of providing a specified level of treatment at Site 1 would exceed (or certainly be no less than) the cost of providing the same specified level of treatment at Site 2. It would also seem the marginal cost at Site 1 would be greater than, or at least no less than, the marginal cost at Site 2 for the same amount of treatment. The relative positions of the cost functions shown in Figure 4.22 are based on these assumptions.

Rewriting the cost function, Equation 4.154, as a linear function converts the model defined by Equations 4.150 through 4.151 into a linear programming model. For this example problem, the linear programming model can be written as:

$$\text{Minimize } c_1x_1 + c_2x_2 \tag{4.158}$$

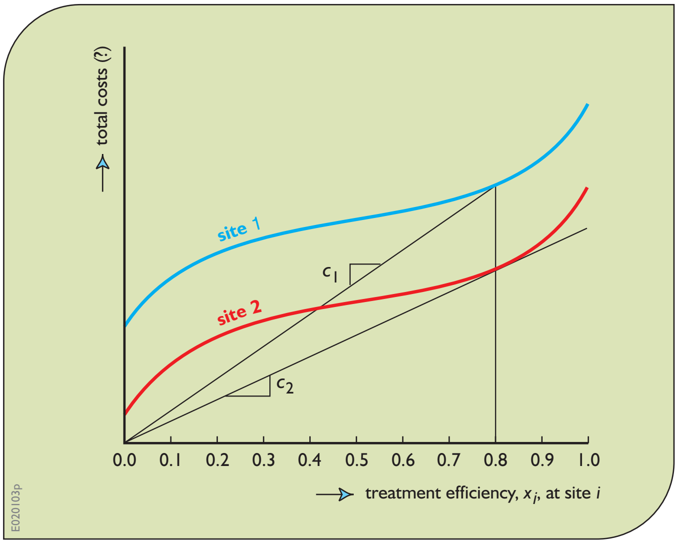


Figure 4.22. General form of total cost functions for wastewater treatment efficiencies at Sites 1 and 2 in Figure 4.20. The dashed straight-line slopes c_1 and c_2 are the average cost per unit (%) removal for 80% treatment. The actual average costs clearly depend on the values of the waste removal efficiencies x_1 and x_2 respectively.

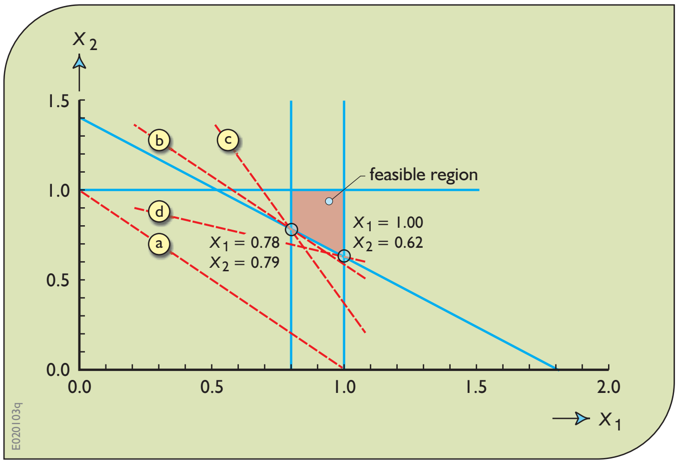


Figure 4.23. Plots of various objective functions (dashed lines) together with the constraints of the water quality management model.

Subject to:

$$x_1 \geq 0.78 \tag{4.159}$$

$$x_1 + 1.28\, x_2 \geq 1.79 \tag{4.160}$$

$$0 \leq x_i \leq 1.0 \quad \text{for } i = 1 \text{ and } 2 \tag{4.161}$$

where the values of c_1 and c_2 depend on the values of x_1 and x_2 and both pairs are unknown. Even if we knew the values of x_1 and x_2 before solving the problem, in this example the cost functions themselves (Figure 4.22) are unknown. Hence, we cannot determine the values of the marginal costs c_1 and c_2. However, we might be able to judge which will likely be greater than the other for any particular values of the decision-variables x_1 and x_2.

First, assume c_1 equals c_2. Let $c_1x_1 + c_2x_2$ equal c and assume $c/c_1 = 1$. Thus $x_1 + x_2 = 1.0$. This line can be plotted onto the graph in Figure 4.23, as shown by line 'a' in Figure 4.23.

Line 'a' in Figure 4.23 represents equal values for c_1 and c_2, and the total cost, $c_1x_1 + c_2x_2$, equal to 1. Keeping the slope of this line constant and moving it upward, representing increasing total costs, to line 'b', where it covers the nearest point in the feasible region, will identify the least-cost combination of x_1 and x_2, again assuming the marginal costs are equal. In this case the solution is approximately 80% treatment at both sites.

Note this particular least-cost solution applies for any value of c_1 greater than c_2 (for example line 'c' in Figure 4.23). If the marginal cost of 80% treatment at Site 1 is no less than the marginal cost of 80% treatment at Site 2, then $c_1 \geq c_2$ and indeed the 80% treatment efficiencies will meet the stream standards for the design streamflow and wasteload conditions at a total minimum cost. In fact, from Figure 4.23 and Equation 4.160, it is clear that c_2 has to exceed c_1 by a multiple of 1.28 before the least-cost solution changes to another solution. For any other assumption regarding c_1 and c_2, 80% treatment at both sites will result in a least-cost solution to meeting the water quality standards for those design wasteload and streamflow conditions.

If c_2 exceeds $1.28c_1$, as illustrated by line 'd', then the least-cost solution would be $x_1 = 100\%$ and $x_2 = 62\%$. Clearly, in this example the marginal cost, c_1, of providing 100% wasteload removal at Site 1 will exceed the marginal cost, c_2, of 60% removal at Site 2, and hence, that combination of efficiencies would not be a least-cost one. Thus we can be confident that the least-cost solution is to remove 80% of the waste produced at both waste-generating sites.

Note the least-cost wasteload removal efficiencies have been determined without knowing the cost functions. Why spend money defining these functions more precisely? The answer: costs need to be known for financial planning, if not for economic analyses. No doubt the actual costs of installing the least-cost treatment efficiencies of 80% will have to be determined for issuing bonds or making other arrangements for paying the costs. However, knowing the least-cost removal efficiencies means we do not have to spend money defining the entire cost functions $C_i(x_i)$. Estimating the construction and operating costs of achieving just one wastewater removal efficiency at each site, namely 80%, should be less expensive than defining the total costs for a range of practical treatment plant efficiencies that would be required to define the total cost functions, such as shown in Figure 4.22.

