TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Page

LIST OF ALGORITHMS

LIST OF ABREVIATIONS

ETS	École de Technologie Supérieure
ASC	Agence Spatiale Canadienne
ANN	Artificial neural network
DP	Dynamic Programming
DDP	Deterministic Dynamic Programming
DSDP	Dual Stochastic Dynamic Programming
GA	Genetic algorithm
QL	Q-Learning
MC	Monte Carlo methods
MDP	Markovian decision process
NDP	Neuro dynamic programming
SDP	Stochastic Dynamic Programming
SSDP	Sampling Stochastic Dynamic Programming
RL	Reinforcement Learning

LISTE OF SYMBOLS AND UNITS OF MEASUREMENTS

α	Learning rate
γ	Discounted factor
ε	Exploration rate
$\mathbb{I}$	Set of reservoirs, where $i \in \mathbb{I} = \{1, \cdots, I\}$
$\mathbb{J}$	Set of immediate upstream reservoirs, where $j \in \mathbb{J} \subset \mathbb{I}$
$\mathbb{T}$	Set of time step on operation of multireservoir systems, where $t \in \mathbb{T} = \{1, \cdots, T\}$
$s_{i,t}$	Water storage in m^3 for reservoir i in time period t
$x_{i,t}$	Pool elevation in m for reservoir i in time period t
$a_{i,t}$	Water release in m^3 for reservoir i in time period t
$a_{i,t}^{tur}$	Water release from turbine in m^3 for reservoir i in time period t
$sp_{i,t}$	Water spillage in m^3 for reservoir i in time period t
$\omega_{i,t}$	Natural water inflow in m^3 for reservoir i in time period t
$x_{i,t}^{min}$	Minimum pool elevation in m for reservoir i in time period t
$x_{i,t}^{max}$	Maximum pool elevation in m for reservoir i in time period t
$a_{i,t}^{min}$	Minimum water release in m^3 for reservoir i in time period t
$a_{i,t}^{max}$	Maximum water release in m^3 for reservoir i in time period t
$CL_{i,t}$	Violation variable on pool elevation for reservoir i in time period t
$CO_{i,t}$	Violation variable on minimum outflow for reservoir i in time period t
M_{CL_i}	Violation penalty on pool elevation for reservoir i

M_{CO_i}	Violation penalty on minimum outflow for reservoir i
$e_{i,t}$	Energy production by turbine in GWH for reservoir i in time period t
$V(x_{i,t})$	Water value function at the end of horizon
$C_{i,t}(x_{i,t}, \omega_{i,t}, a_{i,t})$	Cost function for reservoir i in time period t
π	Policy, decision making rule

INTRODUCTION

Background

As economy grows and population expands, a great competition has raised towards the perfect use of available water resources. This has brought more attention to the integration of accurate tools for water basin management. One common way to accomplish this goal is to construct a new infrastructure for increasing water supplies. Therefore, reservoirs are created and used to pursue two primary objectives– production of hydroelectric power, and management of natural fluctuations in water inflows. The advantages of controlling the natural fluctuations of water inflows are numerous, including flood control, control of water over drought period, recreational use of water, navigation and several other uses of water resources.

Over the last century, hydroelectric power production has been one of the most economical and efficient resources of renewable energy. In spite of their efficiency, the amount of water resources accessible for hydroelectric power production relies on the limited amount of rainfall or snowmelt. Therefore, management of all these water resources is crucial, so reservoirs as one of the practical means to store water, needs to be operated in a way that the above-mentioned objectives could be accomplished. As an example, by the integration of an appropriate strategy in the operation of reservoir systems, a more flexible and reliable trend of hydropower generation could be resulted.

The operation of a hydropower plant can be formulated into a mathematical concept in particular as an optimization problem. However, solving the problem is a quite challenging task for several reasons. First, the problem contains non-linearities in the model and objectives as well as non-convexity in the function of water head of reservoirs and water discharged by power plants. Secondly, the solution of optimization problem strongly depends on the stochastic random inputs such as water inflows, demands, or market prices. The last, but not the least, the

optimal solution of the optimization problem is in the form of feedback where the water release at each power plant is a function of state variables. This makes the problem difficult to be solved as it involves with high dimensional state-space.

In this study, we would like to solve the operation problem of a large-scale multireservoir system over a short time horizon where the primary objective is to optimize the expected value of hydroelectric power production over a stochastic variable (water inflow). To accomplish this goal, the multireservoir operation problem can be formulated as a stochastic optimization model where the optimal water release policies will be computed over all possible value of water inflow. In the literature, many optimization techniques have been applied to solve the multireservoir operation problem in special dynamic programming and its stochastic version, however, these methods may not be successful in dealing with this large-scale problem as a result of two significant limitations. First, the conditional optimization should be performed on all possible realization of the discretized state vector, so that the computational complexity of the problem grows exponentially with the number of state variables, known as "curse of dimensionality." Meanwhile, an explicit model or information of environment, including transition probabilities and rewards functions, is required, known as "curse of modeling."

Methodology

The main contribution described in this study is concentrated on providing an efficient optimization technique on the multireservoir operation problem. To accomplish this goal, the following major steps are required to take:

M1) At the first step, there is a need to conduct an appropriate literature review over the commonly applied optimization techniques in the multireservoir operation problem. This review can provide a sound information like advantages and disadvantages of optimization

approach, thereby, determining the main research direction of this study. This step will be carried out in the Chapter 1.

M2) According to the literature review, we will make a basic assumption over a suitable optimization technique that can effectively solve the multireservoir operation problem. Regarding this point, the basic step is then to introduce and formalize the general structure of the optimization technique. Chapter 2 provides a complete background and explains the detailed information of selected algorithm.

M3) Possessing a comprehensive and detailed information on selected approach, the main interest is to apply the algorithm to the optimization problem of multireservoir systems. This will also result in some major research questions to improve the performance of the basic algorithm. Chapter 3 provides the detailed information of applying the selected approach on the short-term operation of multireservoir systems.

M4) In the last step, the performance of designed approach will be verified considering a multireservoir case study. First, a description of the case study will be presented as well as building the corresponding simulator. Afterward, the designed approach will be implemented on the particular multireservoir system and several experiments will be carried out to calibrate the algorithm's input. Chapter 4 covers the presentation of all simulation results as well as their analysis.

Objectives

As we mentioned before, DP stands out among the optimization techniques in solving the reservoir operation problems specially the problem with a few number of reservoirs (one or two). However, increasing the dimension of the problem, DP may not be successful in handling the optimization problem in particular the stochastic version. In this study, our hypothesis is that a reinforcement learning algorithm could possess better performance compared with

DP algorithm in dealing with the stochastic optimization problem of multireservoir systems. Towards this assumption, we will concentrate on the possible improvements that can assist the quality of the reinforcement learning algorithm. As a result, the following research questions are considered in this study:

Q1) Can reinforcement learning algorithm be applied in the operation of mulitreservoir systems?

Q2) What are the best algorithms' parameters in order to deliver the best performance?

Q3) As the reinforcement learning tries to learn the environment through trial and error mechanism (a simulator), what could be the best training dataset?

Q4) What is the impact of integrating different forms of the cost function in the operation of multireservoir system within the reinforcement learning algorithm?

Q5) Can reinforcement learning algorithm be integrated with function approximation techniques to improve its performance?

In this study, we are trying to find appropriate answers for above-mentioned questions so that the solution provided by the designed approach could be reliable and robust enough in terms of any uncertainties exist in our model.

Organization

In this Chapter, motivations and objectives of this thesis are presented. In Chapter 1, a review will be provided on the optimization techniques that are commonly applied to the operation of multireservoir systems. Chapter 2 provides the basic theory and background of reinforcement learning in particular Q-learning approach. Afterward, the detailed information of Q-learning approach applied to the short-term management of multireservoir systems is represented in Chapter 3. In Chapter 4, a complete simulation analysis is provided to show the performance

of the designed Q-learning approach. Finally, conclusions are presented in the last part of this thesis.

CHAPTER 1

LITERATURE REVIEW

Over recent decades, computers model have been utilized in order to evaluate most feature of hydrology, river, and reservoir systems operation. Further to this point, these models have been incorporated with optimization methodologies with the aim of developing the management of water basin by optimally operating the reservoirs and other water recourses. As a result, the simulation and optimization models simultaneously have been receiving a great interest on the side of the operational use to help water operators in solving water control problems over a time horizon (Labadie, 2004a).

This section provides a review on approaches for the management of the reservoir systems. The methods are divided into two classes. The first one collects traditional algorithms and approaches based on dynamic programming. For the purpose of this review, conventional methods are the ones that have been used in operation of water system over several years. Within the first class, the optimization techniques themselves are also classified as:

- Deterministic dynamic programming,
- Stochastic dynamic programming,
- Stochastic dual dynamic programming,
- Sampling stochastic dynamic programming,
- Dynamic programming with function approximation.

The second class collects some state-of-the-arts on the side of artificial intelligence methods that are applied to water system management. In this way, theories behind these algorithms are heavily borrowed from advances in computer science in particular reinforcement learning in order to provide solid performances. In spite of the efficiency presented by these algorithms,

they have not been commonly used or have just been tested in active recent researches in the field of water resource optimization. In these algorithms, the focus is mostly provided in the concept of learning an optimal policy based on experiences obtained by interacting with an environment. Usually, these approaches have been inspired by living organisms mechanism with the purpose of evolving or learning as well as allowing a computer or machine to get the features of the desired problem initially by concentrating on the statistical information of the inputs.

Moreover, a brief description will be provided on the other commonly used techniques in operation of reservoir system where they lose their efficiency in dealing with multireservoir operation problem. In the following, the above-mentioned categories will be presented by considering the corresponding advantage and the disadvantage of each methodology in hydropower operation problem.

1.1 Dynamic Programming (DP)

1.1.1 Background

The objective of operating reservoir water systems is to optimize the use of water for different applications like water supply, hydropower generation, water quality, etc. The optimization problem is quite a difficult task as decisions correspond to the water release policy depend on an uncertain systems' variable as random water inflow. Dynamic programming is one of the commonly used optimization techniques for solving reservoir optimization problem. Dynamic programming technique introduced by the early work of Bellman (1954), relies on the simple idea known as the principle of optimality (Bellman equation). In DP framework with an application of a multireservoir system, the corresponding "cost-to-go" function and the solution obtained as a release policy are computed as a function of the state variables for all time intervals. The rule here is to find the control decision (water release) so as to maximize the current profit and the expected award from the future operation by the use of the cost-to-go function. Using the Bellman equation, the optimal cost-to-go function $V_t^{\star}(s_t)$ can be calculated as:

$$V_t^{\star}(s_t) = \max_{\pi} \mathop{\mathbb{E}}_{\omega_t} \{ r_t(s_t, a_t) + V_{t+1}(s_{t+1}) \} \tag{1.1}$$

In this approach, the original problem decomposes into several subproblems that should be solved recursively over each time step t. Here the expectation is taken with respect to the random variable ω_t. The policy π is a function that maps the control decision a_t to the state variable s_t. In DP applied to reservoir operation problem, the set of state variables are usually defined as reservoir storage and hydrologic information. Meanwhile, they need to be specified in a finite discretized set. Additionally, the set of control actions (water release) that transform the system from current state to the next one should also be determined in a finite discretized set (Nandalal & Bogardi, 2013).

The above discretization results in some limitations in the integration of DP algorithm to the large-scale multireservoir optimization problem. The main drawback is the exponential growth in computational burden with respect to the number of a discretized variable, that was also introduced by bellman as ”curse of dimensionality”. For more clarification, assuming a multireservoir system with m discretization levels of water for n number of reservoirs could yield in the m^n number of solutions just for the combination of the state variable. In addition, solving the above optimization problem requires a prior accurate knowledge on stochastic variable (water inflow) ω_t. In multireservoir operation problems, having such information in advance like transition probability matrix is quite challenging, known as ”curse of modeling”. In the following sections, variant versions of DP algorithm will be introduced along to their relation with the basic DP approach.

Dynamic programming can be solved by the use of two common approaches as value iteration and policy iteration. In the value iteration approaches, the algorithm starts from the last stage and then works backward, refining an estimate of all cost-to- go functions. The value of the cost-to-go functions are first assigned to an arbitrary value and using the above-mentioned equations, the values are iteratively updated until the algorithm reaches a convergence over the corresponding value of cost-to- go functions. In policy iteration algorithm, it starts with

a policy and iteratively improves it. In spite of value iteration algorithm, policy iteration terminates in a finite number of iteration. In this algorithm, we start with an arbitrary policy π_0 (an approximation to the optimal policy works best) and carries out two main steps as policy evaluation and policy improvement. Once the policies remain unchanged in two consecutive steps, the algorithm stops and returns the optimal value of cost-to-go functions.

1.1.2 Deterministic Dynamic Programming

One of the simplest forms of dynamic programming is the deterministic version where the Bellman equation (equation (1.1)) is being solved by considering a fixed value for the random variable ω_t (water inflow). The study performed by George (1967) was one of the early applications of the deterministic dynamic programming in which a finite horizon optimization model has been applied to a single reservoir system. The extensive use of DP can be the consequence of several facts as the ease of handling non-linearities and non-convexity of objective or constraint functions, inclusion of stochasticity, and the computation of feedback or closed-loop policies of reservoir operation.

