Chapter 5

Water Allocation Model

This chapter introduces the method implemented in Aquarius to solve a water allocation problem. Following definition and mathematical formulation of the problem, the chapter presents details of the sequential optimization technique that allocates the water throughout the flow network. A list with operational constraints that typically restrict reservoir storage and release volumes is presented. The chapter ends with an accounting of the criteria used to find an initial feasible solution and a discussion of the lengths of the time intervals used for simulating a system's operation.

Problem Statement

The aim in using a limited resource, such as water in a river basin, is to realize the greatest possible value from its allocation over some time period consistent with operational and institutional constraints and with the highest reliability. In this model "value" is represented by economic benefit functions that express society's willingness to pay for the various water uses. These benefit functions may be downward sloping and nonlinear. However, the model can be adapted to implement other definitions of value. For instance, the user may wish to reflect a set of priorities such as stipulated by the Doctrine of Prior Appropriation. In this case, the demand functions would be specified as horizontal, with the constant prices of those functions set to represent the seniorities of the demand areas holding the water rights. Allowing for capability of a downward sloping, nonlinear demand function offers the greatest flexibility to the user.

In modeling an actual allocation problem, it is imperative that problem formulation, particularly the objective function, retain the essential characteristics of the system being modeled. Realistic conceptualization of a water allocation problem typically requires relatively complex model formulations. A common source of this complexity is the nonlinear nature of production and benefit functions. Nonlinear optimization problems are commonly encountered in water resources applications. Besides the nonlinearities introduced in the model by hydroelectric power production, the economic benefits of several different water uses are often best represented by nonlinear benefit functions.

Aquarius uses optimization tools to provide the analyst with a way to find the optimal strategies for allocating water among the several uses. The expression "optimal" is used here in a restrictive sense, to indicate a water allocation strategy that is best with respect to the physical and economic criteria specified in a particular model formulation. The model can also analyze alternative operations, the results of which can be compared to aid the decision maker in selecting the most socially desirable policy.

Objective Functional for Optimal Allocation

In Aquarius, water allocation throughout a system and for an entire planning horizon is based on a global objective; to maximize the sum of all economic benefits from the instream and offstream water use. The benefit functions for each of the system components were developed in Chapter 4. What remains is to combine those individual benefit functions *B* into a total benefit function *TB* that reflects all water uses *u* in the basin, HPW^{VH}, HPW^{FH}, IRR, M&I, IRA and RRA, and all time periods *i* in the planning horizon (see also (2.2)). The overall objective is to maximize the total benefit function *TB*:

$$\underset{\mathbf{x}}{maximize}\ \ TB(\$) = \sum_{i=1}^{np} \sum_{u=1}^{nu} B_i^u \tag{5.1}$$

where **x** denotes the set of control variables, *nu* is the number of water uses generating revenue in the basin, and *np* is the number of time periods (optimization horizon).

Equation (5.1) considers only benefits from water use. Costs of water use, such as the cost of operating a hydroelectric plant or of constructing an irrigation canal, are not explicitly considered in the model. The model could be used to evaluate net benefits (i.e., the difference between benefits and costs) by subtracting costs directly from the benefits in the individual benefit functions; essentially making each benefit function a net benefit function.

The problem is to maximize the general nonlinear objective function (5.1), also expressed as $f(\mathbf{x})$, subject to physical, operational, and institutional restrictions such as:

- reservoir storage limitations,
- firm water supply,
- seasonality of water supply,
- firm energy production,
- max/min instream flows, and
- max/min offstream diversions.

Mathematically, these restrictions can be expressed as three types of constraints:

- equality constraints $\sum_{n=1}^{N} a_{kn} \mathrm{x_n} = r_k$ *for* $k = 1, 2, \ldots,$. (5.2a)
- inequality constraints $\sum_{n=1}^{N} a_{kn} \mathrm{x_n} \geq r_k$ *for* $k = K_{e+1}, \ldots, K$ (5.2b)
- bounded variables $\mathrm{x_n^L} \leq \mathrm{x_n} \leq \mathrm{x_n^U}$ *for some of the* x_n (5.2c)

where **x** denotes decision variables, N is the total number of decision variables, K_e is the number of equality constraints and K is the total number of constraints. Except for the nonlinear objective function, the problem above is similar to a standard linear programming problem. The function $f(\mathbf{x})$ can be any type of nonlinear function subject to the requirement of being continuous and differentiable.

Solution Method

There are a variety of approaches for solving the above problem, none of which is uniquely superior. The solution technique implemented in Aquarius takes advantage of the special case of the general nonlinear programming problem that occurs when the objective function is reduced to a quadratic form and all the constraints are linear. Although a quadratic function is the simplest nonlinear approximation that can be used for a nonlinear objective, it is suitable for solving multi-reservoir optimization problems with hydropower generation (Díaz and Fontane 1989). Furthermore, model development indicated that, because of the particular structure of the water allocation problem, a quadratic approximation of the objective function is advantageous from the computational viewpoint. A quadratic approximation is a close representation of the nonlinear objective function defined in (5.1) and also permits larger, valid changes in the control variables at each step of the solution in comparison to a simple linear approximation. This results in a faster convergence to the optimal solution and lower risk of solution divergence (Díaz and Fontane 1989)

The method approximates the original nonlinear objective function with a quadratic equation using Taylor series expansion and then solves the problem as a quadratic programming (QP) problem. Starting with an initial feasible solution, $\mathbf{x}^o$, the algorithm carries out a Taylor series expansion on the nonlinear objective function around the given initial solution, retaining the first and second order terms to form a quadratic function. The general Taylor series expansion of $f(\mathbf{x})$ truncated beyond the second order terms yields:

$$f(\mathbf{x}) \approx f(\mathbf{x}^\circ) + \nabla f(\mathbf{x}^\circ)^T \partial\mathbf{x} + \frac{1}{2} \partial\mathbf{x}^T \mathbf{H}(\boldsymbol{x}^\circ) \partial\mathbf{x} \tag{5.3}$$

where $\partial\mathbf{x}$ is the increment vector and $\nabla f(\mathbf{x}^o)$ is the gradient vector of $f(\mathbf{x})$ measured at $\mathbf{x}=\mathbf{x}^o$. The gradient is computed by the first partial derivatives evaluated at $\mathbf{x}=\mathbf{x}^o$,

$$\nabla \boldsymbol{f} = \mathrm{col}(\partial f/\partial x_1, \partial f/\partial x_2, ..., \partial f/\partial x_N) \tag{5.4}$$

The remaining term, $\mathbf{H}(\mathbf{x}^o)$, denotes the Hessian matrix, which is a real matrix whose elements are evaluated at the same point $\mathbf{x}=\mathbf{x}^o$,

$$\mathbf{H} = \begin{vmatrix} \frac{\partial^2 f}{\partial x_1 \partial x_1} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots\cdots & \frac{\partial^2 f}{\partial x_1 \partial x_N} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2 \partial x_2} & \cdots\cdots & \\ \cdots\cdots & & & \\ \frac{\partial^2 f}{\partial x_N \partial x_1} & \frac{\partial^2 f}{\partial x_N \partial x_2} & \cdots\cdots & \frac{\partial^2 f}{\partial x_N \partial x_N} \end{vmatrix} \tag{5.5}$$