Admittedly this example is relatively simple. It will not always be possible to determine the 'optimal' solutions to linear programming problems, or other optimization problems, without knowing more about the objective function than was assumed for this example. However, this exercise illustrates the use of modelling for purposes other than finding good or 'optimal' solutions. Models can help define the necessary precision of the data needed to find those good solutions.

Modelling and data collection and analysis should take place simultaneously. All too often planning exercises are divided into two stages: data collection and then analysis. Until one knows what data one will need, and how accurate those data must be, one should not spend money and time collecting them. Conversely, model development in the absence of any knowledge of the availability and cost of obtaining data can lead to data requirements that are costly, or even impossible, to obtain, at least in the time available for decision-making. Data collection and model development are activities that should be performed simultaneously.

Because software is widely available to solve linear programming programs, because these software programs can solve very large problems containing thousands of variables and constraints, and finally because there is less chance of obtaining a local 'non-optimal' solution when the problem is linear (at least in theory), there is an incentive to use linear programming to solve large optimization problems. Especially for large optimization problems, linear programming is often the only practical alternative. Yet models representing particular water resources systems may not be linear. This motivates the use of methods that can approximate non-linear functions with linear ones.

The following simple groundwater supply problem illustrates the application of some linearization methods commonly applied to non-linear separable functions – functions of only one unknown variable.

These methods typically increase the number of variables and constraints in a model. Some of these methods require integer variables, or variables that can have values of only 0 or 1. There is a practical limit on the number of integer variables any linear programming software program can handle. Hence, for large models there may be a need to perform some preliminary screening designed to reduce the number of alternatives that should be considered in more detail. This example can be used to illustrate an approach to preliminary screening.

5.3. A Groundwater Supply Example

Consider a water-using industry that plans to obtain water from a groundwater aquifer. Two wellfield sites have been identified. The first question is how much the water will cost, and the second, given any specified amount of water delivered to the user, is how much should come from each wellfield. This situation is illustrated in Figure 4.24.

Wells and pumps must be installed and operated to obtain water from these two wellfields. The annual cost of wellfield development will depend on the pumping capacity of the wellfield. Assume that the annual costs associated with various capacities Q_A and Q_B for Wellfields

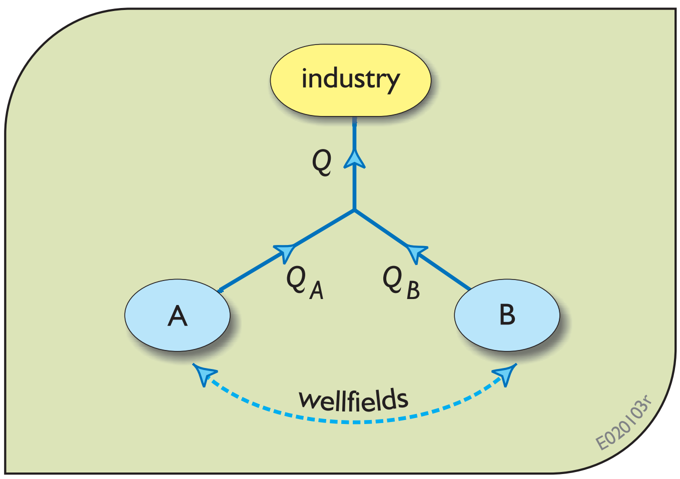


Figure 4.24. Schematic of a potential groundwater supply system that can serve a water using industry. The unknown variables are the flows, Q_A and Q_B, from each wellfield.

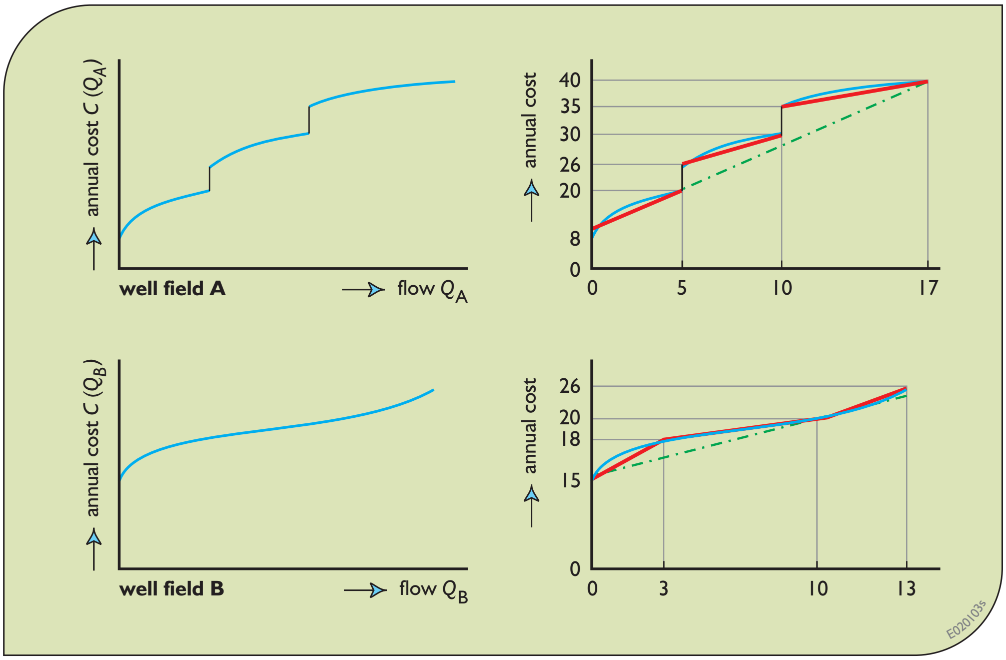


Figure 4.25. Annual cost functions associated with the Wellfields A and B as shown in Figure 4.24. The actual functions are shown on the left, and two sets of piecewise linear approximations are shown on the right.

A and B respectively are as shown in Figure 4.25. These are non-linear functions that contain both fixed and variable costs and hence are discontinuous. The fixed costs result from the fact that some of the components required for wellfield development come in discrete sizes. As indicated in the figure, the maximum flow capacity of Wellfields A and B are 17 and 13, respectively.

In Figure 4.25, the non-linear functions on the left have been approximated by piecewise linear functions on the right. This is a first step in linearizing non-linear separable functions. Increasing the number of linear segments can reduce the difference between the piecewise linear approximation of the actual non-linear function and the function itself. At the same time it will increase the number of variables and possibly constraints.