Although the deterministic dynamic programming eliminates the expectation part of bellman equation, it still suffers from the dimensionality problem. To this end, a variety of improvements has been considered to handle the curse of dimensionality in applying DP algorithm to reservoir operation problem. Larson (1965) introduced the incremental dynamic programming where the basic difference is in the determination of the time step over which a given control decision is applied. In the traditional DP, the time step is a fixed value over the whole process. However, in the new technique, the time interval is considered as the smallest amount of time required for state variable to be changed one unit. One of the main consequences of this approach is that for any control decision taken at a given state, the next state will be located in a neighborhood of the given state. In another case, Differential Dynamic Programming approach has been introduced by Jacobson & Mayne (1970) where it employs the analytical solution instead of discretizing the state space. This approach was further developed by Murray & Yakowitz (1979) to a constrained problem.

1.1.3 Stochastic Dynamic Programming

Stochastic dynamic programming (SDP) is one of the most common and useful versions of DP algorithm where it considers the stochastic nature of incoming inflows in the operation of multireservoir system as in Côté & Leconte (2016). Alike to the standard DP, a key point, here, is the conditional optimization performed on all possible realization of the discretized state vector, consequently, results in the importance of already mentioned drawbacks as a curse of dimensionality and modeling. Lamond (2003) adapted SDP algorithm for solving the optimization problem of a single reservoir using a piecewise polynomial approximation of the future value functions.

In SDP model, the incoming water inflows are typically described by the periodic Markovian process. In addition, the selection of the criteria beyond the choice of inflow relies on the system's properties like the availability of measured information in term of the time interval for a decision maker. Furthermore, the computational aspect should be taken into account in term of how hydrologic information should be formulated in SDP. Huang *et al.* (1991b) introduced several types of inflow representation as state variables in SDP on the Feitsui reservoir complex in Taiwan. The author concluded that using the inflow information in the previous time period results in a much more superior performance in comparison with the use of current time step information.

Furthermore, the value of hydrological information within the SDP optimization model has been studied by Tejada-Guibert *et al.* (1995) in the multireservoir problem where they utilized several hydrological state variables in the Shasta-Trinity reservoir system in California. Afterward, the computed policies based on SDP were compared and analyzed by the use of simulation model. In this research, the authors studied the impact of different types of models with some possible forms of inflow state variable. They concluded that the value of employing an accurate inflow prediction relies on different system properties such as the magnitude of incoming water, power demands and the intensity of the penalties on shortages. In later years, parametric techniques to introduce the water inflow has been proposed by Turgeon (2005) and

Saad *et al.* (1992) with the use of a linear auto-regression (AR) model, in which the generated model used in solving the relative SDP optimization problem within the application of reservoir operation. In this research, it was noted that using multi-lag autocorrelation of inflows instead of the traditional lag-1 models, can possess many benefits. However, in order to eliminate the increase in computational time requirement as a result of the increase in state space, the multistage autocorrelation of inflows was presented by the conditional average of daily inflow.

Summing up, the use of SDP in optimizing the multireservoir operating system is often conducted by proposing an assumption that the natural water inflows are not cross-correlated. Considering such an assumption is resulted in obtaining a rough approximation in the term of optimal operation policy. To overcome this issue, Yeh (1985) recommended to divide the SDP optimization method and the water stream inflow generation, or considering an aggregation or decomposition methods that are introduced by Turgeon (1980).

1.1.4 Stochastic Dual Dynamic Programming

Another technique has been studied with an application of reservoir optimization problem is stochastic dual dynamic programming (SDDP) in which it uses stochastic programming technique in solving the equation (1.1). The SDDP method possesses an interesting feature by trying to eliminate the curse of dimensionality exists in DP based techniques. In this approach, a DP algorithm is reformulated as a two-stage decision process where the first stage associates for the immediate cost and the second stage corresponds to the future cost. The dimensionality problem was avoided by approximating the cost-to-go function obtained from the dual solution of the of the second-stage problem at each time step. The reason behind this elimination is the fact that it does not require the state space discretization, therefore, attracting much more conservative practical strategies.

The benefit provided by the continuous state space representation is replaced by a combinatorial explosion of water inflow scenario along with the number of stages in multistage optimization problem. However, this advantage may result in more intensive optimization problem if

the inflows to the system possess a serial correlation over the time period. This is in a great contradiction with SDP techniques, in which they are computationally affordable by growing almost linearly with respect to the number of time steps. So, SDDP becomes inadequate in problems with complex stochastic structure and it becomes powerless in computing the optimal closed-loop operating policies, in particular, the long-term planning problems. Furthermore, the SDDP employs the linear programming subproblems instead of original counterparts in order to find Bender's cuts, so that it can improve a piecewise linear approximations of the future value function. This might be risky in a term of finding the accurate solution by applying the method to highly non-convex problems in particular hydro-power system optimization.

One of the primary work in the field of SDDP has been done by (Pereira & Pinto, 1991) in which they tried to optimize the operation problem of the multireservoir system. In this study, the SDDP algorithm was implemented to a system of the reservoir including 39 hydroelectric plants in the southern-southeastern Brazilian power pool.

1.1.5 Sampling Stochastic Dynamic Programming

The sampling stochastic dynamic programming (SSDP) is a variation of SDP approach that uses scenarios to represent the stochastic random variable ω_t. The field of SSDP with an application of hydro-power optimization problem has been always faced with an argument behind the possibility of overcoming the representation of the random variables (inflow, demand, prices). This problem even becomes much harder when the single reservoir is replaced by multireservoir system, results in dealing with multivariate random vectors, in particular the correlated random variables. On the other hand, representation of data trajectories or scenario would become the most challenging and time-consuming part of computing the solution of stochastic optimization problems. The goal here would then be to generate trajectories or scenarios so that the best approximation of the distribution of the random variables could be provided within a computationally manageable way in the optimization model Séguin *et al.* (2017).

Kelman *et al.* (1990) introduced SSDP technique that synthesizes the set of inflow scenarios in a straightforward way in the DP in order to represent the different properties of streamflows over all reservoir within a basin. In the SSDP methodology, the main point is that it does not ask for classifying random variables or even their probability distributions in the form of discrete states. Faber & Stedinger (2001) incorporated the SSDP with a concept of feedback (Frequently update) on the streamflow prediction provided by the National Weather Service. As a consequence, the SSDP has become a promising technique in hydro-power optimization model due to its advantage of combining the SDP with the intrinsic scenario properties. After this time, it was mentioned by Côté & Leconte (2016) that the use of SSDP algorithm is an interesting technique to introduce the distributions and temporal cross-correlation of inflows. Recently, different sampling-based approaches have been introduced to overcome the problem of generating scenarios and a brief overview on some of these techniques are provided in the following.

The conditional sampling technique is one of most popular and common method applied in the generation of scenario samples due to its simple structural representation. In this algorithm, an approximation of probability function by empirical component produced by random samples. As a result of computational limitation, these scenarios cannot be represented by a very large value, therefore the empirical observations cannot be a good approximation of the original ones. Pennanen & Koivu (2002) illustrated that modern integration quadratures represent an interesting and elementary solution to random sampling. In that case, these quadratures are designed to provide a good estimation of given probability features by a few quadrature points. Furthermore, Loretan (1997) successfully implemented principal component analysis to compensate the curse of dimensionality in scenario tree. Sampling from principal components allows correlated random vectors to be obtained.

Scenario reduction is another technique used by SSDP in order to incorporate a much smaller number of scenarios while they possess more correlation. In this approach, a subset of scenarios form the whole cardinality is determined, therefore, a probability information based on this set is identified. Based on this point, all deleted scenarios have zero probability.

Heitsch & Römisch (2003) introduced two new approaches for obtaining reduced probability measures in an optimal manner. The advantage by the use of these techniques is the reduction over its generality. On the other hand, no requirements are proposed on the stochastic data processes i. e. the coloration or dependency over the structure of scenarios, the dimension of the stochastic process, etc.

In the later years, Tarim *et al.* (2006) addressed the combinatorial decision problems that are involved with uncertainty and probability by the use of scenario reduction approach. In this research, they considered a semantics for stochastic constraint programs depend on scenario trees. By the use of this framework, they could convert stochastic constraint models into conventional (non-stochastic) constraint model that is resulted in exploiting the entire power of current constraint solvers. Over the last year, a number of studies have been done on these line of study as in Larsen *et al.* (2015). In this study, they evaluated and compared three scenario reduction approaches used to build a multistage scenario tree in the SSDP.

1.1.6 Dynamic programming with function approximation

DP based optimization techniques are commonly adapted by continuous approximation of the discretized "cost-to-go" function. In these approaches, the cost-to-go function is just computed at certain discretized state values, however, the value of the function over the other points can be roughly estimated by using the interpolation of nearby grid points. In the literature, a great interest has been shown in the concept of decreasing the possible computational time requirement by the use of multivariate polynomials or piecewise polynomial functions in the SDP. Tejada-Guibert *et al.* (1993) concluded that computational reduction is firstly as a result of the accuracy of higher order functions which leads to an appropriate solution even if there is a coarse state space discretization. Secondly, efficient gradient-based optimization algorithms can be used to compute better approximations to the optimal solutions.

Johnson *et al.* (1993) employed an accurate approximation of cost-to-go function by the use of high order spline functions in a way that a rough discretization of state variables could be

handled. The spline function composes of some multivariate cubic polynomials where each function is defined over a part of the state space domain. The coefficients used in spline function were specified based on the fact that all desirable grid point should be attainable by the use of interpolated function. Another key point is that this technique has a good computational time consumption even for a system with two or more reservoirs. In another case, Tejada-Guibert *et al.* (1995) employed the above mentioned piecewise cubic functions to estimate the cost-to-go function for the set of hydro-power plant system in Northern California.

In the study performed by Lamond (2003), a piecewise polynomial estimation of the future value function with an application of a single reservoir operation problem has been investigated. The conclusion made by the author was that the newly designed approach is much faster than both discrete and continuous form of DP algorithm using splines on a fixed grid. Furthermore, they recommended that the spline estimation is not well adapted when there is a piecewise linear reward function.

1.2 Reinforcement learning (RL)

Reinforcement learning (RL) is a machine learning algorithm in which it originally inspired by behaviorist psychology. This inspiration comes from the fact that the agent or machine tries to automatically make suitable decisions in response to specific stimuli based on the corresponding feedback as rewards or punishments. The feedback is called reinforcement signal where the agent discovers the specific link between a specific action and a reward (or punishment) and then select a preferred action within a specific context. The whole process is known as reinforcement learning. The main difference between RL and other machine learning algorithms is that RL is not exactly supervised learning since it does not directly depend on set of ”supervised” (or labeled) data (the training set). In fact, it relies on being able to observe the response of the taken actions and evaluate the term of ”reward”. However, it is also not an unsupervised learning either, because we know in advance when we model our ”learner” which is the expected reward.

In the field of operation research and control theory, RL is also known as approximate dynamic programming or neuro-dynamic programming that was introduced by the early work of Bertsekas & Tsitsiklis (1995). Though in most studies, the problem has been considered with the existence of optimal solutions and their characterization. In all these research areas, RL introduces a framework for computational learning in which the agent learns how to behave within a trial-and-error mechanism in an environment so that it can maximize its rewards over time (Kaelbling *et al.*, 1996). In the most challenging and interesting cases, actions may not only affect on the immediate award from the environment, but it can affect the later situation and, through that all future rewards (Sutton & Barto, 1998).

So far, RL has been applied in various disciplines namely: machine learning, psychology, operations research, control theory. Being quite a successful tool in solving different problems with different applications, it became more popular due to its simple algorithmic and mathematical foundations. RL shares some common with DP based algorithms, in particular, the use of Bellman equation to obtain the sequence of an optimal decision to optimize the reward function. However, it is able to avoid the limitations offered by DP algorithm in dealing with large-scale problems. To this end, RL provides the following two key benefits in tackling large-scale problems:

1) Curse of dimensionality: RL uses forward recursion instead of backward recursion used by DP and SDP to solve the Bellman equation. This will avoid the demand for solving the problem in all possible realization of system's state variable, therefore, the computational time required for solving the problem is being reduced as a result of less number of stages. Furthermore, there could be a possibility in the use of function approximation methods in order to decrease more the curse of dimensionality.

2) Curse of modeling: RL does not require to know either the exact transition functions or the transition probability matrix of the system. This will result in the capability of solving the MDP without knowing the exact model of the system.

In RL, the environment is commonly explained with a concept of MDP, as this context is also utilized by DP's. One of the specific distinction between the conventional techniques and RL algorithms is that there is no need to have prior information about the MDP and usually they target large MDPs where exact approaches become infeasible. Meanwhile, RL never requires input/output pairs, nor explicit sub-optimal actions. However, there is a special attention to its online performance that is summarized in obtaining a balance between exploration and exploitation. The concept of exploitation vs. exploration in RL has attracted much attention through the problems classified as finite MDPs in particular multi-armed bandit problems.