To achieve the optimization, the quadratic approximation (5.3) of the nonlinear objective should conform to the standard form of a quadratic programming problem, as defined by:

$$\underset{\mathbf{x}}{Max} \left[f(\mathbf{x}) = w + \boldsymbol{c}^{\mathrm{T}} \mathbf{x} + \frac{1}{2} \mathbf{x}^{\mathrm{T}} \mathbf{Q}\, \mathbf{x} \right] \tag{5.6a}$$

- subject to the linear constraints $\boldsymbol{g}(\mathbf{x}) = \mathbf{A}\,\mathbf{x} \geq \mathbf{r}$ (5.6b)
- and nonnegativity conditions $\mathbf{x} \geq \mathbf{0}$ (5.6c)

where w is a scalar, the **c** and **r** vectors have known components, **A** is the matrix of constraint coefficients, and **Q** is a square matrix of dimension (NxN). The components of w, the **c** vector, and the **Q** matrix can be obtained by equating (5.3) and (5.6a). Details of the derivation are in Díaz and Fontane (1989). The final expressions are:

$$w = f(\mathbf{x}^{\circ}) - \sum_{i=1}^{N} D_i \mathrm{x}_i^{\circ} + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \mathrm{H}_{i,j}\, \mathrm{x}^{\circ} \mathrm{x}_j^{\circ} \tag{5.7}$$

$$c_i = D_i - \sum_{j=1}^{N} H_{i,j}\, \mathrm{x}_j^{\circ} \qquad \textit{for } i = 1,2,....,N \tag{5.8}$$

$$\mathrm{q}_{i,j} = \mathrm{H}_{i,j} \qquad \textit{for } i = 1,2...,N \textit{ and } j = 1,2...,N \tag{5.9}$$

where the only new term is D_i , which equals the ith element of the gradient vector ($\partial f/\partial \mathrm{x}_i$). Once the equivalent **c** vector and **Q** matrix have been explicitly defined, a standard QP code can be used to solve for the set of **x** values that maximizes the objective function.

An efficient QP code is a basic requirement for the success of the proposed solution method. The routine QPTHOR, based on the General Differential Algorithm (Wilde and Beightler 1967) and further developed by Leifsson and Morel-Seytoux (1981), is used in Aquarius.

The optimal solution obtained by standard QP is only true for the approximated objective function (5.3). Because the optimal values of the variables may differ from the initial values upon which the approximation of the nonlinear quadratic objective function was based, it is necessary to repeat the process using the new values for the set of variables as the starting point for the next round of the sequential solution. A succession of these approximations is performed until the solution of the quadratic programming problem reaches the optimal solution, which is when successive optimal values do not differ by more than the stipulated tolerance limit, or when the maximum limit on the number of iterations is reached. Figure 5.1 illustrates the sequential procedure of successively solving quadratic programming problems, known as Sequential Quadratic Programming (SQP). The SQP approach implemented in Aquarius is an extension of the work reported by Díaz and Fontane (1989) and Díaz et al. (1992).

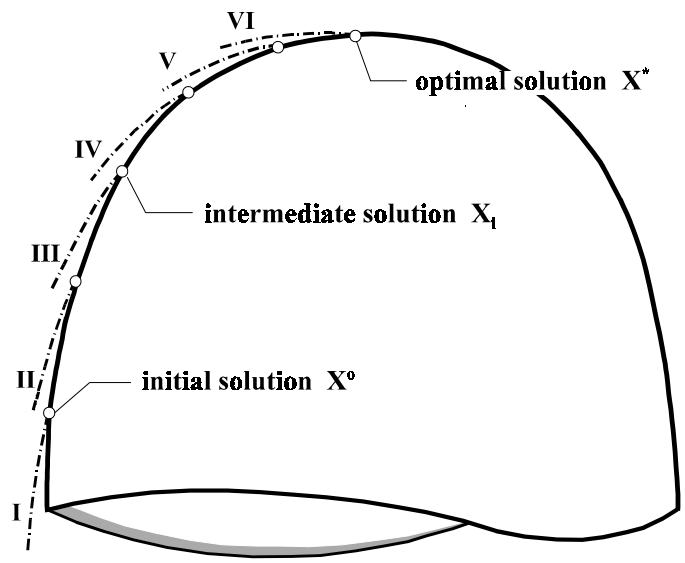


Figure 5.1 Sequential maximization of a concave objective function by Sequential Quadratic Programming (SQP) (Díaz and Fontane 1989).

Linear approximation of the nonlinear objective function (5.1), using sequential linear programming (SLP), is also a viable alternative to solve the water allocation problem with Aquarius. SLP convergency to the optimal solution is only possible when a step-bound solution scheme in the control variables is implemented (Palacios Gomez et al. 1982). This causes SLP to have a slower rate of convergency toward the optimal solution than SQP (Díaz and Fontane 1989). Morever, optimal values tend to be slightly lower with SLP than with SQP, with the

difference more pronounced as the complexity of the water allocation problem increases. Nevertheless, for small flow networks or when only some of the variables in the objective function are nonlinear, SLP is an efficient algorithm.

Objective Function Differentiation

The gradient vector in (5.3) and the Hessian matrix in (5.5) require computation of first, second, and second-cross partial derivatives of the global objective function with respect to the decision variables **x**. For a water allocation problem, which may involve many control variables and complex objective functions, a finite difference scheme may appear to be an expeditious procedure for the differentiation. However, numerical differentiation is inherently inaccurate, particularly for high-order differentiation.

In contrast, differentiation via calculus provides the exact results for partial derivatives of any order. Accuracy of the computations is important if the QP algorithm is to arrive at an exact solution. Calculus also reduces the computation time drastically. The algebraic derivation of the partial derivatives using calculus was an involved and time consuming task as the model had to consider all possible water uses generating revenue in the basin and all possible ways in which they could interrelate within a flow network. The derivative formulas are in Appendix A. Once the formulas are completed, assembling the gradient vector and Hessian matrix for any user-defined network topology is greatly eased, partly due to the object-oriented modeling framework adopted. This subject, called the mathematical connectivity of the network, is covered in Chapter 6. The derivatives in Appendix A are organized by type of price function first (downward sloping benefit functions and constant price functions) and by water user second. For each type of benefit function, several partial derivatives are considered based on all possible control variable roles in a flow network.

Operational Restrictions

As indicated, the model maximizes total return over a planning horizon, subject to all required operational restrictions. These restrictions, or constraints, represent physical limits in the operation of reservoirs, powerplants, diversion structures, and other system components. We present a list of basic operational constraints that are essential for the proper simulation of a water system.