When approximating a non-linear function by a series of straight lines, model developers should consider two factors. The first is just how accurate need be the approximation of the actual function for the decisions that will be made, and second is just how accurate is the actual (in this case non-linear) function in the first place. There is little value in trying to eliminate relatively small errors caused by the linearization of a function when the function itself is highly uncertain. Most cost and benefit functions, especially those associated with future activities, are indeed uncertain.

5.3.1. A Simplified Model

Two sets of approximations are shown in Figure 4.25. Consider first the approximations represented by the light blue dot–dash lines. These are very crude approximations – a single straight line for each function. In this example these straight-line cost functions are lower bounds of the actual non-linear costs. Hence, the actual costs may be somewhat higher than those identified in the solution of a model.

Using the blue dot–dash linear approximations in Figure 4.25, the linear programming model can be written as follows:

$$\text{Minimize CostA} + \text{CostB} \tag{4.162}$$

Subject to:

$$\text{CostA} = 8I_A + [(40-8)/17]Q_A \quad \{\text{linear approximation of } C(Q_A)\} \tag{4.163}$$

$$\text{CostB} = 15I_B + [(26-15)/13]Q_B \quad \{\text{linear approximation of } C(Q_B)\} \tag{4.164}$$

$$I_A,\ I_B \text{ are 0, 1 integer variables} \tag{4.165}$$

$$Q_A \le 17I_A \ \{\text{limits } Q_A \text{ to 17 and forces } I_A = 1 \text{ if } Q_A > 0\} \tag{4.166}$$

$$Q_B \le 13I_B \ \{\text{limits } Q_B \text{ to 13 and forces } I_B = 1 \text{ if } Q_B > 0\} \tag{4.167}$$

$Q = Q_A + Q_B$ {mass balance} (4.168)

$Q, Q_A, Q_B \geq 0$ {non-negativity of all decision variables} (4.169)

Q = some specified amount from 0 to 30. (4.170)

The expressions within the square brackets, [ ], above represent the slopes of the dot–dash linear approximations of the cost functions. The integer 0, 1 variables are required to include the fixed costs in the model.

Solving this model for various values of the water demand Q provides some interesting results. Again, they are based on the dot–dash linear cost functions in Figure 4.25. As Q increases from 0 to just under 6.8, all the water will come from the less expensive Wellfield A. For any Q from 6.8 to 13, Wellfield B becomes less expensive and all the water will come from it. For any Q greater than the capacity of Wellfield B of 13 but no greater than the capacity of Wellfield A, 17, all of it will come from Wellfield A. Because of the fixed costs, it is cheaper to use one rather than both wellfields. Beyond $Q = 17$, the maximum capacity of A, water needs to come from both wellfields. Wellfield B will pump at its capacity, 13, and the additional water will come from Wellfield A.

Figure 4.26 illustrates these solutions. One can understand why in situations of increasing demands for Q over time, capacity-expansion modelling might be useful. One would not close down a wellfield once developed, just to achieve what would have been a least-cost solution if the existing wellfield had not been developed.

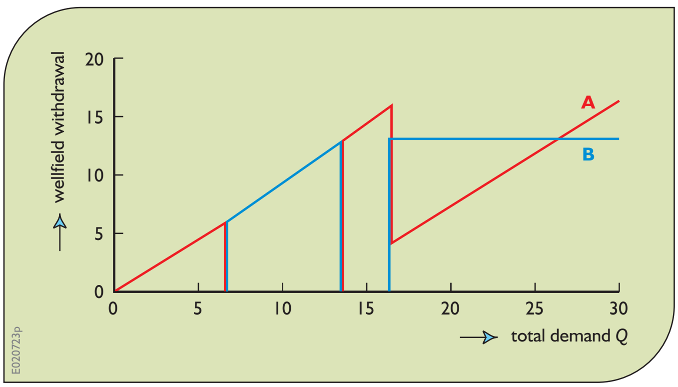


Figure 4.26. Least-cost wellfield use given total demand *Q* based on model defined by Equations 4.162 to 4.170.

5.3.2. A More Detailed Model

A more accurate representation of these cost functions may change these solutions for various values of Q, although not significantly. However consider the more accurate cost minimization model that includes the red solid-line piecewise linearizations shown in Figure 4.25.

Minimize CostA + CostB (4.171)

Subject to:

$$\text{CostA} = \{8I_{A1} + [(20-8)/5]Q_{A1}\} + \{26I_{A2} + [(30-26)/(10-5)]Q_{A2}\} + \{35I_{A3} + [(40-35)/(17-10)]Q_{A3}\} \text{ \{linear approximation of } C(Q_A)\text{\}} \quad (4.172)$$

$$\text{CostB} = \{15I_{B1} + [(18-15)/3]Q_{B1}\} + \{18I_{B2} + [(20-18)/(10-3)]Q_{B2} + [(26-20)/(13-10)]Q_{B3}\} \text{ \{linear approximation of } C(Q_B)\text{\}} \quad (4.173)$$

$$Q_A = Q_{A1} + (5I_{A2} + Q_{A2}) + (10I_{A3} + Q_{A3}) \text{ \{}Q_A \text{ defined\}} \quad (4.174)$$

$$Q_B = Q_{B1} + (3I_{B2} + Q_{B2} + Q_{B3}) \text{ \{}Q_B \text{ defined\}} \quad (4.175)$$

I_{Ai}, I_{Bi} are 0, 1 integer variables for all segments i (4.176)

$Q_{A1} \leq 5I_{A1}$, $Q_{A2} \leq (10 - 5)I_{A2}$, $Q_{A3} \leq (17 - 10)I_{A3}$
{limits Q_{Ai} to width of segment i and forces $I_{Ai} = 1$ if $Q_{Ai} > 0$} (4.177)

$I_{A1} + I_{A2} + I_{A3} \leq 1$ (4.178)
{limits solution to at most only one cost function segment i}

$Q_{B1} \leq 3I_{B1}$, $Q_{B2} \leq (10-3)I_{B2}$, $Q_{B3} \leq (13-10)I_{B2}$ (4.179)
{limits Q_{Bi} to width of segment i and forces $I_{Bi} = 1$ if $Q_{Bi} > 0$}