Recently, RL approaches have received a popularity in the application of water reservoir systems. Castelletti (2002) presented a new methodology known as "Q-learning planning" or "QLP", and employed the newly designed algorithm to the water resource operation problem. The approach combined a Q-learning strategy with conventional SDP techniques to generate a hybrid system for reservoir planning problem. Lee & Labadie (2007) proposed a solution for multireservoir operation problem based on the Q-learning approach. In this study, they studied the impact of operating rules with/without conditioning on predicted hydrologic state variable. For the unconditional case, optimized operation policies obtained with different discount factors show similar patterns. The discount factor is the basic parameters in RL algorithm where the estimate of future value can possess less value. For the conditional cases, two approaches as K-means clustering and percentile value approaches are compared for determining hydrologic states of the model. Although both approaches were integrated within the Q-learning algorithm, K-means clustering approach shows a better performance compared with the other one. Furthermore, an interpolation method was adapted to approximate the optimal value of future reward on the Q-learning algorithm. It was figure out that nearest neighbor methods possess more appropriate and stable convergence of the algorithm. The performance of the Q-learning algorithm was compared with SDP, or SSDP approaches in these studies.

Castelletti *et al.* (2010) employed reinforcement learning in particular fitted Q-learning approach in determining the daily water release policy in a multireservoir system over an annual time period. In this study, they combined the concept of continuous estimation of the value

functions by learning from experience in order to find the optimal daily water release where there is cyclo-stationary behavior in the operating policies (yearly operation). Furthermore, the continuous approximation technique used in Q-learning approach based on tree-based regression algorithm. One of the promising property offered by tree-based regression approach is the possibility of reducing the curse of dimensionality by applying a very rough discretization scale. The introduced fitted Q-iteration algorithm finds a stationary policy, e.g., just one operating rule where it can be represented in a general form of $u_t = m(x_t)$ on the side of operating rule for a stationary system. While the main system is not stationary but periodic over a time period (year). A simple approach has been considered by an extension of basic framework to the non-stationary version, by including the time as a part of the state vector. Finally, the learning experience, in the form of a dataset generated from historical observations, provides the opportunity to overcome the curse of modeling.

In the recent application of reservoir optimization problem, RL techniques have integrated with another newly proposed approach with an attempt to generate a more efficient solution. Mariano-Romero *et al.* (2007) improved hydrologic optimization problem by the use of multi-objective distributed Q-learning, in which it basically takes an advantage of multi-agent approach in Q-learning. Abolpour *et al.* (2007) coupled RL with the fuzzy logic principle in order to improve river basin allocation problem. In another work, Tizhoosh & Ventresca (2008) expanded optimal mid-term policies for a hydropower plant by applying Q-learning, therefore, moved on combining Q-learning with opposition-based learning in the management of single reservoir system with monthly time step over a year.

So far, various applications of RL techniques to either a single reservoir or multireservoir operation problems have been studied with a purpose of hydro-power optimization problem where the operating rules are also verified and tested over whole possible scenarios to reveal their efficiency and reliability. This indicates that the agent has the adequate level of ability to estimate the value of its actions from simulated experience, thereby improving an optimal or near-optimal control policy. In the literature, no studies were found that apply the RL approach to improve short-term planning of hydro-power generation problem of a multireservoir systems

as well as dealing with cross-correlated objectives and constraints. strategy then heavily depends on the learning from the observed experience due to the neural structure of the brain that is used here. Recent studies in the field water basin management have shown the superiority of ANN approach is applied to the

1.2.1 Artificial Neural Networks

An artificial neural network (ANN) is a network which possesses a similar functionality compared with the central nervous system and usually used to estimate or approximate functions based on a large number of known/unknown inputs. The introduced strategy heavily depends on the learning process from observed experience due to the neural structure of brain that is used here. Recently, ANN and its new designed version called deep ANN has been provided a great superiority within the other classes of approximators in several domains including the water basin management problem as in Bhattacharya *et al.* (2003).

ANN has been adapted within SDP framework to approximate the ”cost-to-go” function with fewer sampling points as in Fayaed *et al.* (2013). Saad *et al.* (1996) executed ANN to the long-term stochastic operation planning problem of the multireservoir system of Quebec’s La Grande River. The ANN was trained to disaggregate the stored level of water on each reservoir to an aggregated levels where the training dataset is determined by solving the deterministic optimization problem with a large number of equally likely flow sequences. They concluded that the use of ANN is much more efficient than the principal components approach. In another study, Kim *et al.* (2009) employed ANN for prediction of inflow to a reservoir in an hourly time step by employing the current value of weather prediction information, historical rainfall data.

ANN has also received a considerable reputation in the particular area of water quality operation. Wen & Lee (1998) utilized ANN in multi-objective optimization in order to optimize the operation of water quality towards a water resource management. Chaves & Kojiri (2007) combined neural network with fuzzy SDP algorithm and genetic dynamic programming with

a purpose of water quality. In another work, ANN methodology was also employed to the modeling of water quality as in (Suen & Eheart, 2003).

Recently, Labadie presented an overview of a various application of neural networks applied to the optimization of reservoir operation system in his researches e.g. Labadie (2004b). Rani & Moreira (2010) also provided an overview on the use of neural networks and genetic algorithms with an application of hydro-power optimization problem. Further to these studies, some specific efforts were also made in the literature over the use of ANN with an application of reservoir operation and hydraulic network system problem (Chaves & Kojiri, 2007).

1.3 Other Optimization Techniques

In the recent years, a great verity of mathematical models has been executed in the optimization of hydropower reservoir systems from both operation and management aspects. In this way, the idea beyond the choice of mathematical modeling broadly relies on the characteristics of each application. In the following, we will shortly review and explain several optimization methods that are commonly utilized to solve the reservoir system optimization problem, with a further concentration on the approaches, applied in multireservoir systems. Towards this fact, Labadie (1998) represented a comprehensive review of the optimization approaches used in reservoir operation and management.

1.3.1 Linear Programming

One of the early optimization approaches applied to the reservoir operation problem is linear programming (LP). In this case, employing LP requires all the constraints and objective function whether to be linear or linearized by applying one of the available linearization techniques, such as Taylor series expansion or piecewise linearization. One of the main priority over the LP can be pointed out in its convergence to the global optimal solution. Moreover, in multireservoir optimization problems where the number of constraints and variables are getting large, decomposition methods such as Dantzig-Wolf or Bender decomposition technique

can be considered as an alternative solution to speed up the process Yeh (1985). In addition, problem formulation within the structure of LP is quite easy and could be readily solved by employing commercially available LP solvers Heydari *et al.* (2015).

Turgeon (1987) exploited mixed integer LP formulation in a monthly time step for site selection problem of hydro-power plants. In addition, LP technique also applied by Hiew *et al.* (1989) to a set of reservoir complex contains eight sites in northern Colorado. In another study, an optimization based on short-term LP model has been presented by Shawwash *et al.* (2000) where the model has been developed to compute the optimal short-term management policies that satisfy the hourly demand and maximize the outcome for BC hydro water resources. Using the improved model as a short-term management of hydro-power plant has resulted in 0.25-1.0% improvement in total electricity production as well as the value of extra water storage. The key point in using LP in reservoir optimization model is the possibility of having sensitivity analysis that can be derived with the use of simplex dual variables. This information can be used in planning schedule and in real time operation of the system Martin (Piekutowski *et al.*, 1994).

1.3.2 Multiobjective Optimization

In the multi-objective optimization problems applied to the reservoir optimization model, two approaches were introduced by Labadie (1998). The first approach is to consider one objective in the objective function and treating others as constraints while in the second topology, different weights were considered to each goal. In 1984, Can & Houck (1984) provided an optimization model in a four reservoir system with the use of preemptive goal programming (PGP) technique. In this comprehensive study, it was concluded that PGP makes the policy constraints to be more flexible and it proposed a great superiority on the large-scale problem in comparison with other LP optimal operating model. The principle idea of PGP is to assign an expectation level for each goal and then prioritize them. Therefore, the goal will be accomplished sequentially. A remarkable advantage of PGP is that there is no need to any penalty-benefit function, however, since the goal programming depends on obtaining a

predetermined target value, the global optimal value of the objective may not be discovered Loganathan & Bhattacharya (1990).

1.3.3 Nonlinear Programming

In spite of the popularity attracted in the use of LP towards the optimization of a reservoir system, non-linear programming (NPL) has been obtained less attention. The main fact behind this trend is the result of possessing slow optimization process, consequently, can provide a non-optimal or an inferior solution. However, in a case of non-linear or non-linearizable problems in which the non-linearity may come from either non-linear objective function or constraint, NLP is the only candidate to solve hydropower generation problem. It was mentioned by Labadie (1998) that most reliable and robust NLP techniques used to solve reservoir optimization models are Sequential Linear Programming, Sequential Quadratic Programming, and the Generalized Reduced Gradient Method.

Over the last decades, some studies have been done on hydropower reservoir operations with an application of NLP. In Barros *et al.* (2003), NLP formulation was applied to a large set of a multireservoir system in Brazil. In this case, the multi- objective optimization model was handled by considering LP with Taylor series expansion. A point conducted by this study is that the NLP can be considered as the most accurate and reliable solution for real-time operation than the LP model.

1.3.4 Genetic Algorithm

The genetic algorithm (GA) is one of the commonly used and powerful evolutionary algorithm applied based on the principle of Darwinian natural selection and survival of the fittest. GA performs optimization within a process in which it uses ”the mechanics of natural selection and natural genetics” Goldberg (1989). In this method, the binary or decimal numbers are represented as a point or individual candidate solutions in the search space, known as strings or chromosomes. Then, these individuals go through some genetic operations e.g. selection,

crossover, and mutation operations. Each chromosome is then evaluated and weighted with a fitness function. Based on their ranking, individuals (parents) are chosen for the creation of the next generation by performing three fundamental operations, viz., reproduction, crossover, and mutation. The result is then a new population (offspring) with basically better fitness value. The process continues till the system converges to the desired accuracy after specified numbers of generations (Chandrasekaran *et al.*, 2010). However, there is no guarantee in finding the global optimal solution since the convexity of the objective function cannot be proven.

One of the earliest work successfully accomplished by applying the GA to a four-reservoir system has been done by Wardlaw & Sharif (1999). They concluded that GA presents robust and accurate optimal solutions and could be successfully applied to the real-time operation of hydropower planning problem by considering stochastically generated inflows. In the latter study, Huang *et al.* (1991a) combined GA with SDP in order to handle the dimensionality of long-term planning problem in multireservoir operation system in northern Taiwan. They concluded that even though having GA-based SDP may provide more time consumption as it proceeds from generation to other generation, the model could overcome the curse of dimensionality in searching solutions. Recently, Reis *et al.* (2005) and Hınçal *et al.* (2011) introduced a hybrid genetic algorithm and linear programming approach for reservoir optimization problem. In these works, they handled the randomness behavior of future inflows by using a three stage tree-based generated inflows. They evaluated their approach on a set of four hydrothermal reservoir system and the comparison made on the SDDP model. It was concluded that the newly above-mentioned approach offers some computational benefits despite its computationally expensive behavior.

1.3.5 Fuzzy Programming

Over the last decade, a great interest has been shown in the field of fuzzy set theory and fuzzy logic with certain attention on the hyrdo-power optimization problem as in Deka & Chandramouli (2009). Mousavi *et al.* (2005) used a deterministic dynamic programing to compute the optimal set of water inflows, and reservoir release as long as applying the fuzzy rule–based

model to establish the general operating policies in the second stage. Fuzzy theory is generally introduced based on the imprecision and vagueness in which the variables are numerically can be described by a membership number between 0 and 1.0.

Russell & Campbell (1996) employed the fuzzy logic theory in computing the operating rules for a hydro-power plant where the inflows and energy price could be uncertain. In later works, a fuzzy SDP technique with fuzzy objectives was presented as well as fuzzy cross coloration between current and future water release decisions. Furthermore, Mousavi *et al.* (2004) developed another method called fuzzy-state SDP on the real-time operation of hyrdo-power optimization problem by considering the uncertainties over the hydrologic parameters and the imprecision as a result of variable discretization. To this end, the transition probability matrix can be estimated by the definition of fuzzy Markov chains.

1.4 Hydrological simulation model

Keeping aside the optimization model on the operation of multireservoir complex, there is a need to create a simulation model to evaluate the system dynamics on each time step. In this way, the model used as a transition function of a hydropower plant is a principle part of their operations. This is then much more valuable since the model of the reservoir is highly nonlinear and in some cases, it requires to model some natural elements in the environment. As an example, it highly depends on hydrology and hydromechanics as for the description of the water movement as well as its distribution. Due to this fact, the design of simulator for hydropower plant presents a major challenge. Considering the above mentioned difficulties, some developments have been provided in hydrologic simulation softwares.

Recently, several hydrological softwares has been designed in oder to simulate and compute the physical characteristics of hydropower system. This softwares can then be incorporated with some optimization package in order to obtain an optimal operation of the desired system of a reservoir. Riverware is one of popular hydrological software that is commonly used in reservoir operation system.

Generally speaking, the simulation models of reservoir complex can be considered as a practical tool for identifying and evaluating the effect of various alternative system operations. However, these models are unlikely to be helpful unless they possess the flexibility of simulating the original system. Considering all these possibilities, any further assumptions provided for each of these sources of uncertainty can result in an extensive impact on the planning and operation of the system. These points have served to stimulate the improvement of hydrological models, models that take into account the possibility of providing any natural elements and evaluate their impact on the design and operation on the system.

1.5 Conclusions

The literature review carried out shows that this area of research is still very active and that different optimization approaches and modeling techniques are being tried for dealing with the reservoir systems optimization problem. The review shows that employing an explicit stochastic optimization approach would be the most advantageous since it provides the best representation of this complex problem. The main obstacle that needs to be addressed and resolved, however, is the high dimensionality of the problem.