Given the multi-site and multipurpose nature of the formulation, it is necessary to adopt a notation to help us distinguish between variables associated with different system components (objects) and the relative location of the system components in the flow network. The following notations complement those in the Storage Reservoirs section of Chapter 3:

d = decision variable (controlled flow)
d_M = upper bound of decision variable
d_m = lower bound of decision variable
NF = natural (uncontrolled) flow
S = reservoir storage
S^o = initial reservoir storage
S^f = final reservoir storage
S_M = maximum reservoir operational storage
S_m = minimum reservoir operational storage
E = net reservoir evaporation
L = reservoir spillage
IF = instream flow
IF_M = maximum instream flow
IF_m = minimum instream flow
OF_M = maximum offstream water supply
OF_m = minimum offstream water supply

Because of the generalized model character, the constraint equations presented below are not for any specific topology of a flow network, but rather are generally formulated consistent with the object-oriented approach used throughout this model. Each system component (i.e., each model object) may have one or more constraints associated with it. Constraint equations can be either equalities or inequalities depending on the type of object and the nature of the constraint being imposed.

The superscript u denotes a variable that originates in the portion of the network upstream from the object under consideration. The superscript s denotes a variable that outflows from the same object being analyzed (figure 5.2). The subscript i denotes the time period. For instance, i=3 indicates the third time interval of the optimization horizon. For all variables except reservoir storage, the index i indicates the variable value during the time interval such as inflow to a reservoir during the third time interval. When referring to reservoir storage, the index i should be interpreted as the storage at the end of the time period i.

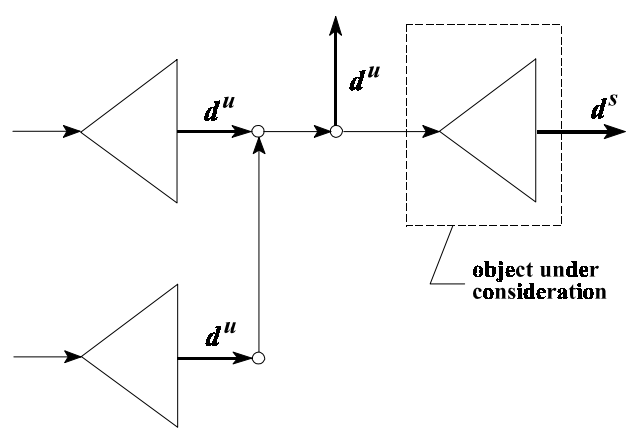


Figure 5.2 Sets of decision variables in relation to the object under consideration.

There may be multiple sources of controlled and uncontrolled flows converging at a given system component. For example, figure 5.2 depicts a network with three d^u sets that are the controlled inflows to the downstream reservoir. Although not shown in the figure, this may also occur with upstream mandatory releases, spillages, and natural flows. To simplify the notation, this situation is denoted in the constraint equations below by underscoring the symbol representing the variables cited above. The signs of the decision sets d^u and d^s (i.e., the sign of the coefficients of the decision sets) and all other terms in the constraint equations are automatically given by the model. In general, signs will be positive or negative depending on whether the variable of interest contributes flow to or removes flow from the object under consideration (this is the water-balance concept). More details concerning the mathematical connectivity of system components are in Chapter 6, Mathematical Connectivity of System Components.

All restrictions are written using the canonical form of expressing a mathematical programming problem. The left-hand sides of the constraint equations are the decision variables (unknown terms). The right-hand sides contain the uncontrolled (known or assumed known) terms. The expressions include two parameters undefined yet, γ and w, which are derived from the so called "explicit modeling of reservoir evaporation". The way γ and w are calculated is presented later in this chapter. The system restrictions available are:

Maximum reservoir storage: ensures that the storage at the end of any time period *i* does not exceed the reservoir's maximum operational capacity (i.e., $S_i \leq S_M$) for $i = 1, 2, ..., np$, where *np* is the total number of periods:

$$\sum_{k=1}^{i} \gamma_k^i \underline{d}_k^u + \sum_{k=1}^{i} \gamma_k^i \underline{d}_k^s \geq (\gamma_0^i S^o - S_M) + \sum_{k=1}^{i} \gamma_k^i (\underline{NF}_k^u + \underline{L}_k^u) - \sum_{k=1}^{i} \gamma_k^i (L_k^s + w_k^s) \tag{5.10}$$

Minimum reservoir storage: ensures that the storage at the end of any time period *i* does not fall below the reservoir's minimum active capacity (i.e., $S_i \geq S_m$) for $i = 1, 2, .. np$:

$$\sum_{k=1}^{i} \gamma_k^i \underline{d}_k^u + \sum_{k=1}^{i} \gamma_k^i \underline{d}_k^s \geq (S_m - \gamma_0^i S^o) - \sum_{k=1}^{i} \gamma_k^i (\underline{NF}_k^u + \underline{L}_k^u) + \sum_{k=1}^{i} \gamma_k^i (L_k^s + w_k^s) \tag{5.11}$$

Reservoir final storage: imposes a final reservoir storage equal or above a certain level S^f. This prevents the model from generating extra benefits at the expense of depleting a reservoir's storage at the end of the optimization horizon:

$$\sum_{k=1}^{np} \gamma_k^i \underline{d}_k^u + \sum_{k=1}^{np} \gamma_k^i \underline{d}_k^s = (S^f - \gamma_0^{np} S^o) - \sum_{k=1}^{np} \gamma_k^i (\underline{NF}_k^u + \underline{L}_k^u) + \sum_{k=1}^{np} \gamma_k^i (L_k^s + w_k^s) \tag{5.12}$$

Maximum instream flow: ensures that flows do not exceed a specified maximum, as may be required for instream water-related activities such as recreation or environmental protection (i.e., $IF \leq IF_M$):

$$\underline{d}_i^u \geq -IF_{M_i} + (\underline{NF}_i^u + \underline{L}_i^u) \qquad for\ i = 1,..,np \tag{5.13}$$

Minimum instream flow: ensures that flows do not fall below a specified minimum, as may be required for instream water-related activities such as recreation or environmental protection (i.e., $IF \geq IF_m$):

$$\underline{d}_i^u \geq IF_{m_i} - (\underline{NF}_i^u + \underline{L}_i^u) \qquad for\ i = 1,..,np \tag{5.14}$$

Diversion node: ensures that the flow diverted from a diversion node does not exceed the incoming flow:

$$\underline{d}_i^u + \underline{d}_i^s \geq -(\underline{NF}_i^u + \underline{L}_i^u) \qquad for\ i = 1,..,np \tag{5.15}$$

Maximum offstream flow: ensures that flow supplied to an irrigation or urban area does not exceed a specified maximum (i.e., $OF \leq OF_M$):

$$\underline{d}_i^u \geq -OF_{M_i} + (\underline{NF}_i^u + \underline{L}_i^u) \qquad for\ i = 1,..,np \tag{5.16}$$