$I_{B1} + I_{B2} \leq 1$ (4.180)
{limits solution to at most only the first segment or to the second and third segments of the cost function. Note that a 0, 1 integer variable for the fixed cost of the third segment of this function is not needed since its slope exceeds that of the second segment. However the flow, Q_{B3}, in that segment must be bounded using the integer 0, 1 variable, I_{B2}, associated with the second segment, as shown in the third of Equations 4.179}

$$Q = Q_A + Q_B \text{ \{mass balance\}} \tag{4.181}$$

$$Q, Q_A, Q_B \geq 0 \text{ \{non-negativity of all decision variables\}} \tag{4.182}$$

$$Q = \text{some specified amount from 0 to 30.} \tag{4.183}$$

The solution to this model, shown in Figure 4.27, differs from the solution of the simpler model, but only in the details. Wellfield A supplies all the water for $Q \leq 4.3$. For values of Q between 4.4 and 13 all the water comes from Wellfield B. For values of Q in excess of 13 to 14.8, the capacity of Wellfield B remains at 13 and Wellfield A provides the additional amount of needed capacity over 13. For $Q = 14.9$ to 17, the capacity of Wellfield B drops to 0 and the capacity of Wellfield A increases from 14.9 to 17. For values of Q between 17 and 18 Wellfield B provides 13, its capacity, and the capacity of A increases from 4 to 5. For values of Q from 18.1 to 27, Wellfield B provides 10, and Wellfield A increases from 8.1 to 17.

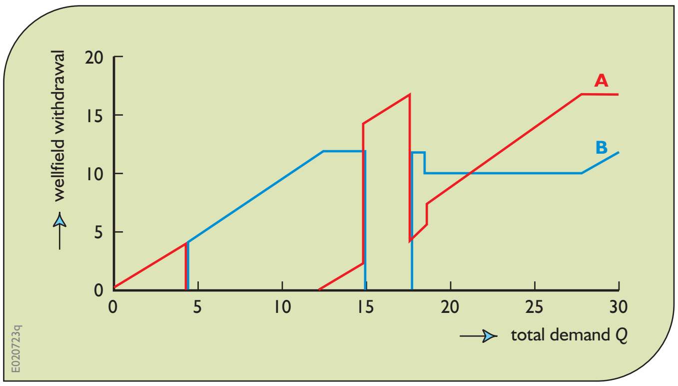


Figure 4.27. Least-cost wellfield use given total demand *Q* based on Equations 4.171 to 4.183.

For values of Q in excess of 27, Wellfield A remains at 17, its capacity, and Wellfield B increases from 10 to 13.

As in the previous example, this shows the effect on the least-cost solution when one cost function has relatively lower fixed and higher variable costs compared with another cost function having relatively higher fixed and lower variable costs.

5.3.3. An Extended Model

In this example, the simpler model (Equations 4.162 to 4.170) and the more accurate model (Equations 4.171 to 4.183) provided essentially the same allocations of wellfield capacities associated with a specified total capacity Q. If the problem contained a larger number of wellfields, the simpler (and smaller) model might have been able to eliminate some of these wellfields from further consideration. This would reduce the size of any new model that approximates the cost functions of the remaining wellfields more accurately.

The model just described, like the capacity-expansion model and water quality management model, is another example of a cost-effective model. The objective was to find the least-cost way of providing a specified amount of water to a water user. Next, consider a cost–benefit analysis in which the question is just how much water the user should use. To address this question we assume the user has identified the annual benefits associated with various amounts of water. The annual benefit function, $B(Q)$, and its piecewise linear approximations, are shown in Figure 4.28.

The straight, blue, dot–dash–dash linear approximation of the benefit function shown in Figure 4.28 is an

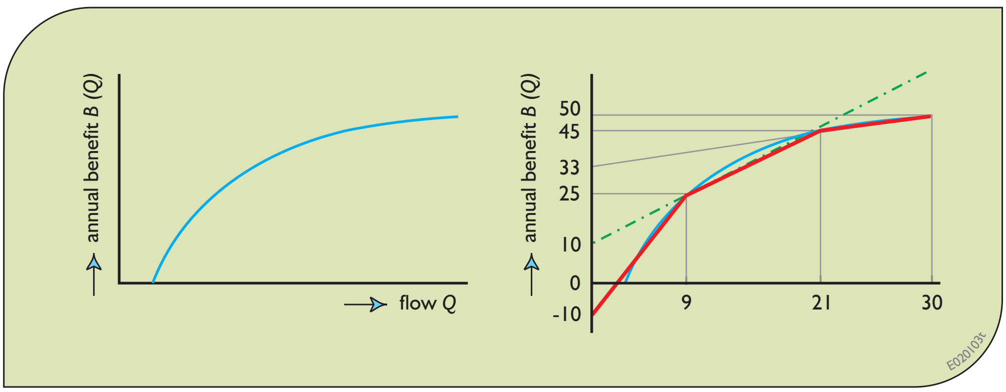


Figure 4.28. Benefit function of the amount of water provided to the water user. Piecewise linear approximations of that function of flow are shown on the right.

upper bound of the benefits. Incorporating it into a model that uses the dot–dash linear lower bound approximations of each cost function, as shown in Figure 4.25, will produce an optimistic solution. It is unlikely that the value of Q that is based on more accurate and thus less optimistic benefit and cost functions will be any greater than the one identified by this simple optimistic model. Furthermore, if any wellfield is not in the solution of this optimistic model, with some care we might be able to eliminate that wellfield from further consideration when developing a more accurate model.

Any component of a water resources system that does not appear in the solution of a model that includes optimistic approximations of performance measures that are to be maximized, such as benefits, or that are to be minimized, such as costs, are candidates for omission in any more detailed model. This is an example of the process of preliminary screening.

The model defined by Equations 4.162 through 4.170 can now be modified. Equation 4.170 is eliminated and the cost-minimization objective Equation 4.162 is replaced with:

$$\text{Maximize } Benefits - (\text{CostA} + \text{CostB}) \tag{4.184}$$

where

$$Benefits = 10 + [(45 - 25)/(21 - 9)]Q \quad \{\text{linear approximation of } B(Q)\} \tag{4.185}$$

The solution of this model, Equations 4.163 to 4.169, 4.184, and 4.185 (plus the condition that the fixed benefit of 10 only applies if $Q > 0$, added because it is clear the benefits would be 0 with a Q of 0) indicates that only Wellfield B needs to be developed, and at a capacity of 10. This would suggest that Wellfield A can be omitted in any more detailed modelling exercise. To see if this assumption, in this example, is valid, consider the more detailed model that incorporates the red, solid-line linear approximations of the cost and benefit functions shown in Figures 4.25 and 4.28.

Note that the approximation of the generally concave benefit function in Figure 4.28 will result in negative values of the benefits for small values of Q. For example, when the flow Q, is 0 the approximated benefits are –10. Yet the actual benefits are 0 as shown in the left part of Figure 4.28. Modelling these initial fixed benefits the same way as the fixed costs have been modelled, using another 0, 1 integer variable, would allow a more accurate representation of the actual benefits for small values of Q.