From this literature review, it can be concluded that DP algorithms remain a very powerful technique for handling the nonlinear, stochastic reservoir optimization problem even if they possesses some difficulties in dealing with large-scale problems. Among the numerous efforts attempted to alleviate the curse of dimensionality, which is aggravating the large-scale SDP method, function approximation techniques and/or sampling techniques resulted in some successful applications in the multireservoir hydropower generation operations planning problem. However, the newly addressed theory known as reinforcement learning provides a great flexibility in dealing with large-scale Markovian decision problems. To this end, the following Chapter presents the main concepts and computational aspects of RL techniques.

CHAPTER 2

THE Q-LEARNING (QL) APPROACH

2.1 Introduction

Q-learning (QL) algorithm is a version of model-free reinforcement learning. It can also be considered as a variant form of DP that is less computationally expensive. The main feature of this approach is the ability to learn how to behave optimally in an uncertain environment. Using the learning mechanism, an agent (decision maker) is able to compute an optimal action-selection policy. A policy, here, is a function that maps the current state of the agent to the specific action (decision). Once the learning process is completed, the agent has an estimation of action-value function, also known as Q-factor that is the value of taking a specific action by being in a particular state. Considering these estimated values, the optimal policy can then be created by simply selecting the action with the best value in each state.

The QL algorithm can be viewed in two main steps. First, the training process of QL algorithm needs to be done where the purpose is to compute the Q-factors, thereby, obtaining the optimal policies. Once such functions and policies are constructed, they will be used to solve the main problem which can be addressed as the verification of trained Q-factors. This part is performed in the second step of implementing the QL approach. This Chapter just covers the training step of QL approach by studying the corresponding principles and theories.

2.2 QL basic concept

The basic components of QL algorithm can be summarized by an agent (decision maker) and an environment and a communication signal between them. The interaction between the agent and the corresponding environment can be described within the framework of Markov Decision Processes (MDP). The agent and the environment communicate by a sequence of decisions that are taken on each time steps over a time horizon T. At each time step t, the decision maker receives some information about the current state of a system $s_t \in S$ where S is the discretized

set of state. Given the information of s_t, it chooses an action a_t from the discretized set of possible action $\mathbb{A}$. By applying the action, the agent will receive a feedback from environment as an indication of a reward $r_t(s_t, a_t)$ for corresponding pair of the state and the action. In addition, the new state of the system shown by s_{t+1}, will be revealed. Within this process, the agent will try to take sequence of actions so as to maximize its accumulated rewards over the whole time horizon. Figure 2.1 shows the interaction of agent and environment through the QL algorithm.

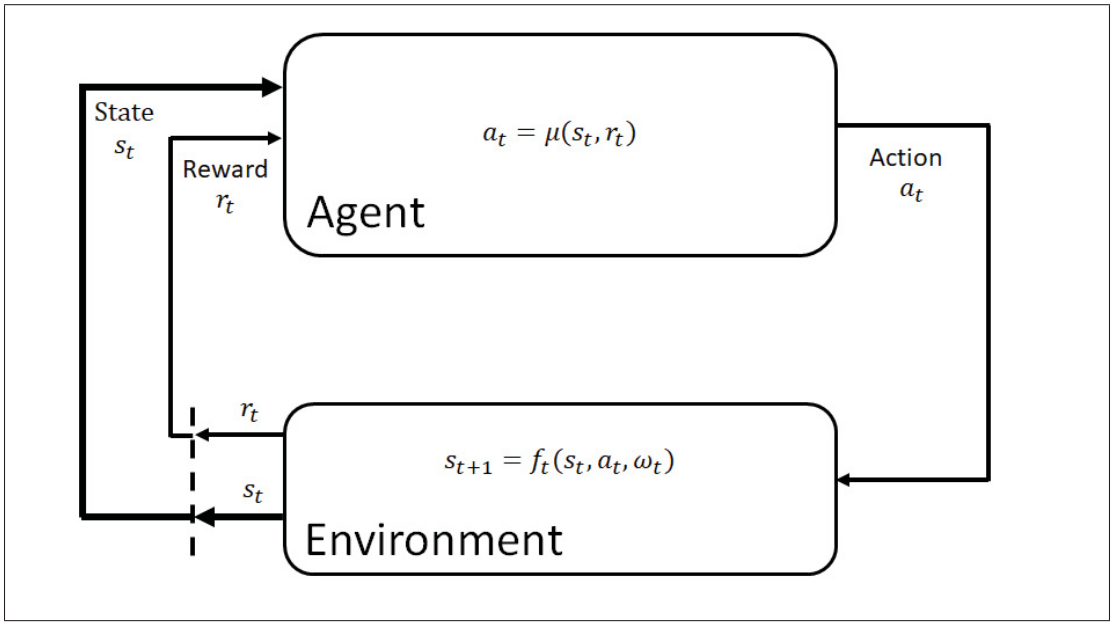


Figure 2.1 Agent-Environment interaction in RL

2.3 QL algorithm: lookup table version

QL algorithm is one of breakthroughs in the area of reinforcement learning. The name of Q-learning was initially chosen because of the variable $Q(s,a)$ which is the estimation value of being in particular state s and taking an action a. The lookup table representation could be considered as the simplest form of QL approach where the Q-factors are stored in a so-called Q-table and the agent will update Q-factors iteratively once it has a new observation of Q-

factors. QL algorithm can also be viewed as sample-based Q-value iteration. Algorithm 2.1 shows a procedural list of the QL algorithm.

Algorithm 2.1 Q-learning algorithm I

Q-learning algorithm I

(Initialization):
A) Initialize the state variable s_0 and Q-factors $Q_t^0(s,a)$ for all states $s \in S$ and feasible decisions a, $t = \{1, \cdots, T\}$
B) Set n = 1
Choose a sample path $\left(\omega_t^n\right)_{t=1}^T$
for t=1:T **do**
    Determine the action using $\varepsilon-$greedy (Section 2.5)
    Sample $\omega_t^n = \omega_t(\omega^n)$ and compute the next state $s_{t+1}^n = f_t(s_t^n, a_t^n, \omega_t^n)$
    Compute the new estimation of Q-factors (Section 2.7)
    Update the new estimation of Q-factors (Section 2.7)
end for
if n$\leq$ N **then**
    Increment n and go to step 2
else
    Return the Q-factors $(Q_t)_{t=1}^T$
end if

This algorithm can be summarized in four main steps. At the first step, the parameters and variables used in the algorithm are required to be initialized with the appropriate values. Also, the state and action variable should be discretized as well. Second, the uncertainty of environment should be determined in a form of the sample data trajectory. The data generated here for random variables are also called as training dataset with the cardinality of N. Towards the third step, the algorithm takes and applies the action, observes a reward and therefore update the initial value of Q-factors based on a new observation. Once the third step is done for either all time steps and training data set, the algorithm returns the table of Q-factors where it contains the estimated action-value functions. This table is then applied to an actual problem where the random variables mentioned above could be replaced by simulation dataset.

In the following sections of this Chapter, we will introduce the principles of QL algorithm as well as explaining some key features such as how to update the Q-factors or how to take an

action during the training process. As a result, this Chapter will go through the key concepts as Bellman principle of optimality, Monte Carlo algorithms, and the concept of rewards. At the end, the QL approach will be extended by the use of a continuous function approximation instead of the lookup table representation.

2.4 Monte Carlo property

As it was mentioned before, DP techniques need to have a prior knowledge on the model of the environment, like transition probabilities and the immediate reward functions. Nonetheless, having an access to the exact model of a system is not possible in many cases. In fact, in the case of large-scale system, building the exact model of a system could be a tough task. One of the efficient and simple solution is to estimate the value function and accordingly the (near)-optimal policy directly from experience. In Monte Carlo (MC) technique, the principle is to estimate the model of the system by using random sampling simulations. In this algorithm, four main steps are required to be performed to complete the numerical analysis. The first step is to define the random variables in order to extract the corresponding statistical properties. Second, by using the determined statistical properties, we need to generate a large set of possible data trajectories for random variables. At the third step, a deterministic calculation will be performed by the use of already generated data trajectories. Eventually, the results will be analyzed statistically.

As it was mentioned earlier, QL algorithm tries to estimate the action-value function within the process of experiencing. This experience can be obtained by sampling the sequences of states, actions, and rewards by communicating with the environment. At this point, it can be seen that QL uses the MC property to avoid performing an optimization over random variable. Theoretically speaking, the basic principle in MC algorithm is to produce a number of samples from the random variable like $v_1, v_2, \cdots, v_N$ and consequently tries to estimate the mean of the random variable by computing a sample mean as below:

$$M_N = \frac{1}{N} \sum_{i=1}^{N} v_i. \tag{2.1}$$

where this sample mean value can be calculated recursively with respect to

$$M_{N+1} = M_N + \frac{1}{N+1}(v_{N+1-M_N}), \tag{2.2}$$

where it begins with $M_1 = v_1$. As it can be seen, MC technique utilizes the mean return of a large number of samples of random variables in order to approximate either the value function V_π or the action-state value function Q_π. One of the primary properties is the number of samples used by the MC algorithm, therefore, as more samples of the random variable is observed, better convergence will occur in the estimated value of the function.

2.5 Exploration/Exploitation

One of the basic and essential features of the QL approach is that the agent does not have any prior information about the environment. However, the agent follows a strategy of learning the uncertain environment within the process of Exploration/Exploitation so that it can estimate the value of Q-function, thereby, determining an optimal action. In QL approach, the agent sometimes tries to reach the high-value immediate rewards by selecting the action with the best value of Q-factor under the current state. This is commonly considered as **Exploitation**. While the agent learns the environment through the actions it tries, it should explore new decisions so that it can be brought in either a better or worse position in the future. This procedure is also known as **Exploration**. Following these strategies, the agent has a possibility of revealing better rewards according to the feedback obtained from the environment.

The basic challenge in QL approach is to keep a balance between exploration and exploitation. Usually, too much exploration not only results in a less accumulated reward but it wastes the time by exploring an improper part of the environment. On the other hand, too much exploitation cannot be either a good choice as it can keep the agent in a local optimum. This issue is known as the exploration/exploitation dilemma. There are several common approaches in order to overcome this problem in the framework of QL algorithm. One of the most commonly used exploration techniques is ε-greedy. The benefit of this choice is that it does not depend on

any specific information of environment. In the ε-greedy approach, the agent acts in a greedy way with probability of $1-\varepsilon$. This can be interpreted as the case of choosing an action with the highest measured Q-factor formulated as below:

$$a_t^n = \underset{a_t \in \mathbb{A}}{argmax}\, Q_t^{n-1}(s_t^n, a_t) \tag{2.3}$$

where a_t^n is the action taken at time period t under scenario n. The $Q_t^{n-1}(s_t^n, a_t)$ is also the value of Q-factors have been observed so far. Afterward, a non-greedy exploratory action will also be taken with a probability of ε which is the indication of taking a random action uniformly without considering its value.

In ε-greedy, there are possibly two ways of choosing the value of ε. One could consider a fixed exploration rate through the whole process of training by assigning a constant value for ε. Assuming $\varepsilon > 0$, the QL approach will converge to the optimal action-value functions as well as optimal policies with probability 1, if $n \to \infty$. On the other hand, we can also specify an alternative version of ε-greedy approach by considering a varying rate of exploration during the training process. Once the agent starts exploring an environment, some states may have visited many number of times. Obviously, these states are not prior to be explored anymore compared to the ones have been never explored. One possible solution to tackle such a problem is to decrease the rate of exploration with respect to the number of visit a state already has. In this case, the following formula has been considered for the value of ε:

$$\varepsilon^n = \frac{\varepsilon_0}{N^n(s)} \tag{2.4}$$

Where n is the number of iteration, varying from one up to the cardinality of training dataset ($n \in [1, N]$). $N^n(s)$ is the number of times a state s has been visited already. According to Powell (2011), this version of ε-greedy guarantees that all states will be visited infinitely if $n \to \infty$, therefore, the algorithm converges to the optimal values. The performance of ε-greedy approach could be improved by considering Boltzmann exploration strategy where the exploration process does not consider an equal value between all the actions in contrast with ε-

greedy. In this approach, the probability of taking an action is ranked based on their estimated values of Q-function so that the one with highest action selection probability will be selected and all other actions are then weighted proportionally with respect to their Q-values. In this way, the probability of taking an action a at a particular state s can be computed as below:

$$P(s,a) = \frac{exp\{-Q(s,a)/T^n\}}{\sum_{a\in\mathbb{A}} exp\{-Q(s,a)/T^n\}} \tag{2.5}$$

Here, T is a temperature parameter which controls the probability distribution and $Q(s,a)$ is the Q-factor for corresponding pair of the state and the action. In this formulation, by an increase of the temperature, the probability of selecting various actions becomes similar which means that the agent acts more randomly (exploration). On the contrary, low temperature results in a bigger difference in the probability of taking actions which are the indication of acting according to the best actions have been tried so far. Based on this concept, the temperature T Gradually reduces over scenarios to limit the exploration process by the use of the following equation:

$$T^n(s) = \frac{N^n(s)}{\delta Q(s)} \tag{2.6}$$

where $\delta Q(s) = \max_a |Q(s,a^{max}(s)) - Q(s,a)|$. In this approach, there is a more reasonable choice of decision that put more attention on better actions, while it provides some extent of degree on exploration. This idea is good for the class of problems with a relatively small number of decisions a is relatively small, while, there is a need to have a kind of pre-estimation on the values of $Q(s,a)$. Singh *et al.* (2000) shows that the QL approach is still convergent by using the Boltzmann exploration rule.