Minimum offstream flow: ensures that flow supplied to an irrigation or urban area does not fall below a specified minimum (i.e., $OF \geq OF_m$):

$$\underline{d}_i^u \geq OF_{m_i} - (\underline{NF}_i^u + \underline{L}_i^u) \qquad for\ i = 1,..,np \tag{5.17}$$

Seasonality of water demand: ensures that water deliveries to demand areas follow a user defined seasonal pattern (see figure 3.7). The constraint is applicable for time periods $i = 1, 2, ..., npc$, where npc denotes the total number of periods within the annual cycle ($npc=12$ for monthly intervals). To activate this constraint, the optimization horizon should encompass entire annual cycles:

$$\underline{d}_i^u - \alpha_i \sum_{k=1}^{npc} \underline{d}_k^u = -\underline{UI}_i^u + \alpha_i \sum_{k=1}^{npc} \underline{UI}_k^u \qquad for\ \ 0 \leq \alpha_i \leq 1\ \ and\ \sum_{k=1}^{npc} \alpha_k = 1 \tag{5.18}$$

Annual firm water supply: enforces firm levels of annual supply to offstream demand areas, where *AFW* represents the contracted annual volume and *npc* denotes the number of periods within the annual cycle. To activate this constraint, the optimization horizon should encompass entire annual cycles:

$$\sum_{k=1}^{npc} \underline{d}_k^u \geq AFW - \sum_{k=1}^{npc} \underline{UI}_k^u \tag{5.19}$$

The sequential-approximation algorithm SQP is well suited for solving this water allocation problem, which includes only linear constraints. However, the algorithm can be extended to problems with nonlinear constraint functions by using linear approximations of the nonlinear constraint equations (not included in the present version of the model).

In the formulation of the water allocation problem, the model automatically includes the restrictions in equations (5.10) through (5.12). The user has the option to enable or disable restrictions by toggling on/off the corresponding check-boxes in the input dialog boxes. Even a mid-size network may include hundreds of constraint equations, depending on the number of system components and the length of the optimization horizon.

Controlling flows and storage are not the only restrictions commonly found in the operation of real-world water systems. For instance, the two restrictions below, which are related to hydropower production, will be incorporated into the model in a future version:

Hydropower Firm Energy. This constraint would be used to guarantee that a reservoir/ powerplant subsystem were operated to guarantee the delivery of a preestablished amount of electrical energy (megawatt-hours) with assured availability to the electrical network. Firm-energy levels could be demanded at each powerplant separately or from the system of powerplants as a whole depending on ownership and agreements among the electrical facilities within the river basin.

Maximum Energy Sale. This constraint would allow the analyst to impose a maximum value to the total amount of energy, per simulation period or over the year, that the hydro system could sell to the electrical grid. Nevertheless, the user can limit the powerplant discharges as an approximate way to control the electrical output generated.

Modeling Uncontrolled Releases

The term uncontrolled releases *UR* (see figure 3.1 and equation (3.3)) refers to the volume of water evaporated from the reservoir *E* and the spillway releases at the dam *L*. Previous versions of the model use a simplified approach for handling such losses, which treats *E* and *L* as constants for each solution of the QP problem. These constant values are based on the result of the previous QP solution. By treating evaporation and spills as constant terms in the formulation, the nonlinear relationships between evaporation-storage and outflow-storage, typical of the reservoir dynamics, are omitted, yielding a strictly linear set of constraints.

However, moving reservoir losses to the right side of the constraint equations gives no control to the model over those outflows, causing *E* and *L* to assume a passive role in the optimization. Version 2000 introduces an alternative formulation of reservoir evaporation in which evaporation losses are included "explicitly" in the formulation of the reservoir dynamics, rather than "implicitly". This more elaborate formulation gives the model the capability to anticipate periods with extreme evaporation losses and affect the operation of reservoirs accordingly. Furthermore, version 2000 also presents improvements in the way reservoir spillages are treated. Both alternative formulations are introduced below.

Implicit Formulation of Reservoir Evaporation

The consequences of assuming known values for evaporation when solving the QP problem are: 1) the optimal reservoir storage provided by each sequence of the SQP process (i.e., by each individual QP solution) must be adjusted using an iterative reconciliation procedure to find accurate estimates of *E*; and 2) because new values of *E* were obtained from the previous step, all RHSs must be recomputed before going to the next optimization sequence. The iterative reconciliation procedure, performed after each QP solution, works as follows: 1) based on the last optimal state of the reservoirs, evaporation losses are computed; 2) because the new evaporation losses probably differ from the old ones (those adopted when solving the last QP problem), new (adjusted) reservoir storages will result; 3) based on these adjusted reservoir storages, evaporation losses are recomputed; 4) step 3 is repeated until *S* and *E* values are reconciled for all reservoirs; that is, until the difference in storages between two consecutive iterations is below a given threshold, which is a user defined parameter.

Such computations raise the following question: Will a QP solution that is feasible with respect to its constraint set (i.e., the set containing *E* from the previous QP solution) become infeasible in the new constraint set? To answer this question, consider the hypothetical case of a reservoir storage trajectory constrained by upper and lower bounds in figure 5.3. The thin solid line (old solution) represents the optimal trajectory derived from the prior QP solution, as adjusted using the iterative reconciliation procedure. This solution is used as the initial feasible solution for the next QP solution, which yields a new storage trajectory represented by the dashed line (new solution). For the portion of the trajectory that is successively moving toward the lower bound (left side of the graph), the water losses during computation of the new QP solution (from the old

solution) are larger than the true losses that would occur at the new solution storage level (i.e., the surface area of the reservoir decreases). After evaporation losses and the storage level agree the new optimal trajectory will recede inward because lowering the evaporation loss will increase the remaining storage. This is shown by the thick solid line (adjusted new solution) on the graph. Similarly, for the portion of the trajectory that moves closer to the upper bound (right side of the