Alternatively, to save having to add another integer variable and constraint to the model, one can allow the benefits to be negative. If the model solution shows negative benefits for some small value of Q, then obviously the more preferred value of Q, and benefits, would be 0. This more approximate trial-and-error approach is often preferred in practice, especially when a model contains a large number of variables and constraints. This is the approach taken here.

5.3.4. Piecewise Linearization Methods

There are a number of ways of modelling the piecewise linear concave benefit function shown on the right side of Figure 4.28. Several are defined in the next several sets of equations. Each method will result in the same model solution.

One approach to modelling the concave benefit function is to define a new unrestricted (possibly negative-valued) variable. Let this variable be *Benefits*. When being maximized this variable cannot exceed any of the linear functions that bound the concave benefit function:

$$Benefits \leq -10 + [(25 - (-10))/9]Q \tag{4.186}$$

$$Benefits \leq 10 + [(45 - 25)/(21-9)]Q \tag{4.187}$$

$$Benefits \leq 33 + [(50 - 45)/(30-21)]Q \tag{4.188}$$

Since most linear programming algorithms assume all unknown variables are non-negative (unless otherwise specified), unrestricted variables, such as *Benefits,* can be replaced by the difference between two non-negative variables, such as *Pben* – *Nben*. *Pben* will equal *Benefits* if its value is greater than 0. Otherwise –*Nben* will equal *Benefits*. Thus in place of *Benefits* in the Equations 4.186 to 4.188, and those below, one can substitute *Pben* – *Nben*.

Another modelling approach is to divide the variable Q into parts, q_i, one for each segment i of the function. These parts sum to Q. Each q_i, ranges from 0 to the width of the user-defined segment i. Thus for the piecewise linear benefit function shown on the right of Figure 4.28:

$$q_1 \leq 9 \tag{4.189}$$

$$q_2 \leq 21 - 9 \tag{4.190}$$

$$q_3 \leq 30 - 21 \tag{4.191}$$

and

$$Q = q_1 + q_2 + q_3 \tag{4.192}$$

The linearized benefit function can now be written as the sum over all three segments of each segment slope times the variable q_i:

$$\text{Benefits} = -10 + [(25 + 10)/9]q_1 + [(45 - 25)/(21 - 9)]q_2 + [(50 - 45)/(30 - 21)]q_3 \tag{4.193}$$

Since the function being maximized is concave (decreasing slopes as Q increases), we are assured that each q_{i+1} will be greater than 0 only if q_i is at its upper limit, as defined by constraint Equations 4.189 to 4.191.

A third method is to define unknown weights w_i associated with the breakpoints of the linearized function. The value of Q can be expressed as the sum of a weighted combination of segment endpoint values. Similarly, the benefits associated with Q can be expressed as a weighted combination of the benefits evaluated at the segment endpoint values. The weights must also sum to 1.

Hence, for this example:

$$\text{Benefits} = (-10)w_1 + 25w_2 + 45w_3 + 50w_4 \tag{4.194}$$

$$Q = 0w_1 + 9w_2 + 21w_3 + 30w_4 \tag{4.195}$$

$$1 = w_1 + w_2 + w_3 + w_4 \tag{4.196}$$

For this method to provide the closest approximation of the original non-linear function, the solution must include no more than two non-zero weights and those non-zero weights must be adjacent to each other. Since a concave function is to be maximized, this condition will be met, since any other situation would yield less benefits.

The solution to the more detailed model defined by Equations 4.184, 4.172 to 4.182, and either 4.186 to 4.188, 4.189 to 4.193, or 4.194 to 4.196, indicates a value of 10 for Q will result in the maximum net benefits. This flow is to come from Wellfield B. This more precise solution is identical to the solution of the simpler model. Clearly the simpler model could have successfully served to eliminate Wellfield A from further consideration.

5.4. A Review of Linearization Methods

This section presents a review of the piecewise linearization methods just described and some other approaches for incorporating special conditions into linear programming models.

If–then–else conditions

There exist a number of ways 'if–then–else' and 'and' and 'or' conditions (that is, decision trees) can be included in linear programming models. To illustrate some of them, assume X is an unknown decision-variable in a model whose value may depend on the value of another unknown decision-variable Y. Assume the maximum value of Y would not exceed Ymax and the maximum value of X would not exceed Xmax. These upper bounds and all the linear constraints representing 'if–then–else' conditions must not restrict the values of the original decision-variable Y. Four 'if–then–else (with 'and/or') conditions are presented below using additional integer 0.1 variables, denoted by Z. All the X, Y and Z variables in the constraints below are assumed to be unknown. These constraints would be included in the any linear programming model where the particular 'if–then–else' conditions apply.

These illustrations are not unique. At the boundaries of the 'if' constraints in the examples below, either of the 'then' or 'else' conditions can apply.

a) If $Y \le 50$ then $X \le 10$, else $X \ge 15$.
   Define constraints:
   $Y \le 50Z + Y\text{max}(1 - Z)$ where Z is a 0, 1 integer variable.
   $Y \ge 50(1 - Z)$
   $X \le 10Z + X\text{max}(1 - Z)$
   $X \ge 15(1 - Z)$
b) If $Y \le 50$ then $X \le Y$, else $X \ge Y$.
   Define constraints:
   $Y \ge 50Z$ where Z is a 0, 1 integer variable.
   $Y \le 50(1 - Z) + Y\text{max}Z$
   $X \le Y + X\text{max}Z$
   $X \ge Y - Y\text{max}(1 - Z)$
c) If $Y \le 20$ or $Y \ge 80$ then $X = 5$, else $X \ge 10$.
   Define constraints:
   $Y \le 20Z_1 + 80Z_2 + Y\text{max}(1 - Z_1 - Z_2)$
   $Y \ge 20Z_2 + 80(1 - Z_1 - Z_2)$
   $Z_1 + Z_2 \le 1$ where each Z is a 0, 1 integer variable.
   $X \le 5(Z_1 + (1 - Z_1 - Z_2)) + X\text{max}Z_2$
   $X \ge 5(Z_1 + (1 - Z_1 - Z_2))$
   $X \ge 10Z_2$

d) If $20 \leq Y \leq 50$ or $60 \leq Y \leq 80$, then $X \leq 5$, else $X \geq 10$.