2.6 Return function

In QL, an agent is assumed to receive rewards on each time steps and the purpose is to try to maximize the expected value of receiving rewards from environment over the time horizon. So, the reward function $r_t(s_t,a_t)$ is quite important feature for the agent. The question here is that how potential could be the impact of current decision of agent on the future reward. The

answer to this question is fairly simplified in the term of time horizon that specify how long an agent wants to live in a particular environment.

QL approach or generally all techniques that can be formulated in Markovian domain are classified in two groups with respect to the time horizon: finite horizon and infinite horizon. In finite horizon, QL algorithm deals with a finite number of steps over the future instants. So the return function is accumulated rewards over the finite time horizon. In contrast, the infinite horizon QL could go further and consider an infinite sequence of communications between the agent and the environment. In this study, the discounted finite horizon QL algorithm is considered, consequently, the return function is the summation of the discounted rewards that the agent receives from the initial step up to the last instant as below:

$$R_t = c_{t+1} + \gamma c_{t+2} + \gamma^2 c_{t+3} + \cdots + \gamma^{T-1} c_T \tag{2.7}$$

Here, T is the time horizon, R_t is the reward received after t time steps. The discount factor γ determines the present value of future rewards where $\gamma \in [0,1]$. If $\gamma = 0$, the agent is said to be myopic and only considers immediate rewards. As γ increases, the returns increase and the agent gives more consideration to future rewards.

2.7 Learn/Update process

QL and in general most RL algorithms compute the value function of DP to obtain an optimal policy. This can be considered as the basic reason why RL gets its main roots from DP algorithm. In DP, the value function is computed by using the Bellman optimality equation as equation (1.1). The basic point is that the expectation will be dropped from equation (1.1) as a result of considering Monte Carlo property. This is because of the random sampling simulation provided by MC algorithm as mentioned in the section 2.4. The Bellman equation can then be rewritten in a deterministic from as:

$$V_t^{\star}(s_t) = \max_{\pi} \{ r_t(s_t, a_t) + V_{t+1}(s_{t+1}) \} \tag{2.8}$$

In DP, the value function is associated with the state of the system. However, in QL algorithm, the so called Q-factor (action-value function) is a function of state-action pair. Considering this point, the bellman equation can be reformulated in term of Q-factor as:

$$Q_t^{\star}(s_t, a_t) = \max_{\pi} \{ r_t(s_t, a_t) + \gamma Q_{t+1}(s_{t+1}, a_{t+1}) \} \tag{2.9}$$

here $Q_t(s,a)$ is the value of action-value function under the given sample scenario path (ω_t^n), starts with the state s_t, takes the action a_t based on a specific policy π. The equation (2.9) indicates that the value of being in a state is equal to the summation of immediate reward plus the discounted value of being in the next state. The additional parameter γ is a discounted factor that can also be considered for DP equation in a similar way. It can be seen that QL algorithm is a sample-based Q-value iteration where the values are expressed by Q-factors. So, the Q-factors can be estimated by the use of following value iteration formula:

$$Q_t(s_t, a_t) \leftarrow \{ r_t(s_t, a_t) + \gamma \max_{a_{t+1} \in \mathbb{A}} Q_{t+1}(s_{t+1}, a_{t+1}) \} \tag{2.10}$$

As mentioned above, the QL algorithm is equivalent to regular value iteration, while there is a slightly difference on the updating process compared to the basic value iteration algorithm. In QL approach, two steps are mainly required to update the Q-factors. The first step is to compute the new estimation of action-value function observed over the particular scenario n that can be calculated by using the following equation:

$$\hat{q}_t^n = r_t(s_t^n, a_t^n) + \gamma \max_{a_{t+1}} Q_{t+1}^{n-1}(s_{t+1}, a_{t+1}) \tag{2.11}$$

Here q^n is the current estimation of Q-factor evaluated over scenario n. To clarify more, the agent can update the estimate of action-value function based on the already observed value $Q^{n-1}(s_{t+1}, a_{t+1})$ plus the immediate rewards observed from environment $r_t(s_t^n, a_t^n)$. The second step is then to calculate the new estimation of action- value function based on the current estimation on scenario n and previously observed value on scenario $n-1$. This averaging can

be done by the use of Robbins-Monro algorithm as follows:

$$Q_t^n(s^n,a^n) = (1-\alpha_n)\, Q_t^{n-1}(s^n,a^n) + \alpha_n \hat{q}^n. \tag{2.12}$$

here α_n is a learning rate (step size) in which it can hold a value from zero up to one ($\alpha_n \in [0,1]$). The update formula applied above calculates a stochastic approximation of the $Q_t^n(s^n,a^n)$, which indicates that:

$$New\ Estimate \leftarrow Old\ estimate + Step\_size.[Current\ estimation - Old\ estimate] \tag{2.13}$$

Here, the new estimation of Q-factor is shown by $\hat{q^n}$ in equation (2.12). Meanwhile, the term $[Current\ estimation - Old\ estimate]$ is the indication of the error in the estimation over the current scenario. This error can be decreased by choosing a step towards the optimal value of Q-factor. Considering the equation (2.12), the step size or learning rate is one of the essential factor that has a major impact on the QL algorithm convergence. In practice, two issues are usually addressed within the selection of good step size rule. The first point is summarized in capability of the step size rule to provide a provably convergent algorithm. On the other hand, the second one is the question of whether the step size function can produce the fastest rate of convergence. In this study, two common choices of step size function are selected. The primary choice of step-size is chosen by considering the Harmonic step-size which is the general form of $\frac{1}{n}$ defined by:

$$\alpha_n = \frac{a}{a+n-1} \tag{2.14}$$

Here a is a constant value that affects greatly in the rate of convergence in QL algorithm. This rule is able to satisfy the conditions for convergence of QL approach. Considering $\alpha > 1$ provides larger step-size compared with $\frac{1}{n}$ while the increase in a will resulted in slow rate of convergence. An alternative form of step-size function is McClain's Formula that is defined as follows:

$$\alpha_n = \frac{\alpha_{n-1}}{1+\alpha_{n-1}-\bar{\alpha}} \tag{2.15}$$

here, $\bar{\alpha}$ is specific parameter. According to Powell (2011), the interesting characteristic of this formula is the possibility of preserving the benefits from the step-size rule of $\frac{1}{n}$ for stationary data and constant step-size that is appropriate for non-stationary data. This formula can be quite efficient in the case that we don't know how many iterations is required to be in the convergent as well as being appropriate for nonstationary environments. This rules also satisfies the convergence property of QL algorithm while the choice of $\bar{\alpha}$ will affect the rate of convergence.

2.8 Overview of complete QL approach

Considering all above-mentioned details of QL approach, the complete procedural list of the QL algorithm can be seen in Algorithm 2.2.

Algorithm 2.2 Q-learning algorithm II

Q-learning algorithm II (look-up table representation)

(Initialization):
A) Initialize the state variable s_0 and Q-factors $Q_t^0(s,a)$ for all states $s \in S$ and feasible decisions a, $t = \{1, \cdots, T\}$
B) Set n = 1
Choose a sample path $\left(\omega_t^n\right)_{t=1}^T$
for t=1:T **do**
    (Exploration) With probability ε choose an action a_t^n randomly form the set of possible actions $\mathbb{A}$. (Exploitation) With probability $1-\varepsilon$ choose a_t^n by

$$a_t^n = \underset{a_t \in \mathbb{A}}{argmax}\, Q_t^{n-1}(s_t^n, a_t)$$

    Sample $\omega_t^n = \omega_t(\omega^n)$ and compute the next state $s_{t+1}^n = f_t(s_t^n, a_t^n, \omega_t^n)$.
    Compute the new estimation of Q-factor by

$$\hat{q}_t^n = r_t(s_t^n, a_t^n) + \gamma \max_{a_{t+1} \in \mathbb{A}_{t+1}} Q_{t+1}^{n-1}(s_{t+1}^n, a_{t+1})$$

    Update Q_t^n using

$$Q_t^n(s_t, a_t) = (1-\alpha_n) Q_t^{n-1}(s^n, a^n) + \alpha_n \hat{q}_t^n.$$

end for
if n$\leq$ N **then**
    Increment n and go to step 2.
else
    Return the Q-factors $(Q_t^n)_{t=1}^T$
end if

2.9 Function approximation

In QL, the value functions play an important role in selecting an action or evaluating the current state that can be used to select an action. The most common representation of state value functions for discrete state and action spaces is to employ a look-up table. Considering this point, the problems with finite state space can be easily handled using a lookup table to store the value for a pair of state and action. In spite of the simplicity, most problems exist in real-world applications have quite large or infinite state space, including continuous variables. At this point, the lookup table representation of QL approach will be infeasible due to the need to store elements of lookup-table as it needs large memory and long computational time.

QL can be adapted with function approximation techniques in order to overcome above mentioned drawbacks. Here, the principle idea is to estimate the value of Q-factors for unseen pair of state-action based on the already visited states. In the literature, several possible methodologies have been introduced such as: function fitting (neural networks and regression), function interpolation (K-nearest-neighbors and Kernel methods), and state aggregation (Gosavi 2003). Using one of these function approximation techniques, one can create the following approximated action-value function as:

$$\begin{aligned} Q(s_t, a_t) &\approx \Psi_t(s_t, a_t) \\ &= \sum_{f=1}^{F} \theta_{f,t} \phi_f(\underline{s}_t, \underline{a}_t) \end{aligned} \tag{2.16}$$

Here, the basis function is the linear combination of state and action vector and parameter F is the cardinality of this function. Considering a linear approximation techniques, recursive least square approach is used as a result of non-stationary Q-factors where it would be preferred to consider more weight on more recent observations. In this approach, the main objective is to minimize a weighted sum of errors

$$\min_{\theta_{f,t}} \sum_{m=1}^{n} \lambda_t^{n-m} \left(\hat{q}_t^m - (\theta_{0,t} + \sum_{f=1}^{F} \theta_{f,t} \phi_{f,t}^m) \right)^2 \tag{2.17}$$

where λ is a factor to discount older observations with the new ones. The updating equation for θ is

$$\theta_t^n = \theta_t^{n-1} - H_t^n \phi_t^n \hat{\varepsilon}_t^n, \tag{2.18}$$

Here, H^n is a matrix computed using

$$H_t^n = \frac{1}{\gamma_t^n} B_t^{n-1}. \tag{2.19}$$

The error $\hat{\varepsilon}^n$ is computed by

$$\hat{\varepsilon}_t^n = Q_{s,a}(\theta_t^{n-1}) - \hat{q}_t^n. \tag{2.20}$$

In a regression problem, it is common to consider the error as "actual minus predicted" while we are using "predicted minus actual". A possible reason is to converge the predicted model to the actual one. B^{n-1} is an $|F|$ by $|F|$ matrix, which is calculated recursively by

$$B_t^n = \frac{1}{\lambda_t^n}\Big(B_t^{n-1} - \frac{1}{\gamma_t^n}(B_t^{n-1}\phi_{f,t}^n(\phi_{f,t}^n)^T B_t^{n-1})\Big) \tag{2.21}$$

and γ_t^n is a scalar computed by

$$\gamma_t^n = \lambda_t^n + (\phi_{f,t}^n)^T B_t^{n-1} \phi_{f,t}^n. \tag{2.22}$$

Here, λ works in a similar way compared to the step size, although in the opposite direction. The λ can be computed by

$$\lambda_t^n = \alpha_{n-1}\Big(\frac{1-\alpha_n}{\alpha_n}\Big). \tag{2.23}$$

Considering $\lambda = 1$, an equal weight will be considered on all observations, while decreasing the value of λ considers more weight on recent observations. Considering this strategy, one could avoid to store the table of Q-factor. Obviously, less storage space is needed and a large state space can thus be handled. Algorithm 2.3 shows a general pseudocode of QL approach by the use of a function approximation technique.

Algorithm 2.3 Approximated Q-learning Algorithm

Approximated Q-learning Algorithm

(Initialization):
A) Initialize the state variable s_0 and Q-factors $Q_t^0(s_t, a_t)$ for all states x_t and feasible decisions a_t, $t = \{1, \cdots, T\}$
B) Set n = 1
for t=1:T **do**
    (Exploration) With probability ε choose an action a_t^n randomly form the set of possible actions $\mathbb{A}$. (Exploitation) With probability $1-\varepsilon$ choose a_t^n by

$$a_t^n = \underset{a_t^n \in \mathbb{A}}{argmax}\, r(s_t^n, a_t^n) + \gamma \max_{a_{t+1} \in \mathbb{A}} \Psi_{t+1}(s_{t+1}^n, a_{t+1})$$

    Sample $\omega_t^n = \omega_t(\omega^n)$ and compute the next state $s_{t+1}^n = f_t(s_t^n, a_t^n, \omega_t^n)$
    Compute the new estimation of Q-factor by

$$\hat{q}_t^n = r_t(s_t^n, a_t^n) + \gamma \max_{a_{t+1} \in \mathbb{A}_{t+1}} \Psi_{t+1}^{n-1}(s_{t+1}^n, a_{t+1})$$

    Update the function $\Psi_t(s_{t+1}^n, a_{t+1}^n)$ using a desired function approximation.
end for
if n$\leq$ N **then**
    Increment n and go to step 2.
else
    Return the function $\left(\Psi_t(s_t, a_t)\right)_{t=1}^T$
end if

2.10 Summary

In this Chapter, an overview and description of QL algorithm were presented. In the first step, the main idea and theory behind the QL were presented. This was followed by describing the lookup table representation of QL approach. Then, a special attention was provided in the main features of QL algorithm as Monte Carlo property, Exploration/Exploitation mechanism, and learn/update process. At the end, the QL approach adapted with function approximation technique was introduced in order to reduce the dimensionality problem.