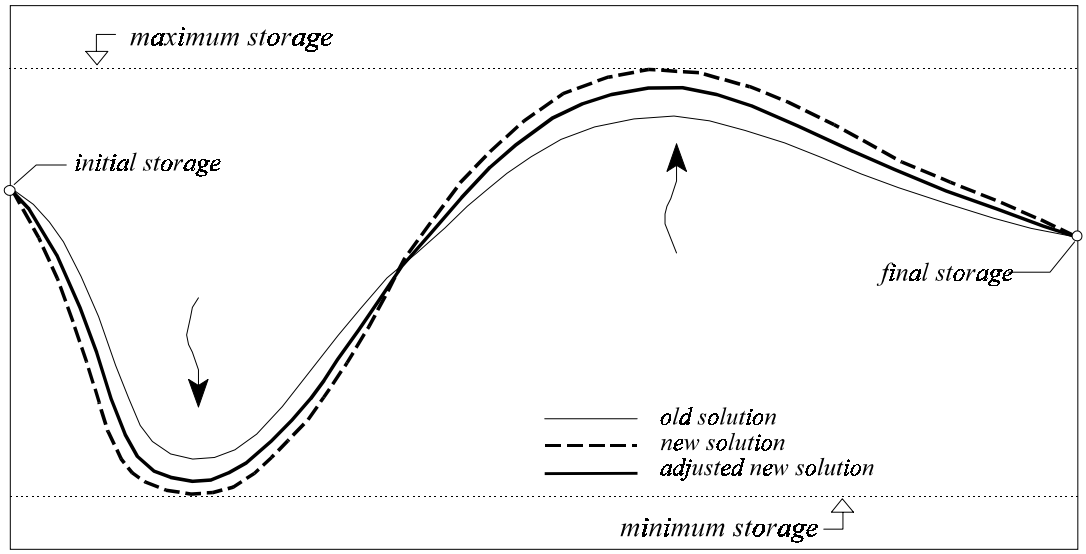


Figure 5.3 A storage trajectory adjusted for evaporation losses between Quadratic Programming solutions.

graph), the water losses during computation of the new QP solution are smaller than the true losses that would occur at the new storage level because the new storage level is higher than that of the old solution from which the assumed losses were taken. After bringing storage and losses into agreement using the iterative procedure, the new optimal trajectory will recede inward as indicated by the thick solid line. These two cases indicate that the prior optimal solution will remain feasible to the new constraint set, allowing the sequential optimization procedure to continue. The same conclusion is reached when analyzing a sequence of storage trajectories that tend to depart from upper and lower reservoir bounds.

Analysis of this hypothetical case shows that the iterative process by which evaporation losses are evaluated, except under very unusual circumstances, should not cause infeasible solutions as the storage trajectory progresses toward optimality. Furthermore, the sequential nature of the SQP algorithm allows for the convergence of uncontrolled outflows toward their exact values.

Explicit Formulation of Reservoir Evaporation

This section presents an alternative formulation in which evaporation losses are not assumed known at the beginning of each optimization sequence, but are included explicitly in the derivation of the reservoir dynamics. In this manner, the model strives to minimize reservoir evaporation losses at the same time that water demands are meet. This formulation brings about changes in the reservoir storage equation, and consequently, in the expressions for reservoir physical constraints (see equations (5.10) through (5.12)), and in those benefit functions linked to reservoir regulation (see equations (4.9c) and (4.30b)).

Linearization of Reservoir Evaporation

According to Chapter 3, Storage Reservoirs, reservoir evaporation losses are computed by multiplying the surface area of the lake times the seasonal evaporation rate, denoted here by e_i [L/T]. The power function that relates reservoir area A [L^2] versus reservoir storage S [L^3] (equation (3.4b)) can be linearized as shown in figure 5.4 such that evaporation losses E [L^3] during any given time interval i can be expressed as:

$$E_i = e_i A_i = e_i a + e_i b S_i \qquad (5.20)$$

where a and b are the coefficients of the linear relationship, computed automatically by the model using a weighted least-square method. For convenience, (5.20) is rewritten as:

$$E_i = w_i + v_i S_i \qquad (5.21)$$

where $w_i = e_i a$ and $v_i = e_i b$. The proposed linear model between surface area and storage in a reservoir (figure 5.4) is a reasonable approximation in most cases.

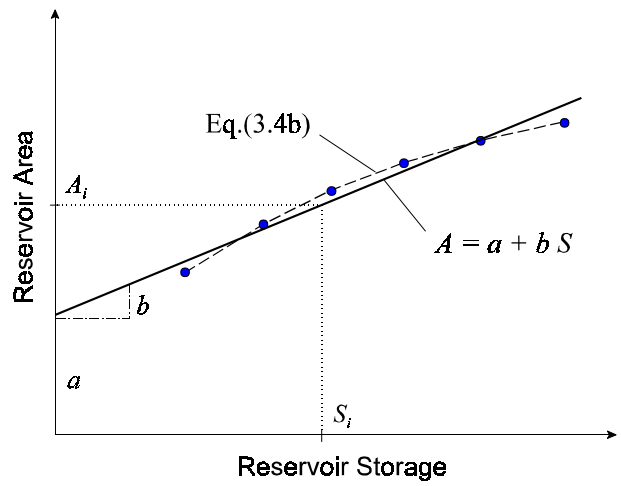


Figure 5.4 Linearization of the reservoir area-storage relation.

Modified Reservoir Storage Equation

Using the general notation introduced in Chapters 3 and 5, and adopting the expressions for inflows and outflows to a reservoir given by (3.2) and (3.3), respectively, the storage of a reservoir at the end of the first period of operation S_1 is expressed as:

$$S_1 = S^o + I_1 - O_1 - [w_1 + v_1 \frac{S^o + S_1}{2}] \tag{5.22}$$

where S^o is the storage of the reservoir at the beginning of the simulation, I_1 indicates all inflows to the reservoir (controlled and uncontrolled), O_1 indicates all outflows from the reservoir (controlled and uncontrolled) except for evaporation losses, which has been written separately within the term in brackets. Evaporation for the period is computed as a function of the average storage during the time interval. Rearranging terms in (5.22) yields:

$$S_1 = \frac{(1 - \frac{v_1}{2})}{(1 + \frac{v_1}{2})} S^o + \frac{1}{(1 + \frac{v_1}{2})} (I_1 - O_1 - w_1) \tag{5.23}$$

in which again variables are grouped to simplify the notation,

$$S_1 = \alpha_1 S^o + \beta_1 (I_1 - O_1 - w_1) \tag{5.24}$$

where $\alpha_i = \dfrac{(1 - \frac{v_i}{2})}{(1 + \frac{v_i}{2})}$ and $\beta_i = \dfrac{1}{(1 + \frac{v_i}{2})}$ (5.25a), (5.25b)

Similarly, the reservoir storage at the end of the second period can be written as:

$$S_2 = \alpha_2 S_1 + \beta_2 (I_2 - O_2 - w_2) \tag{5.26}$$

Substituting S_1 above by the expression in (5.24), and after rearranging terms we obtain,

$$S_2 = \alpha_2 \alpha_1 S^o + \alpha_2 \beta_1 (I_1 - O_1 - w_1) + \beta_2 (I_2 - O_2 - w_2) \tag{5.27}$$

In general, reservoir storage at the end of any time period i can be expressed as:

$$S_i = (\prod_{k=1}^{i} \alpha_k) S^o + (\prod_{k=2}^{i} \alpha_k) \beta_1 (I_1 - O_1 - w_1) + (\prod_{k=3}^{i} \alpha_k) \beta_2 (I_2 - O_2 - w_2) + \ldots\ldots$$
$$\ldots\ldots + (\prod_{k=i}^{i} \alpha_k) \beta_{i-1} (I_{i-1} - O_{i-1} - w_{i-1}) + \beta_i (I_i - O_i - w_i) \tag{5.28}$$