Define constraints:

$Y \leq 20Z_1 + 50Z_2 + 60Z_3 + 80Z_4 + \text{Ymax}(1 - Z_1 - Z_2 - Z_3 - Z_4)$

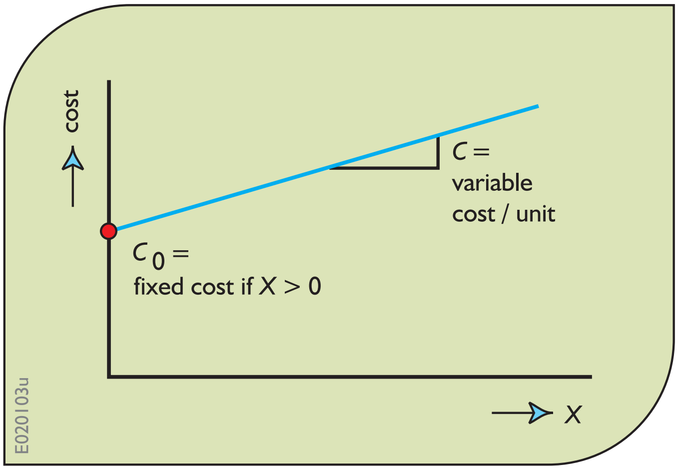
$Y \geq 20Z_2 + 50Z_3 + 60Z_4 + 80(1 - Z_1 - Z_2 - Z_3 - Z_4)$

$Z_1 + Z_2 + Z_3 + Z_4 \leq 1$ where each Z is a 0, 1 integer variable.

$X \leq 5\ (Z_2 + Z_4) + X\text{max}^*(1 - Z_2 - Z_4)$

$X \geq 10^*\ ((Z_1 + Z_3) + (1 - Z_1 - Z_2 - Z_3 - Z_4))$

Fixed costs in cost functions:



Cost $= C_0 + CX$ if $X > 0$,

$= 0$ otherwise.

To include these fixed costs in a LP model, define Cost $= C_0I + CX$ and constrain $X \leq MI$ where M is the maximum value of X, and I is an unknown 0, 1 variable.

Minimizing the maximum or maximizing the minimum

Let the set of variables be $\{X_1, X_2, X_3, \ldots, X_n\}$

Minimize maximum $\{X_1, X_2, X_3, \ldots, X_n\}$ is equivalent to:

Minimize U subject to $U \geq X_j$, $j = 1, 2, 3, \ldots, n$.

Maximize minimum $\{X_1, X_2, X_3, \ldots, X_n\}$ is equivalent to:

Maximize L subject to $L \leq X_j$, $j = 1, 2, 3, \ldots, n$.

Minimizing the absolute value of the difference between two unknown non-negative variables:

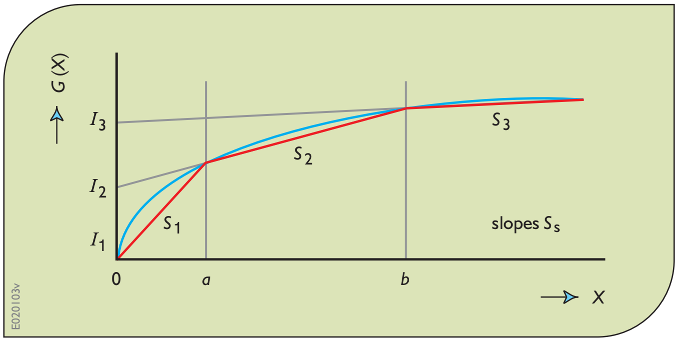
Minimize $|X - Y|$ is equivalent to

Minimize D subject to $X - Y \leq D$; $Y - X \leq D$; $X, Y, D \geq 0$.

or Minimize $(PD + ND)$

subject to: $X - Y = PD - ND$; $PD, ND, X, Y \geq 0$.

Minimizing convex functions or maximizing concave functions

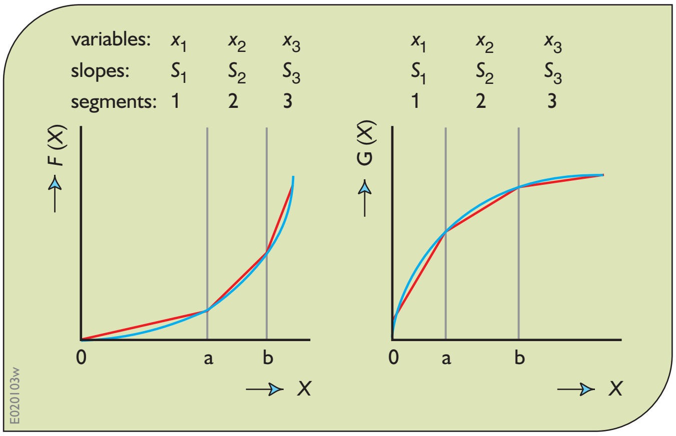


Maximize $G(X) \cong$ Maximize B

Subject to: $I_1 + S_1X \geq B$

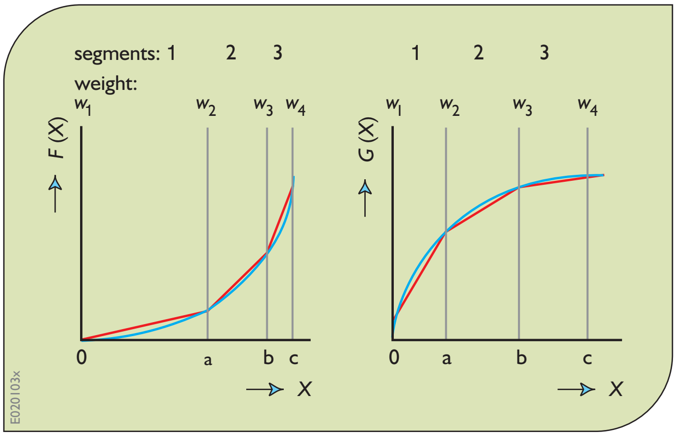
$I_2 + S_2X \geq B$

$I_3 + S_3X \geq B$



$F(X) \cong S_1x_1 + S_2x_2 + S_3x_3$;
$G(X) \cong S_1x_1 + S_2x_2 + S_3x_3$

$X = x_1 + x_2 + x_3$; $x_1 \leq a$; $x_2 \leq b - a$



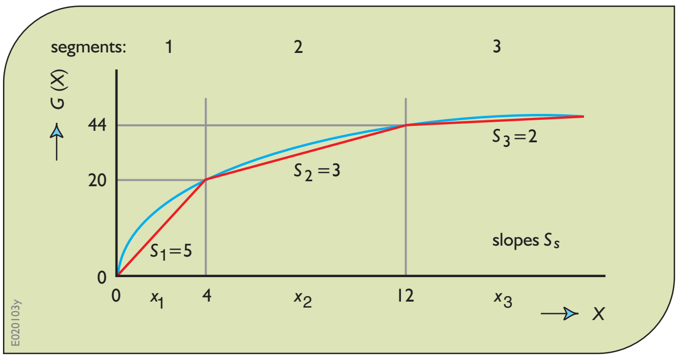
Minimize $F(X) \cong F(0)w_1 + F(\mathrm{a})w_2 + F(\mathrm{b})w_3 + F(\mathrm{c})w_4$