CHAPTER 3

Q-LEARNING IN OPERATION OF MULTIRESERVOIR SYSTEMS

3.1 Introduction

Once the training process of Q-factors are completed, the next step is to use the trained Q-factors in solving the specific problem. Consequently, this Chapter is concentrated in two main parts where the main purpose is to divide the general problem and the optimization technique. In this way, the first section presents the general mathematical formulation of operating multireservoir systems, in particular, the short-term operation problem. As we explained in Chapter 1, the problem could be solved by the use of numerous approaches. However, these techniques, in particular DP based algorithms, may not be able to tackle the operation problem of multireservoir systems due to the curse of modeling and dimensionality. With respect to this point, our hypothesis is that QL algorithm could be the suitable candidate to deal with this type of problem as it avoids the above-mentioned limitation. So, the second section will be concentrated on possible ways of assisting the quality of basic QL algorithm.

3.2 Multireservoir operation problem

In this section, the main goal is to present the general short-term operation problem of multireservoir systems where all the objectives and constraints are formulated within the mathematical concept. To accomplish this goal, the first step is to define the notations that are going to be used in the multireservoir operation problem as follow:

$\mathbb{I}$ Set of reservoirs, where $i \in \mathbb{I} = \{1, \cdots, I\}$

$\mathbb{J}$ Set of immediate upstream reservoirs, where $j \in \mathbb{J} \subset \mathbb{I}$

$\mathbb{T}$ Set of time step on operation of multireservoir systems, where $t \in \mathbb{T} = \{1, \cdots, T\}$

$s_{i,t}$ Water storage in m^3 for reservoir i in time period t

$x_{i,t}$ Pool elevation in m for reservoir i in time period t

$a_{i,t}$	Water release in m^3 for reservoir i in time period t
$a_{i,t}^{tur}$	Water release from turbine in m^3 for reservoir i in time period t
$sp_{i,t}$	Water spillage in m^3 for reservoir i in time period t
$\omega_{i,t}$	Natural water inflow in m^3 for reservoir i in time period t
$x_{i,t}^{min}$	Minimum pool elevation in m for reservoir i in time period t
$x_{i,t}^{max}$	Maximum pool elevation in m for reservoir i in time period t
$a_{i,t}^{min}$	Minimum water release in m^3 for reservoir i in time period t
$a_{i,t}^{max}$	Maximum water release in m^3 for reservoir i in time period t
$CL_{i,t}$	Violation variable on pool elevation for reservoir i in time period t
$CO_{i,t}$	Violation variable on minimum outflow for reservoir i in time period t
M_{CL_i}	Violation penalty on pool elevation for reservoir i
M_{CO_i}	Violation penalty on minimum outflow for reservoir i
$e_{i,t}$	Energy production by turbine in GWH for reservoir i in time period t
$V(x_{i,t})$	Water value function at the end of horizon
$C_{i,t}(x_{i,t}, \omega_{i,t}, a_{i,t})$	Cost function for reservoir i in time period t
π	Policy, decision making rule

Considering the above-mentioned notations, the next step is to formulate the objective function of multireservoir operation problems. So, the general objective function required to operate I number of reservoirs over a specific time horizon of N could be determined as:

$$\max_{\pi} \mathbb{E}\Big\{\sum_{t\in\mathbb{T}}\sum_{i\in\mathbb{I}} C_{i,t}(x_{i,t}, \omega_{i,t}, a_{i,t})\Big\} \tag{3.1}$$

where $C_{i,t}$ is the cost function that we would like to optimize over the uncertain water inflow $\omega_{i,t}$. The reason to compute the excepted value of the objective function comes from the fact that the uncertainty of the incoming water inflow still varies even considering the short-term planning problem of multireservoir system. The solution of this problem is to derive the optimal policy, thereby, computing the optimal water release policy $a_{i,t}$ at time instant t for reservoir i. In this problems, the cost function associates to the $x_{i,t}$ level of water in $\{m\}$, $\omega_{i,t}$

natural water inflow in $\{m^3\}$ and the decision for the water release value denoted by $a_{i,t}$. The cost function can be written as:

$$C_{i,t}(x_{i,t}, \omega_{i,t}, a_{i,t}) := \begin{cases} e(x_{i,t}, \omega_{i,t}, a_{i,t}) & \forall i \in \mathbb{I}, t \in \{1, \cdots, T-1\} \\ V(x_{i,t}) & \forall i \in \mathbb{I}, t = T \end{cases} \tag{3.2}$$

The above cost function consists of two parts. The first part is responsible to compute the hydroelectric power production on reservoir i for all time step except the last one. The second part is the final value of water at the end of the horizon. This function is considered to avoid making reservoir empty at the end of time horizon. Afterward, the key step is to compute the reservoir dynamics where they can be used by optimization technique. One common technique in the literature is to use of the mass balance equation as follow:

$$s_{i,t+1} = s_{i,t} + \left(\omega_{i,t} - a_{i,t} + \sum_{j \in \mathbb{J}} a_{j,t}\right) \tag{3.3}$$

where $s_{i,t+1}$ is the storage of reservoir i at time step $t+1$. Computing the water storage at next time period allows us to calculate other systems' parameters like pool elevation. The pool elevation is commonly pretested as a function of water storage for each reservoir as:

$$x_{i,t} = H_i(s_{i,t}) \tag{3.4}$$

where, H_i is a nonlinear mapping function that computes the pool elevation based on the corresponding water storage of reservoirs. A key point here is that the water release denoted by $a_{i,t}$ consists of two parts. The first part is the water outflow passed by turbines and the second one is the water outflow passed as the water spillage. Considering this point, the water spillage can also be computed by:

$$sp_{i,t} = max(0, a_{i,t} - a_{i,t}^{tur}) \tag{3.5}$$

where, the $a_{i,t}^{tur}$ is the maximum amount of water can be released from turbine for reservoir i at time period t. In the next step, several constraints also exist in the operation problem of multireservoir system. The general constraints are required to be considered are:

$$\begin{cases} x_{i,t}^{min} \leq x_{i,t} \leq x_{i,t}^{max} & \forall i \in \mathbb{I}, t \in \mathbb{T} \\ a_{i,t}^{min} \leq a_{i,t} < a_{i,t}^{max} & \forall i \in \mathbb{I}, t \in \mathbb{T} \end{cases} \tag{3.6}$$

where the $x_{i,t}^{min}$ and $x_{i,t}^{max}$ correspond to the minimum and maximum pool elevation of reservoir i at time period t. Similarly, $a_{i,t}^{min}$ and $a_{i,t}^{max}$ are the minimum and maximum boundary for the water outflow. The constraint on water outflow consists of two parts. The upper limitation of water outflow is the maximum capacity of a reservoir to release water. This limitation is a hard constraint as a result of system limitation, and it does not have a stochastic nature. However, the minimum outflow which is the environmental constraint depends on the stochastic water inflow, so, it cannot be handled in a deterministic way. Practically, this constraint can be violated under a condition when a reservoir does not have enough amount of water to release. So, the constraint on the water outflow could be simplified as a case of violating the minimum water outflow. A possible solution to satisfy these constraints is to use and modify the cost function. In this case, the cost function can be penalized once the constraints are violated. Considering this point, the cost function will be modified as:

$$C_{i,t}(x_{i,t}, \omega_{i,t}, a_{i,t}) = e_{i,t}(x_{i,t}, \omega_{i,t}, a_{i,t}) - M_{CO_i} CO_{i,t} \tag{3.7}$$

where

$$CO_{i,t} := \begin{cases} a_{i,t}^{min} - a_{i,t} & if \quad a_{i,t} < a_{i,t}^{min} \\ 0 & if \quad a_{i,t} > a_{i,t}^{min} \end{cases} \tag{3.8}$$

Similarly, considering the constraint on pool elevation, the cost function can be reformulated as:

$$C_{i,t}(x_{i,t}, \omega_{i,t}, a_{i,t}) = e_{i,t}(x_{i,t}, \omega_{i,t}, a_{i,t}) - M_{CL_i} CL_{i,t} - M_{CO_i} CO_{i,t} \tag{3.9}$$

where

$$CL_{i,t} := \begin{cases} |x_{i,t} - x_{i,t}^{min}| & if \quad x_{i,t} < x_{i,t}^{min} \\ |x_{i,t} - x_{i,t}^{max}| & if \quad x_{i,t} > x_{i,t}^{max} \\ 0 & if \quad x_{i,t} \in [x_{i,t}^{min}, x_{i,t}^{max}] \end{cases} \tag{3.10}$$

In this strategy, the violation variable changes proportionally once it passes the boundary. In the next step, the constraint on the water outflow can also be satisfied by using a similar approach, however, there is no need to include the violation on the maximum boundary of water outflow. Generally, constraints could be integrated in to the mathematical model in different ways depending on their priority. This means that they can either represented within a linear, non-linear or exponential function. In this study, we have considered a linear function for the constraints.

3.3 QL in multireservoir operation problem

In this section, the main goal is to solve the previously mentioned multireservoir operation problem by the use of QL approach and providing the basic step to get an answer on research questions highlighted in the introduction. As it was mentioned before, the QL algorithm possesses several interesting features that can be the appropriate technique to solve a large-scale optimization problem. Considering this point, the first step is taken to apply the basic lookup table version of QL approach in the multireservoir operation problem. Through this part, we will go through the definition of state/action variables used in QL algorithm as well as some basic initializations of the algorithm (Research Question 1). While, assuming the performance of basic QL approach, the main motivation is to improve the performance of the QL by pro-

viding the most suitable input's of the algorithm. In the subsection 3.3.3, the calibration of the algorithm's parameters are explained (Research Question 2). Afterward, subsection 3.3.4 tries to explain different methodologies in generation of a more appropriate training dataset (Research Question 3). In the subsection 3.3.5, an alternative form of the cost function used in QL approach will be proposed (Research Question 4). Finally, subsection 3.3.6 provides the basic features of applying the linear approximation technique on multireservoir operation problem (Research Question 5).

3.3.1 State and action definition

The state variable is one of the main components in the design of QL algorithm. In the application of multireservoir operation system, the state variable is commonly taken as either water level/storage or combination of water level/storage and natural water inflow to the reservoir. In this study, the combination of water level and the natural water inflow is considered for the state vector of QL approach. So, the state vector can be represented as:

$$\underline{s}_t = \{\underline{\omega}^n_{t-1}, x_{1,t}, \cdots, x_{I,t}\} := \{\omega^n_{1,t-1}, \cdots, \omega^n_{I,t-1}, x_{1,t}, \cdots, x_{I,t}\} \tag{3.11}$$

In general, the reservoir dynamics like storage, water inflow/outflow are time dependent variable, so, they should basically consider as continuous variables due to the physical characteristics of the system. However, the nature of optimization technique requires to discretize the continuous variables to the discrete variable. With respect to this point, the pool elevation of each reservoir $x_{i,t}$ is discretized with a scale of n_x, similarly, the natural water inflow $\omega_{i,t}$ is also transformed to a discrete space with a grid of n_ω. Sometimes, the system of reservoirs are closely located with each other so that the incoming water inflows are highly positively correlated. In this case, a possible simplification could be considered to reduce the complexity of the problem by using an approximation value of hydrological state variables. So, instead of using the random variable vector, a mean value of this vector is considered in the state vector.

Therefore, the state variable can be reformulated as below:

$$\underline{s}_t = \{\bar{\omega}_{t-1}, x_{1,t}, \cdots, x_{I,t}\} \tag{3.12}$$

where

$$\bar{\omega}_{t-1} = \frac{1}{I}\sum_{i=1}^{I} \omega_{i,t-1} \tag{3.13}$$

It is common to consider a fix discretization grid for the continuous variable over the all time steps as it is also followed in this study. The boundary on the grid of state variable is defined by the maximum and minimum value of pool elevation $x_{i,t} \in [x_{i,t}^{min}, x_{i,t}^{max}]$ and similarly for the natural water inflow $\omega_{i,t-1} \in [0, \omega_{i,t-1}^{max}]$. The $\omega_{i,t-1}^{max}$ is the maximum possible water inflow to reservoir i at time period $t-1$ based on the observed historical dataset. Further to the state variable, the water release of each reservoir is the typical choice of the decision variable in the reservoir operational planning problem. In this case, the action vector is the combination of possible water release in reservoir i at time step t, where $\underline{a}_t = \{a_{1,t}, \cdots, a_{I,t}\}$. In the QL algorithm, the action variable belong to the discrete set ($a_{i,t} \in [a_{i,t}^{min}, a_{i,t}^{max}]$) where the boundaries correspond to the minimum and maximum possible water outflow on each reservoir and the scale of discretization is n_a.