Equation (5.28) can be written in a more compact form in which the products $\prod_k \alpha_k$ and β terms are grouped under the new symbol γ as indicated in (C.10):

$$S_i = \gamma_0^i S^o + \gamma_1^i (I_1 - O_1 - w_1) + \gamma_2^i (I_2 - O_2 - w_2) + \ldots \\ \ldots + \gamma_{i-1}^i (I_{i-1} - O_{i-1} - w_{i-1}) + \gamma_i^i (I_i - O_i - w_i) \tag{5.29}$$

Equation (5.29), which computes reservoir storage at the end of any time interval of simulation as a function of inflows, outflows and evaporation parameters, can also be expressed in matrix form as shown in (5.30):

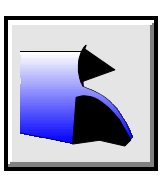
$$\begin{bmatrix} S_1 \\ S_2 \\ S_3 \\ \cdot \\ \cdot \\ S_i \end{bmatrix} = \begin{bmatrix} \gamma_0^1 \\ \gamma_0^2 \\ \gamma_0^3 \\ \cdot \\ \cdot \\ \gamma_0^i \end{bmatrix} S^o + \begin{bmatrix} \gamma_1^1 & 0 & 0 & 0 & 0 & \ldots & 0 & 0 \\ \gamma_1^2 & \gamma_2^2 & 0 & 0 & 0 & \ldots & 0 & 0 \\ \gamma_1^3 & \gamma_2^3 & \gamma_3^3 & 0 & 0 & \ldots & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \gamma_1^i & \gamma_2^i & \gamma_3^i & \gamma_4^i & \gamma_5^i & \ldots & \gamma_{i-1}^i & \gamma_i^i \end{bmatrix} \begin{bmatrix} I_1 - O_1 - w_1 \\ I_2 - O_2 - w_2 \\ I_3 - O_3 - w_3 \\ \cdot \\ \cdot \\ I_i - O_i - w_i \end{bmatrix} \tag{5.30}$$

In order to rebuild the benefit functions for water uses affected by variable water levels in a reservoir (hydropower and reservoir recreation), it is convenient to introduce here the expression for the average reservoir storage during any given time interval *i*:

$$\bar{S}_i = \frac{1}{1 + \frac{v_i}{2}} \left(S_{i-1} + \frac{I_i - O_i - w_i}{2} \right) = \beta_i \left(S_{i-1} + \frac{I_i - O_i - w_i}{2} \right) \tag{5.31}$$

where S_{i-1} is given by equation (5.29) and β_i is introduced to simplify the notation.

Modeling Reservoir Spillages



Aquarius uses a simple approach for modeling reservoir spillage, treating *L* as constant for each solution of the QP problem. The assumption of *L* as constant is clear in the reservoir's constraint equations, where all the known (or assumed known) terms are on the right-hand side. Spillages from each reservoir are detected at the beginning of the calculations by the algorithm that finds the initial feasible solution (IFS) to the water allocation problem at hand (topic covered later in this

chapter). This simple approach has the advantage of not increasing the mathematical dimensionality of the optimization problem. However, moving reservoir losses to the right side of the constraint equations gives no control to the optimization model over those outflows.

Version 2000 incorporates a new system component, termed the "Spill Controller", which helps incorporate IFS spills into the optimization, and potentially minimizes reservoir spillages subject to the reservoir's storage constraints. The spill controller is a pseudo water user that, rather than rendering benefits, imposes an economic penalty to the operation of the system every time a reservoir spills. The spill controller acts upon the reservoir to which it is connected to discourage uncontrolled outflows, and indirectly, over all the system components upstream that provide inflows to the reservoir under consideration. It should be noted that the spill controller does not perform a detailed simulation of the spillway operation. In fact, operation of a gate-controlled spillway is unnecessary for monthly time steps of simulation. For monthly operation the model passes any excess flood water downstream of the reservoir as a block.

Spills are controlled based on seasonal penalty functions of the form:

$$Penalty^{spill}\ [\$/\text{Mcm}] = -a_i \exp(-L_i / b_i) \tag{5.32}$$

where L is the quantity of water spilled by the reservoir (in Mcm) and a and b are the ordinate and the coefficient that controls the rate of decay of the exponential model, respectively. The penalty function is characterized by negative marginal returns to spillages (figure 5.5), such that initial units of spillage are penalized more heavily than additional units (small spillages are eliminated first). The use of an economic penalty function to prevent (or minimize) spillages increases the dimensionality of the optimization problem because a set of controlled variables is assigned to each spill controller.

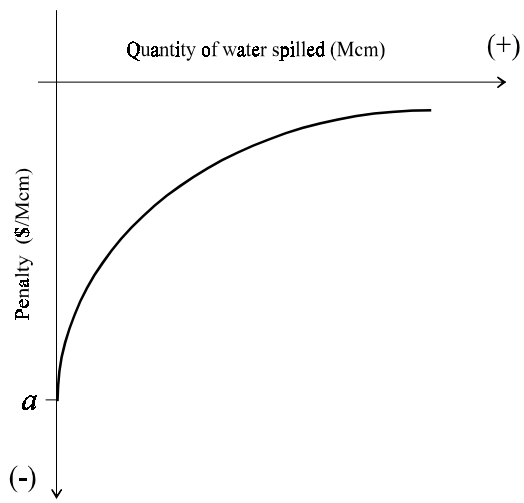


Figure 5.5 Spillages penalty function.

With the spill controller(s) in place, the model drives an initial solution containing spillages toward a different and improved solution where excess water is reallocated over the whole operational horizon in favor of a more profitable water allocation. In this manner, spills from the IFS are converted to the extent feasible (and subject to the penalty function imposed) into storage and early releases, giving the optimization scheme the opportunity to transform the excess volume of water into economically beneficial reservoir releases during other time periods.

The utilization of the spill controller requires some prior familiarization of the analyst with the river system at hand. Because where and when spillages will occur is unknown before hand, the flow network should be solved first without the presence of spill controller(s). If the optimal solution presents no spillages, then the problem has been optimally resolved and no additional runs are needed. However, if spillages are observed in the solution, the user has the option to minimize them by re-running the model including the spill controller object. To limit computation time, a minimum number of spill controllers should be added to the flow network. Typically, there should be one spill controller per river branch in which spillages occur. The magnitude of the coefficients *a* and *b* can be adjusted to minimize spillages until the point that the physical characteristics of the reservoirs prevent further reduction.

The economic effect of the spill penalty function upon water distribution throughout the network is difficult to assess. Hence, it is possible that once all (or most of) the spillages with the presence of the spill controller(s) are eliminated, to set the *a* coefficient of the penalty functions to zero, and continue the search for a new equilibrium (optimal) point. This last water allocation reached with the *a* coefficients equal to zero will not carry distortions introduced by the explicit cost assigned to spills (as the original solution did).