Maximize $G(X) \cong G(0)w_1 + G(\mathrm{a})w_2 + G(\mathrm{b})w_3 + G(\mathrm{c})w_4$

Subject to:

$X = 0w_1 + \mathrm{a}w_2 + \mathrm{b}w_3 + \mathrm{c}w_4$;
$w_1 + w_2 + w_3 + w_4 = 1$

Minimizing concave functions or maximizing convex functions



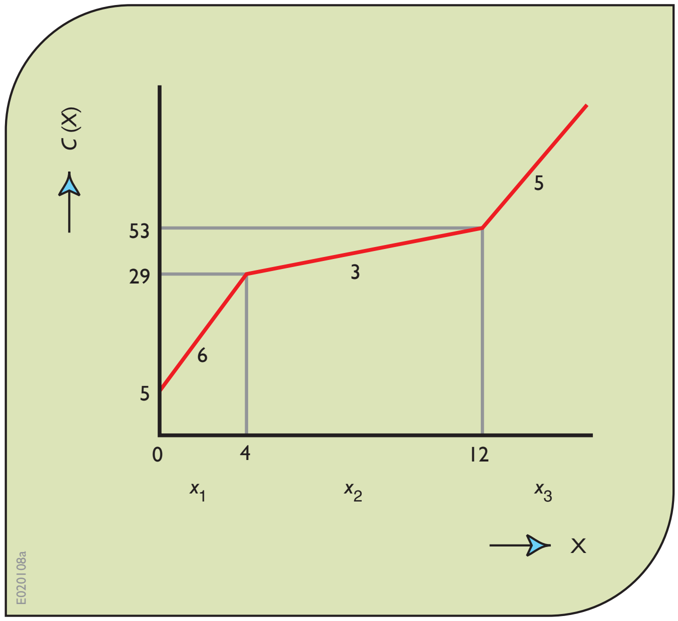
Minimize $G(X) \cong 5x_1 + (20z_2 + 3x_2) + (44z_3 + 2x_3)$

Subject to:

$x_1 + (4z_2 + x_2) + (12z_3 + x_3) = X$;
$z_s = 0$ or $1\ \forall s$;

$x_1 \le 4z_1$; $x_2 \le 8z_2$; $x_3 \le 99\ z_3$; $z_1 + z_2 + z_3 = 1$.

Minimizing or maximizing combined concave–convex functions



Maximize $C(X) \cong (5z_1 + 6x_1 + 3x_2) + (53z_3 + 5x_3)$

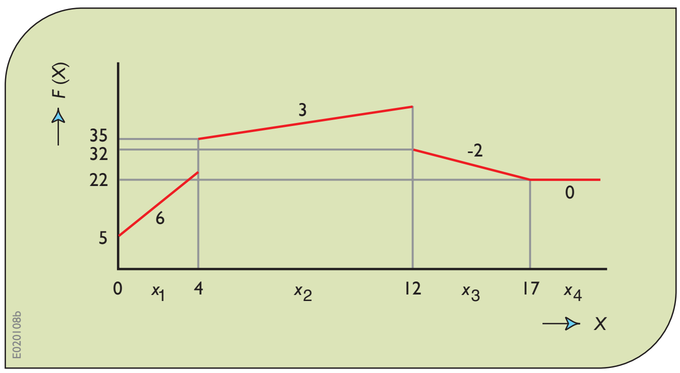
Subject to: $(x_1 + x_2) + (12z_3 + x_3) = X$;

$x_1 \le 4z_1$; $x_2 \le 8z_1$; $x_3 \le 99z_3$; $z_1 + z_3 = 1$;
$z_1, z_3 = 0, 1$.

Minimize $\mathrm{C}(X) \cong (5z_1 + 6x_1) + (29z_2 + 3x_2 + 5x_3)$

Subject to: $x_1 + (4z_2 + x_2 + x_3) = X$;

$x_1 \le 4z_1$; $x_2 \le 8z_2$; $x_3 \le 99\ z_2$; $z_1 + z_2 \le 1$;
$z_1, z_2 = 0, 1$.



Maximize or Minimize $F(X)$

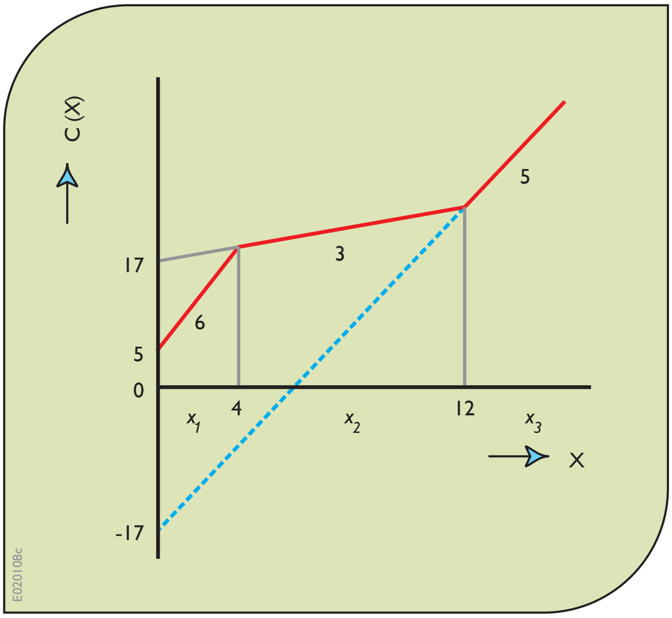
$$F(X) \cong (5z_1 + 6x_1) + (35z_2 + 3x_2) + (32z_3 - 2x_3) + 22z_4$$