3.3.2 Initialization of QL algorithm

Recalling from Algorithm 2.1, the update of the action-value function is performed by using the following formula:

$$Q_t^n(\underline{s}_t, \underline{a}_t) = Q_t^{n-1}(\underline{s}_t, \underline{a}_t) + \alpha_n\{\gamma[r_t(\underline{s}_t, \underline{a}_t) + \underset{\underline{a}_{t+1}}{max} Q_{t+1}^{n-1}(\underline{s}_{t+1}, \underline{a}_{t+1})] - Q_t^{n-1}(\underline{s}_t, \underline{a}_t)\} \tag{3.14}$$

Considering the above formula in the application of multireservoir operation problem, the agent takes an action at each time step as a sequence of water releases and applies them to the environment (multireservoir systems). Then, the environment evaluates the reward function as $r_t(\underline{s}_t, \underline{a}_t)$ in a energy production unit GWH as well as the state of the system in the next time step by $\underline{s}_{t+1}$. This evaluation is done by using a simulator of multireservoir systems. However,

considering the procedural list of QL shown in Algorithm 2.1, the first step in implementation of QL algorithm is to initialize state variable (s_0) and Q-factors ($Q_t^0(s_t, a_t)$). As we mentioned, the state vector consists of two parts: one corresponds to the pool elevation ($x_{i,t}$) and the other is the natural water inflow to the system. In this study, the initial value of pool elevation is provided by industrial partner. On the side of water inflow $\bar{\omega}_{i,t-1}$, we consider the water inflow observed in the day before the starting date of operation on historical dataset.

The next step in calculation of $Q_t^n(\underline{s}_t, \underline{a}_t)$ is to know the initial value of Q-factor itself. In the literature, the initial value of Q-factors are considered sometimes randomly or zero and it is noted that the algorithm should converge to the optimal value by having enough exploration. However, we tried to carefully assigned the initial value so the algorithm can converge to the optimal solution as fast as possible. In this study, the final value of water is approximated based on the water storage of each reservoir, given the amount of energy can be produced based on the specific storage. The function can be written as:

$$\begin{aligned} V(x_{i,t}) &= H(s_{i,t}, u_{i,t}^{turbine}) \\ &\approx C_i \times s_{i,t} \end{aligned} \tag{3.15}$$

where $s_{i,t}$ is the water storage at reservoir i by hm^3. The parameter C_i is the average production efficiency of a reservoir i as a function of power over flow rate from turbine.

3.3.3 Calibration of QL parameters

As it was discussed in the section 2.5, two types of exploration strategies are adapted in this study. First, the ε-greedy algorithm with fix rate of exploration is considered where $\varepsilon \in \{25\%, 50\%, 75\%, 100\%\}$. In the second attempt, the following function has been assigned in order to compute the value of ε in a suitable way.

$$\varepsilon = \frac{\varepsilon_0}{N^n(\underline{s}_t)} \tag{3.16}$$

where $\varepsilon_0 \in \{25\%, 50\%, 75\%, 100\%\}$. The $N^n(\underline{s}_t)$ is the number of visits that a particular state vector has been reached so far. It is obvious that the growth in the number of iterations, the exploration reduces, thus the exploitation will be increased as well. Finally, as the index of iteration gets larger and larger, the agent will act according to the greedy actions which are the best policy learned so far.

The learning rate or step size is the main feature that controls the learning process of QL algorithm. To clarify more, it basically provides a control on how fast do we want to adjust the estimation of the state-action value function. In section 2.7, two common choices of functions for learning rate were introduced. In the first step, the Harmonic step-size function is considered where the constant parameter $a \in \{10, 50, 500\}$. The other version of step-size function was McClain's Formula. In this case, the constant parameter is taken as zero, where $\alpha_0 \in \{.25, 0.5, 0.75\}$. These values are adopted based on the literature where they applied the QL algorithm to the long term operation of multireservoir system.

3.3.4 Training dataset

According to Chapter 2, the QL algorithm uses the Monte Carlo property to estimate the value function and consequently to take an optimal action where it uses a set of sample scenario trajectories for random variables. The noticeable point here is that the use of wrong training dataset as a sample scenario path may lead QL approach to become less efficient from the side of performance. In this study, two possible datasets are available to generate the training dataset for random water inflow. The first dataset is the historical one where data were recorded in a specific time period in the past. Another available dataset is the synthetic one in which a hydrological software is being used to generate the random series of water inflow.

In the case of synthetic dataset, a hydrological software called "SAMS" is used for stochastic analysis and modeling of water stream flow. In this software, the stochastic characteristic of water streamflow is formulated based on mathematical models which have been proposed in several studies like (Salas, 1993; Hipel and McLeod, 1994). By the use of these models

and given the recorded historical series, the software can model the statistics of a historical dataset, thereby it can reproduce different water stream flows where they share the same statistical properties. This software provides the following features towards the characteristics of hydrological water inflow:

1) Analyze the stochastic features of annual and seasonal data

2) Ability to deal with single and multiple site

3) Ability to generate unlimited number of sample scenarios

Considering two dataset mentioned above, the next step is to generate the sample scenario trajectories (training dataset) used in the QL algorithm. One possible and basic idea is to sample the data occurred in the exact time period of operation $\{1,\cdots,T\}$ from dataset. To clarify more, if the operations start by d^{th} day of a year, the training dataset is all the water inflows exist between d^{th} and $d^{th}+T$ on the specific year. The training dataset generated by this approach are usually not so large. In this way, a possible solution can be considered by repeating the training dataset m number of times.

In spite of the fact that many studies use the above idea to generate sample scenario indices, this re-sampling strategy may be misleading. One of the key reason here is to make difference between different value of hydrological state variable. If the operation is on the wet period, it is not logical to train the Q-factor based on the inflow occurred in a dry period. To this end, a simple algorithm is introduced in a term of selecting the similar pattern of water inflow without considering the only period of operation. In this strategy, a boundary δ is considered for $\bar{\omega}_0$ that is the water inflow observed at the day before the starting day of operation on a specific year. Then, all the samples of water inflow are compared with this criteria and those that meet the condition are selected in the training dataset. To clarify more, if any samples meet the condition of being in a our desired boundary, it will be selected together with the next T sequence of samples in to our training dataset. This approach will be called **Pre-processing Strategy**.

3.3.5 Surrogate form of cost function

A cost function is an equation used to formulated the desired behaviors of system that are expected to see. In QL, the choice of cost function is quite important in term of converging to the optimal value of Q-factor, thereby obtaining the optimal policy. The key question here is the suitable choice of the cost function with respect to the application of operating multireservoir system. One natural choice is to consider the electricity power production of reservoir system since the main motivation is to have the maximum power production over the specific time horizon. Consequently, the function introduced by equation (3.10) is taken as the first function for our optimization problem.

Further to maximization of hydro-electric power production, it is usually desired to minimize the water spillage of multireservoir systems. In reality, the function responsible for the final value of water is a nonlinear and time variant, consequently, the creation of exact function is too complicated and time demanding. As it was formulated in the section 3.3.1, a simplified and approximated linear function was used on the side of computing the final value of water at the end of horizon. As it can be expected, this modification could not preserve all the features that are provided by using the realistic function. One possible strategy to eliminate the limitation of our simplified function is to add some additional elements to our cost function. So, a simple modification is considered in our cost function by adding the water spillage with a negative sign so that the optimization algorithm could itself try to minimize the rate of water spillage. The new form of cost function can be represented as:

$$C_{i,t} = e_{i,t} - M_{CL_i} CL_{i,t} - M_{CO} CO_t - M_{sp} sp_{i,t} \tag{3.17}$$

where $sp_{i,t}$ is the water spillage for reservoir i at time period t. The parameter M_{sp} is the constant coefficient for penalizing how water spillage is important in the multireservoir operation problem.

3.3.6 Approximated QL on multireservoir operation problem

One of the key advantages provided by lookup table version of QL algorithm is the possibility of avoiding the calculation and storage of transition probability matrix. Despite of this benefit, the use of lookup tables could be impractical by increasing the size of state and action variables. So, the introduced algorithm is adapted in a way that a larger number of state and decision variables could be handled. In this study, linear function approximation technique is used where the state space is continuous rather than a discretized grid. In this way, the function $\Psi_t(s_t, a_t)$ used in Algorithm 2.3 is reformulated considering the new state and action variable as:

$$\begin{aligned} \Psi_t(\underline{s}_t, \underline{a}_t) &= \sum_f \theta_{f,t} \phi_f(\underline{s}_t, \underline{a}_t) \\ &= (\theta_{1,t}, \cdots, \theta_{F,t})^T [\bar{\omega}_{t-1}, x_{1,t}, \cdots, x_{I,t}, a_{1,t}, \cdots, a_{I,t}] \end{aligned} \tag{3.18}$$

where $\theta_{f,t} = (\theta_{1,t}, \cdots, \theta_{F,t})^T$ is the vector of regression coefficients at time period t and $\phi_f(s,a)$ is a basis function. Here, the basis function is the linear combination of state and action vector and parameter F is the cardinality of this function.

3.4 Summary

In this Chapter, we first provided the general problem of operating multireservoir systems within the mathematical notation. Afterward, the QL algorithm was applied to solve this large-scale stochastic optimization problem. At the first step, the basic initializations required to setup the Q-leaning algorithm were presented. Being able to run the basic QL approach on our problem, we then tried to assist the performance of QL approach by suitably providing the best algorithm inputs. To reach this goal, the algorithm parameters are tuned based on different functions and values. Afterward, a simple algorithm was adapted in the generation sample scenario path in order to produce more correlated dataset to train Q-factors. In addition, we studied different the role of assigning different from of cost function. At the end, the QL algorithm was adapted by the use of linear function approximation technique. In this

case, recursive least square approach was used in order to estimate the coefficient of linearized function.

CHAPTER 4

MODEL TESTING AND IMPLEMENTATION

4.1 Introduction

In this Chapter, the main focus is to test and analyze the basic and adapted QL algorithm introduced in Chapter 3. In the first step, a quick introduction to the case study selected in this project including the reservoir operating dynamics and characteristics will be presented and discussed. Afterward, the QL algorithm will be applied to solve the optimization problem of the selected multireservoir system. To assist the performance of QL approach, the modifications introduced in the previous Chapter will be implemented and the results are then compared to find the best algorithm's inputs.

Performing all steps mentioned above, we will put a step forward in providing a complete simulation analysis for verifying the effectiveness of proposed approach on the short-term management of multireservoir system. In this project, the QL approach initially will run in training mode, where the action-value function and the optimal control policies are being trained. Once the training process is completed, the algorithm will be applied to solve the actual daily operation problem using operational subset of data that is independent of the training dataset.

4.2 Description of Romaine complex

The Romaine multireservoir in Canada is chosen as a case study to illustrate the applicability of the QL algorithm on short-term optimization of the hydroelectric power production. This multireservoir system is operated by Hydro-Quebec as a public utility that manages the generation, transmission, and distribution of electricity in Quebec. This multireservoir system with a power production capacity of 1,550-MW is constructed on the Romaine river basin, north of the municipality of Havre-Saint-Pierre on the north shore of the St. Lawrence as shown in Figure 4.1(b) and the corresponding view from above in Figure 4.1(a).

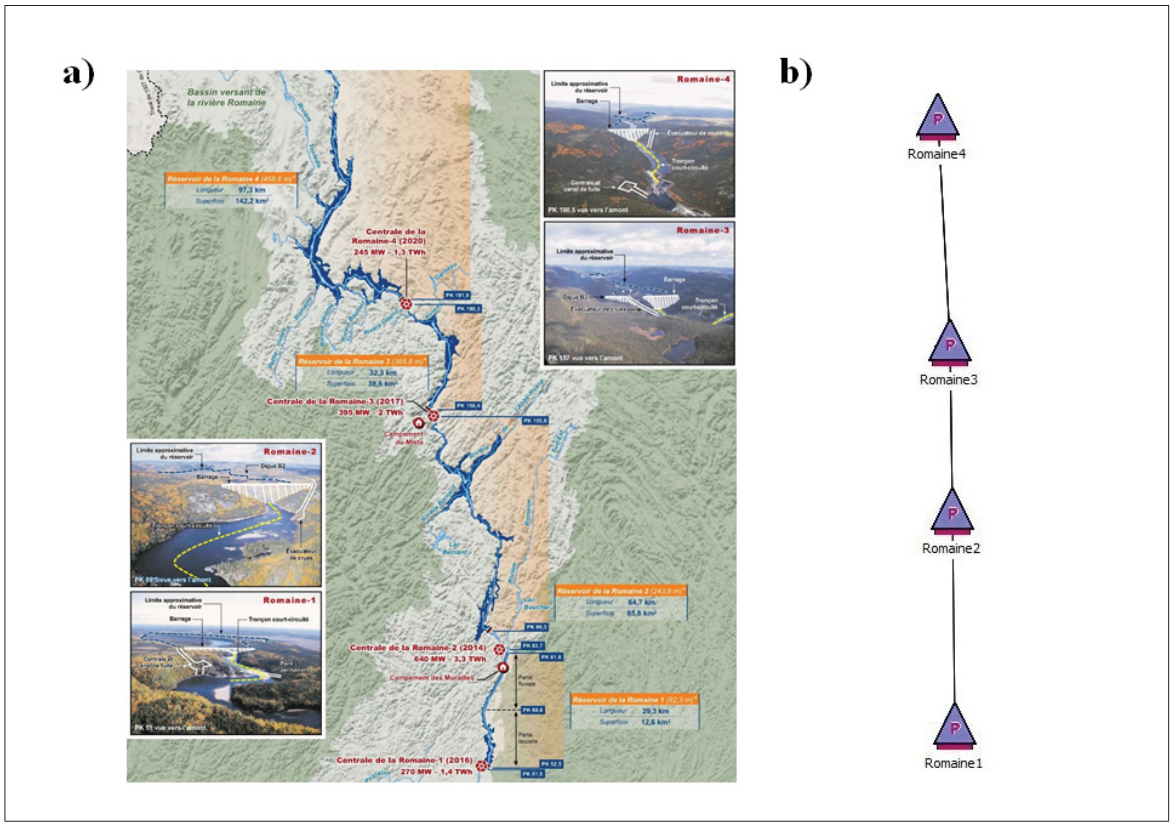


Figure 4.1 Romaine river basin site in Quebec (a) The geographical view (b) The schematic view

The complex consists of four hydropower generating stations with average annual output of 8.0 TWh. Construction of the Romaine-2 development began in 2009. Romaine-2 was commissioned in 2014, and the Romaine-1 development was commissioned in 2015. The Romaine-3 and Romaine-4 will be operational in 2017 and 2020, respectively and the developments are underway. This complex is one of the most important projects of its kind in North America. It will generate approximately $3,5 billion in economic spin offs in the province of Quebec and an average of 975 job site workers per year for 11 years. The main impact that needs to be considered is the preservation of ecological diversity and natural environment as well as social environment and local heritage. This project helps the host communities by ensuring that land users can continue their activities during and after construction.