It should be noted that the monetary loss created by the spill controller(s) is never included in the global objective function result reported by the model. Nevertheless, the value of the penalty generated by a spill controller can be known by accessing the economic output from that specific object. At this point the reader may recognize another obvious application of the spill controller in a flow network. We refer to the possibility of using this water component to control flooding in a river reach, minimizing economic losses.

Search for a Feasible Solution

The method of solution used by Aquarius requires an initial feasible solution (IFS) to start the optimization process. This is an important requirement, especially for large, highly constrained water systems. For small flow networks and short periods of analysis might be reasonable to expect the user to provide the required IFS. However, more complex situations such as those that the current model is designed for, require a modeling approach.

Aquarius contains two different approaches to obtaining an initial feasible solution. The first is an expeditious “flow cascading” method with some limitations. Nevertheless, this approach should provide an IFS for most systems and flow conditions. The second approach is a more robust method based on the same concepts as the “Two-Phase Method of Linear Programming”, to the expense of long execution time. If the flow cascading method failed to provide the IFS, the second approach would complete the task. Because of the long running time of the two-phase method, version 2000 offers only the flow cascading approach. However, both methods are described below.

If, due to intricacies of the flow network, the model is unable to find an IFS that fully complies with the originally stipulated constraints, the analyst may relax restrictions that impede finding an initial solution. After an IFS is found with the relaxed set of constraints, the analyst will have an opportunity to gradually steer the water allocation process in the appropriate direction during the optimization process. This is possible because the water allocation problem is formulated using economic variables that can be manipulated during the sequential optimization process without affecting the feasibility of any intermediate solution. Once the necessary level of flow is attained as a result of the modified economic conditions, the optimization can be interrupted, the relaxed constraints reinstated, and the original price structure restored, allowing the model to continue with the optimization until the final optimal solution is reached. (Note: the feature just described is still not available in V2000.)

Flow Cascading

The flow cascading approach consists of routing the natural flows originating in the basins downstream through the flow network. This is done without any operational rules. The routing algorithm starts at each water source (i.e., a natural flow basin object) and moves downstream through the network always following the main water course. Computationally, this is a pass of the algorithm. As the water travels downstream in the network, it satisfies minimum instream and offstream demands, provided that enough water is available. A demand area that is only partially satisfied after the first pass may become fully satisfied after the second or third pass, depending on the number of water sources being routed through that specific network component. For example, in figure 5.6 only two passes of the algorithm (I and II) provide water to the offstream irrigation area since the third water source is located downstream from the demand zone.

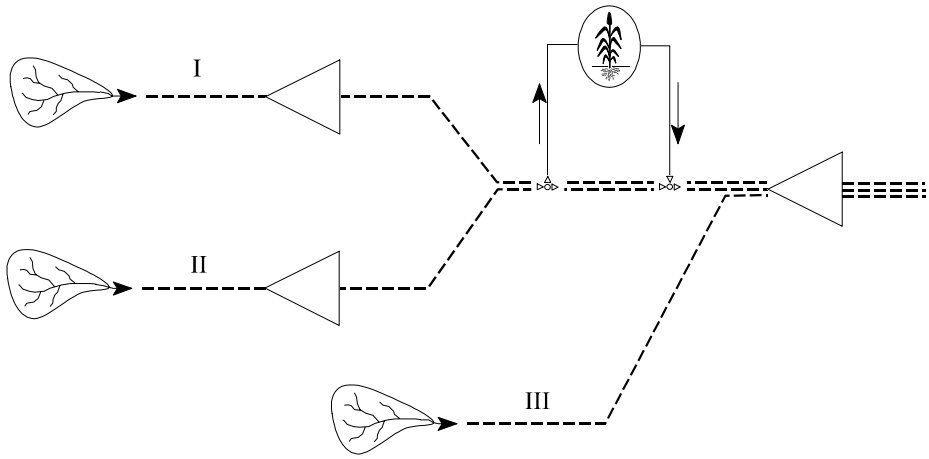


Figure 5.6 Cascading flows through a network to find an initial feasible solution.

Inflows to a reservoir are passed downstream from the structure and used to satisfy reservoir demands. Minimum flow requirements are satisfied first, and any remaining water is arbitrarily assigned to any of the water users connected to the reservoir until maximum capacity is reached. This occurs without any regulation of reservoir storage. However, for very large inflows and when the capacity of the demand areas to accept flow is exceeded, the algorithm regulates reservoir storage by pre-emptying the reservoir during time periods prior to the period with excessive inflows. If storage regulation alone is unable to contain the inflows, the reservoir is forced to spill.

The cascading approach may find an IFS, depending on how tightly constrained the water system is given water availability in the basin. In general, normal and wet flow conditions are best for a successful run. If flow conditions are very dry, relaxation of constraints, such as specific minimum flow requirements, increases the chance of a successful run.

Phase 1 of the Two-Phase Method

To find an initial feasible solution as required by the General Differential Algorithm (QPTHOR), it is possible to use the same procedures as the Simplex Method of Linear Programming (LP) uses for the same purpose. Some of these procedures involve use of artificial variables to obtain an initial basic feasible solution to a slightly modified set of constraints (although for QPTHOR, the initial solution does not need to be basic). From among these procedures, we adopted a well known algorithm, the Two-Phase Method, for use in Aquarius. Below is a brief description of the method. For a complete description see Bazaraa and Jarvis (1977).

During the denominated Phase 1 of the Two-Phase method, the restriction set in (5.6b) is changed by adding a vector of artificial variables $\mathbf{x_a}$, leading to the system of equations $\mathbf{Ax+x_a=r}$, $\mathbf{x \geq 0}$, $\mathbf{x_a \geq 0}$. By construction, **A** contains an identity matrix associated with the artificial vector. This gives an immediate basic feasible solution of the new problem, namely, $\mathbf{x_a=r}$ and $\mathbf{x=0}$. However, even though an initial feasible solution now exists, the problem has in effect been changed. To get back to the original problem, the artificial variables should be forced to zero to attain feasibility in the original problem because $\mathbf{Ax=r}$ if and only if $\mathbf{Ax+x_a=r}$ with $\mathbf{x_a=0}$. The modified LP problem solved during Phase 1 with the basic feasible solution $\mathbf{x=0}$ and $\mathbf{x_a=b}$ is:

Minimize $\quad \mathbf{1\ x_a}$ (5.33a)

Subject to ... $\quad \mathbf{A\ x + x_a = r}$ (5.33b)

$\mathbf{x, x_a \geq 0}$ (5.33c)

where, since the objective function is replaced by an auxiliary function, it is inconsequential that the original objective function (5.6a) is nonlinear. Then, the Simplex Method is used to minimize the sum of the artificial variables over the feasible region for the revised problem. The solution for Phase 1 should have all the artificial variables equal to zero so the solution is also feasible for

the original problem. At the end of Phase 1 we get either a basic feasible solution of the original problem or $\mathbf{x_a=0}$, which implies that the original problem has no feasible solution.