Subject to:

$$x_1 + (4z_2 + x_2) + (12z_3 + x_3) + (17z_4 + x_4) = X;$$

$$x_1 \leq 4z_1; \quad x_2 \leq 8z_2; \quad x_3 \leq 5z_3; \quad x_4 \leq 99z_4;$$

$$\sum_s z_s = 1; \quad z_s = 0, 1 \ \forall s.$$



$$\text{Maximize } C(X) \cong (5z_1 + 6x_1 + 3x_2) + (-17z_3 + 5x_3)$$

Subject to: $(x_1 + x_2) + x_3 = X$; $z_1, z_3 = 0, 1.$

$$x_1 \leq 4z_1; \quad x_2 \leq 8z_1; \quad x_3 \leq 99z_3; \quad z_1 + z_3 = 1;$$

$$\text{Minimize } C(X) \cong (5z_1 + 6x_1) + (17z_2 + 3x_2 + 5x_3)$$

Subject to: $x_1 + (4z_2 + x_2 + x_3) = X$; $z_1, z_2 = 0, 1.$

$$x_1 \leq 4z_1; \quad x_2 \leq 12z_2; \quad x_3 \leq 99z_2; \quad z_1 + z_2 \leq 1$$

6. A Brief Review

Before proceeding to some other optimization and simulation methods in the following chapters, it may be useful to review the topics covered so far. The focus has been on model development as well as model solution. Several types of water resources planning and management problems have been used to illustrate model development and solution processes. Like their real-world counterparts, the example problems all had multiple unknown decision-variables and multiple constraints. Also like their real-world counterparts, there are multiple feasible solutions to each of these problems. Hence, the task is to find the best solution, or a number of near-best solutions. Each solution must satisfy all the constraints.

Constraints can reflect physical conditions, environmental regulations and/or social or economic targets. Especially with respect to environmental or social conditions and goals, it is often a matter of judgement to decide what is considered an objective that is to be minimized or maximized and what is considered a constraint that has to be met.

Except for relatively simple problems, the use of these optimization models and methods is primarily for reducing the number of alternatives that need to be further analysed and evaluated using simulation methods. Optimization is generally used for preliminary screening – eliminating inferior alternatives before more detailed analyses are carried out. Presented were some approaches to preliminary screening involving calculus-based Lagrange multiplier and non-linear programming methods, discrete dynamic programming methods, and linear programming methods. Each method has its strengths and limitations.

The example problems used to illustrate these modelling and model solution methods have been relatively simple. However, simple applications such as these can form the foundation of models of more complex problems, as will be shown in later chapters.

7. Reference

BELLMAN, R.E. 1957. *Dynamic programming*. Princeton, N.J., Princeton University Press.

Additional References (Further Reading)

BASSSON, M.S.; ALLEN, R.B.; PEGRAM, G.G.S. and VAN ROOYEN, J.A. 1994. *Probabilistic management of water resource and hydropower systems*. Highlands Ranch, Colo., Water Resources Publications.

BEROGGI, G.E.G. 1999. *Decision modelling in policy management: an introduction to analytic concepts*. Boston, Mass., Kluwer Academic.

BLANCHARD, B.S. and FABRYCKY, W.J. 1998. *Systems engineering and analysis*, 3rd edn. Upper Saddle River, N.J., Prentice Hall.

BURAS, N. 1966. *Dynamic programming and water resources development, advances in hydroscience*, Vol. 3. New York, Academic Press.

DORFMAN, R.; JACOBY, H.D. and THOMAS, H.A. Jr. 1972. *Models for managing regional water quality*. Cambridge, Mass., Harvard University Press.

ESOGBUE, A.O. (ed.). 1989 *Dynamic programming for optimal water resources systems analysis*. Englewood Cliffs, N.J., Prentice Hall.

HALL, W.A. and DRACUP, J.A. 1970. *Water resources systems engineering*. New York, McGraw-Hill.

HILLIER, F.S. and LIEBERMAN, G.J. 1990. *Introduction to operations research*, 5th edn. New York, McGraw-Hill.

HILLIER, F.S. and LIEBERMAN, G.J. 1990. *Introduction to stochastic models in operations research*. New York, McGraw-Hill.

HUFSCHMIDT, M.M. and FIERING, M.B. 1966. *Simulation techniques for design of water-resource systems.* Cambridge, Mass., Harvard University Press.

KARAMOUZ, M.; ZINSSER, W.K. and SZIDAROVSZKY, F. 2003. *Water resources systems analysis*. Boca Raton, Fla., Lewis.

LOUCKS, D.P.; STEDINGER, J.S. and HAITH, D.A. 1981. *Water resource systems planning and analysis*. Englewood Cliffs, N.J., Prentice Hall.

MAASS, A.; HUFSCHMIDT, M.M.; DORFMAN, R.; THOMAS, H.A. Jr.; MARGLIN, S.A. and FAIR, G.M. 1962. *Design of water-resource systems*. Cambridge, Mass., Harvard University Press.

MAJOR, D.C. and LENTON, R.L. 1979. *Applied water resources systems planning*. Englewood Cliffs, N.J., Prentice Hall.

MAYS, L.W. and TUNG, Y.-K. 1992. *Hydrosystems engineering and management*. New York, McGraw-Hill.

NEWNAN, D.G.; LAVELLE, J.P. and ESCHENBACH, T.G. 2002. *Essentials of engineering economic analysis*, 2nd edn. New York, Oxford University Press.

RARDIN, R.L. 1998. *Optimization in operations research*. Upper Saddle River, N.J., Prentice Hall.

REVELLE, C. 1999. *Optimizing reservoir resources*. New York, John Wiley.

SOMLYODY, L. 1997. Use of optimization models in river basin water quality planning: WATERMATEX'97. In: M.B. Beck and P. Lessard (eds.), *Systems analysis and computing in water quality management: towards a new agenda*. London, IWA.

TAYLOR, B.W. III. 1999. *Introduction to management science*, 6th edn. Upper Saddle River, N.J., Prentice Hall.

WILLIS, R. and YEH, W.W.-G. 1987. *Groundwater systems planning and management*. Englewood Cliffs, N.J., Prentice Hall.

WINSTON, W.L. 1994. *Operations research: applications and algorithms*, 3rd edn. Boston, Mass., Duxbury.

WINSTON, W.L. and ALBRIGHT, S.C. 2001. *Practical management science*, 2nd edn. Pacific Grove, Calif., Duxbury.

WRIGHT, J.R. and HOUCK, M.H. 1995. Water resources planning and management. In: W.F. Chen (ed.), *Civil engineering handbook*, Chapter 37. Boca Raton, CRC.

WURBS, R.A. 1996. *Modelling and analysis of reservoir system operations*. Upper Saddle River, N.J., Prentice Hall.