Since there is no objective function driving the Phase 1 solution to any particular state of the system, the constraint set (5.33b) basically defines the IFS. The same set of constraint equations used to restrict the SQP optimization (see Chapter 5, Operational Restrictions) is also used for (5.33b). An obstacle during the computation of the constraint set (5.33b) is the dependance of some of the RHSs on uncontrolled spills and evaporation losses, which are unknown before-hand (as discussed in Chapter 5, Modeling Uncontrolled Releases). To start the computation process, evaporation losses are initialized to some reasonable values, whereas spills are initially assumed equal to zero. In this manner, the model is forced to find an IFS free of spills.

However, when the inflows are characterized by very wet periods, the above restriction may prevent a solution that satisfies the set of constraints. When this occurs, the Phase 1 problem is reformulated assuming positive semi-infinite reservoirs and the problem is resolved. Then, water held artificially in storage (i.e., above the maximum operational storage capacity) is transformed into spills from the reservoir. The last estimates of spills are used to reformulate the Phase 1 problem for a third time, which finally yields the required IFS.

Although this approach is more involved than the simple flow cascading procedure, the efficiency of the LP algorithm and the numerical stability of the Two-Phase Method ensures a much better chance of finding an IFS under all flow conditions.

Time Intervals of Analysis

The analysis performed using Aquarius defines the length and number of time intervals to be considered. For example, although an analysis at the monthly level may be sufficient during the planning of a water resources project, simulating the actual operation of a river system may require weekly or daily time intervals. The length and number of time intervals of an analysis is also affected by the availability of time-dependent data, especially flow data, required to run the model and by execution time.

Modeling water allocation in a basin may require considerable detail for some specific water uses. An example is instream flow to maintain an activity such as whitewater rafting. This water use is highly seasonal, occurring only a few months of the year, and has a nonuniform demand within the recreation season. The demand for whitewater rafting changes within the week because use is greater on the weekends than during weekdays. Detailed consideration of the water supply for this activity in competition with other water uses in the basin may justify modeling system operation at a daily time step. Moreover, if the reservoir system contains enough storage to carry over from one year to the next, then a multi-year operation must be modeled. In this manner, benefits of storing excess water during wet years for future releases during dry years can be realized. Contrarily, if the reservoirs fill practically every year, the period

of analysis can be limited to a single year, known as within-year operation. Each system will present different characteristics and will impose different time step and time horizon requirements for optimization.

Aquarius was conceived to simulate the water allocation in a basin using any time interval of analysis including daily, weekly, and monthly time intervals. Moreover, the model is envisioned to operate under time intervals of uniform as well as nonuniform length. As an example of the latter, consider a one-year optimization horizon subdivided into the first 7 days (short-term operation), the following 3 weeks (medium-term), and the remaining 11 months (long-term). This partition of the within-year operation into intervals of unequal length may coincide with the way inflows to the system are forecasted. Because of limitations in the graphical user interface (GUI) for entering data in the present version of model, only monthly time intervals are accepted. Future versions of the model may include the necessary GUIs to work at shorter time intervals.

Aquarius can be used in a full optimization mode for general planning purposes or in a quasi-simulation mode with restricted foresight capabilities. For the latter, the model distinguishes between the "period of analysis" used to specify the length of the whole segment of time for which the model will simulate the allocation of water in the basin and the "optimization horizon", which is used to specify how far into the future the model should look to build the optimal operational policies.

Optimizing the operation of a complex water system for an extended period of time (e.g., a period equivalent to the system's economic life, as in an analysis performed for planning purposes) is computationally impractical given typical computing capability. Aquarius offers an alternative approach, termed a "quasi-continuous optimization", that allows the user to study the response of a system for a very long time period (e.g., several decades) in a computationally manageable manner. The approach decomposes the full solution of the problem into many consecutive overlapping solutions. Figure 5.7 illustrates this approach for a monthly operation where the overlapping solutions are each three years.

The solid line of figure 5.7 represents a hypothetical optimal reservoir storage trajectory resulting from optimizing the system operation for many years. In this example, identical initial and final storage conditions are imposed, indicated by the horizontal dotted line. The model begins by optimizing over the first three years (i.e., using an optimization horizon equal to 36 months). The ending storage for this three-year solution must meet the final storage constraint, as indicated by the dashed line showing the end of the three-year trajectory. The model disregards the results from the operation for the third year, saving the state of the system for years one and two. When the next set of three years is optimized, the model adopts, as the initial state of the reservoir, the optimal storage found at the end of the second year of operation. The cycle continues for the total number of years with data available. In figure 5.7 it is assumed that the effect of the imposed final storage does not extend farther back than one year; that is, the dashed and full lines only depart from each other during the third year of operation (filled area).

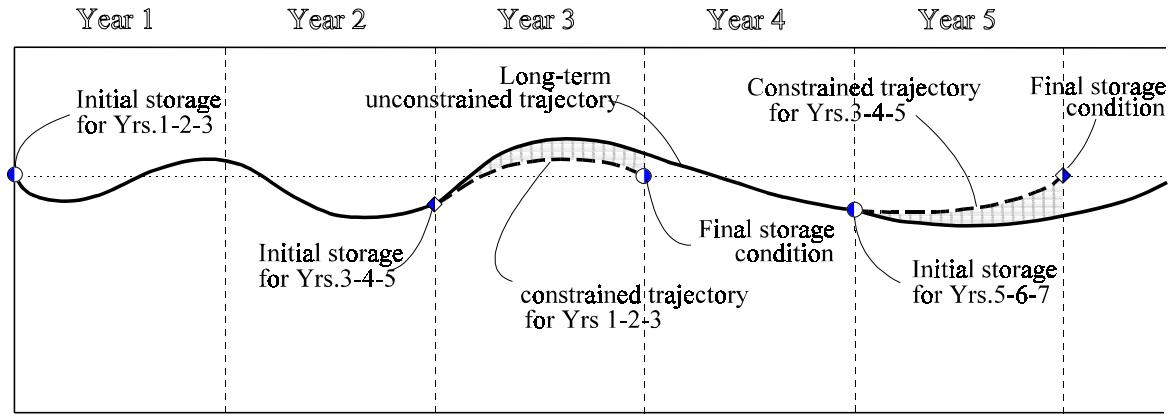


Figure 5.7 Scheme of quasi-continuous optimization.

Repeated testing shows that this is often a reasonable estimate, although the best length for the overlapping period should be determined for each specific system. This procedure of quasi-continuous optimization is believed to circumvent the sometimes undesirable effect of the final boundary conditions on reservoir operation. In addition, running the model with restricted foresight capabilities, for example, with an optimization horizon equal to only one or two time periods, would generate operation policies close to what an operator of a real-world system would produce. The optimization model will behave very much like a simulation model. Running Aquarius under these conditions requires the addition of targets for reservoir storages as part of the data input requirements.