Table 4.1 Main characteristics of each site

Parameters	Romaine-1	Romaine-2	Romaine-3	Romaine-4
Dam location (km)	52.5	90.3	158.4	191.9
Main dam height (m)	37.6	109	92	87.3
Active storage (hm^3)	18	419	475	1762
Net head (m)	62	158	119	88
Installed power (MW)	270	640	395	245
Annual output (TWh)	1.4	3.3	2	1.3
Minimum level (m)	81.8	238.8	352.8	442.1
Maximum level (m)	82.3	244.1	365.8	459.6

The key characteristics of each site are listed in Table 4.1. All this information will be used to build the simulator of multireservoir system. As it can be seen, the biggest reservoir according to the active storage area is Romaine-4, and the smallest one is Romaine-1. In this multireservoir system, the reservoirs are located in series with respect to each other. The point in this complex that the water released from upstream reservoir will arrive to next site within the similar day.

4.3 Model of environment

Based on the introduced case study, the next step is to design a simulator that can be considered as an environment of our QL algorithm to compute the system dynamic. Initially, we have used Riverware software which is a river system modeling tool. This software provides the opportunity of performing different tasks like operational policy analysis, system optimization, water right allocation, and long-term resource management. However, the integration of optimization algorithm on this software requires some major implementation challenges. As a result, we will use Matlab software to create the simulator to implement the QL algorithm.

At the first step, the water storage of each reservoir is calculated based on the continuity equation introduced by equation (3.3). In this equation, the water release from the upstream reservoir does not possess any delay in arriving in next reservoir due to the small distances between the location of each site. After computing the water storage in all reservoirs, the pool elevation of the corresponding reservoir can also be calculated by a 2D look-up table provided by industrial partner. The main step is then to compute the energy production in GWH as our reward function for QL algorithm. In the reservoir operation problem, the power production by turbine depends on the water release $a_{i,t}$ and net head $nh_{i,t}$ in the reservoir. The net head ($nh_{i,t}$) of reservoir i in time period t is the difference of pool elevation and tail water ($tw_{i,t}$) for reservoir i at time step t. The tail water level is computed by using a 2D lookup table as below

$$tw_{i,t} = G(a_{i,t}); \tag{4.1}$$

The power production can be computed by the use of 3D lookup table as follows:

$$Power_{i,t} = F(a_{i,t}, op_{i,t}); \tag{4.2}$$

Given the computed electric power in gigawatt, the electrical energy can be simply computed as:

$$e_{i,t} = \frac{Power_{i,t} \times \Delta T_{hour}}{1024} \tag{4.3}$$

Here, the coefficient ΔT_{hour} is equal to 24. The $e_{i,t}$ indicates the daily amount of electricity can be generated in TWH. In all above equations, functions F and G are the regression functions that are obtained by the use of linear interpolation method. The data used in all these steps are provided by industrial partner of this project. A validation is also performed in order to compare the created simulator and Riverware software as in Figure 4.2 under a similar simulation setup.

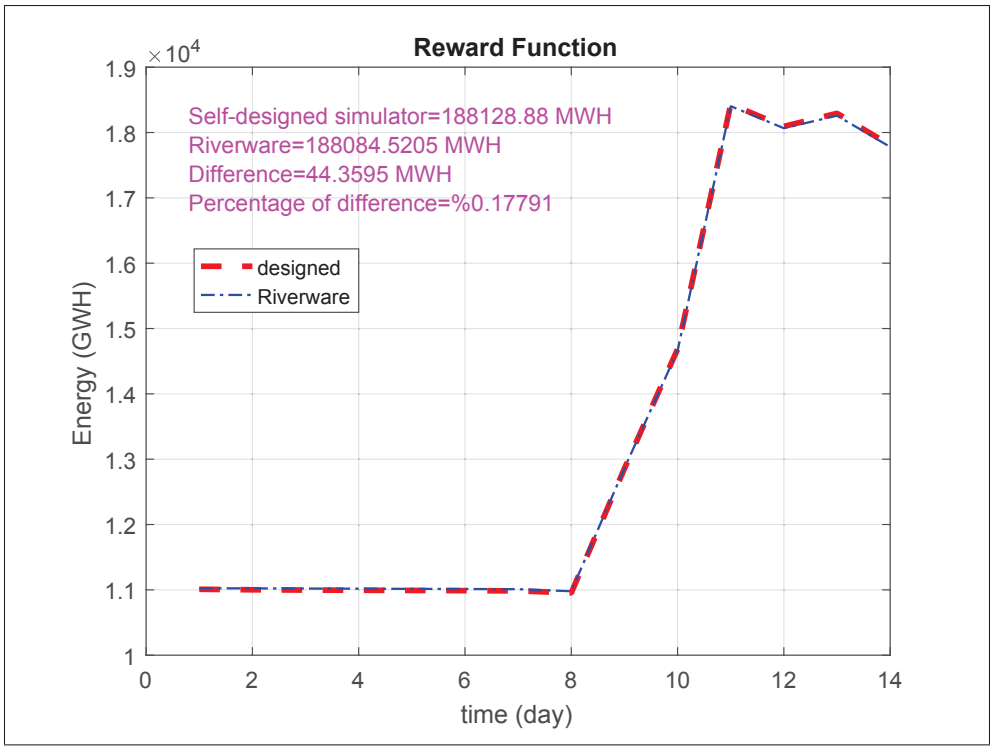


Figure 4.2 Validation of the designed simulator in Matlab Compared with the hydrological software "Riverware"

4.4 Experimental plan

In this study, the main focus is to test and analyze the basic and adapted QL algorithm introduced in the previous Chapter. Figure 4.3 shows the block diagram required to perform a simulation with the Q-learning algorithm. As it can be seen, the QL approach initially runs in training mode, where the action-value function and the optimal control policies are being trained. In this step, several parameters should be calibrated so that the best performance could be observed in the operation process applied to the optimization problem of the multireservoir system. The main parameters considered in this study are exploration rate, learning rate, and training dataset (scenario). Once the training process is completed, the algorithm will be applied to solve the actual daily operation problem using the operational subset of data that is independent of the training dataset. In this step, we will perform a complete simulation analysis for verifying the effectiveness of proposed approach on the short-term management of the multireservoir system.

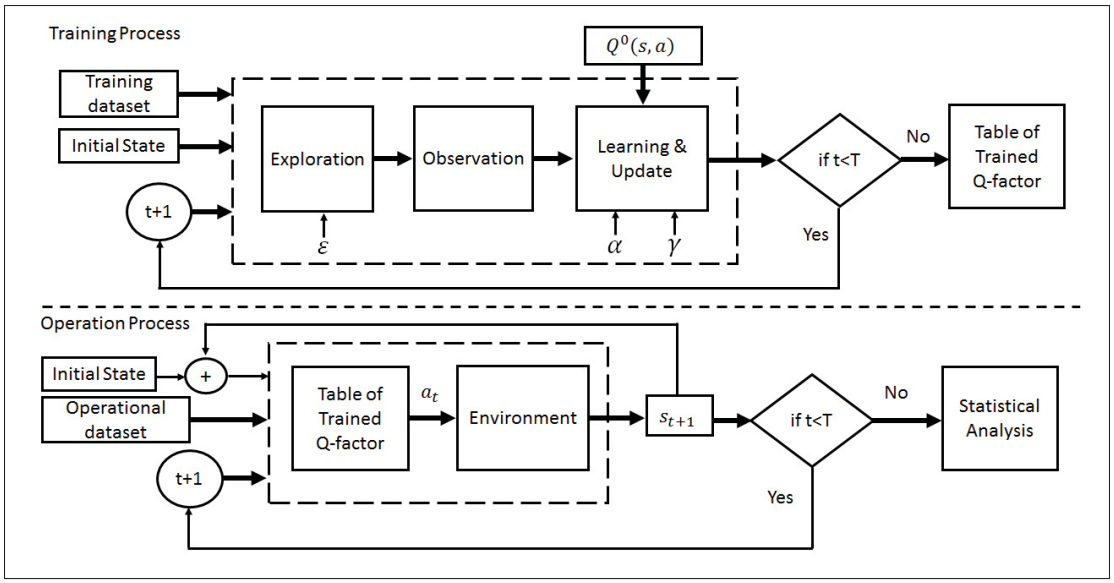


Figure 4.3 Experimental simulation scheme of Q-learning algorithm

4.5 Experimental setting

In this section, the parameters of QL model needs to be initialized to execute a basic simulation. In this project, the time horizon T is considered as two weeks, alternatively fourteen days $(T = 14)$, where a daily time step is selected accordingly. The selected case study consists of three main reservoirs and a run-of-the-river power plant so that the parameter $\mathbb{I}$ as an indication of a number of reservoirs is three. The synthetic streamflow are generated by using multivariate stochastic models such as SAMS. The cardinality of synthetic dataset is equal to 50000 contains daily water inflow (50000 training years). This means that the number of scenario N provided for QL algorithm is assigned with 50000. Also, the cardinality of historical dataset is 50 corresponds to 50 years (from 1960 to 2010).

One of the basic steps towards the implementation of lookup table QL approach is the discretization of variables. So, the grid of discretization for pool elevation, water outflow, and water inflow are chosen as $n_x = 20$, $n_u = 50$, and $n_\omega = 50$. In this project, the boundary for possible water outflow is considered zero as the least possible water release and 1000 as the maximum capacity of reservoir to release water, so the set of feasible action can be

defined by the discretization of original set $[0,1000]$. The penalty factors for violation of constraints M_{CL_i}, M_{CO_i} have been assigned with a value of 1000. The penalty coefficient for water spillage takes the value of one. The production efficiency coefficients C_i for final value of water for each plant including the run-of-the-river power plant are considered by $[C_1, C_2, C_3, C_4] = [0.155833, 0.551111, 0.848877, 0.848877]$. This coefficients are provided by industrial partner. The value of γ is also assigned with one in whole simulation of this Chapter as it obtained the best result in some experiments.

The simulation results will be presented in the following can be classified into two groups. The first group is the simulations carried out under for the specific operational year. In this case, the period of operation is 20^{th} of May up to 4^{th} of June in the year 1981. The reason for selecting this year is the result of large incoming natural inflow received by the reservoir compared to the others in the historical dataset. In addition, this period has shown the highest amount of water inflow in comparison with the other period of the year. This simulation settings is considered in the Section 4.6, Section 4.7, Section 4.8, and Section 4.9. Once the best setting has been determined for QL algorithm through these sections, some extended simulation results will be performed in the Section 4.11 in order to measure the performance of QL algorithm under different initialization of time periods and different water scenarios occurs in different years. To this end, the single operational year will be replaced by the all number of the available series in the historical dataset. Also, the operation time period will be changed by considering the autumn period.

4.6 Evaluation of adaptive step-size and ε-greedy

In this section, the evaluation of basic QL algorithm under different leaning and exploration strategies, introduced in Chapter 2, are analyzed to obtain the best parameters setting. A point here is that the choice of the rule for ε and α heavily depends on the application or case study that the algorithm will be implemented on. In the following, the simulations are executed in the period mentioned above. The training dataset was generated by extracting the inflow occurred in the exact period of above-mentioned operation over the synthetic dataset (not preprocessing).

The cardinality of training dataset is 50000. Table 4.2 represents the corresponding simulation result for comparing different strategies in QL approach.

Table 4.2 Study results towards the step-size and ε-greedy

Case	Power Production	Objective function	Water spillage
$\alpha_n = \varepsilon =$cte	357	2520	3391
$\alpha_n =$cte, $\varepsilon \neq$cte	348	2556	1913
$\alpha_n = \frac{\alpha_{n-1}}{\alpha_{n-1}+1}$, $\varepsilon =$cte	343	2556	2025
$\alpha_n = \frac{10}{10+n}$, $\varepsilon =$cte	347	2559	1805
$\alpha_n = \frac{50}{50+n}$, $\varepsilon =$cte	367	2539	2589
$\alpha_n = \frac{500}{500+n}$, $\varepsilon =$cte	341	2559	1817

As it can be seen, the adaptive ε-greedy approach improves the value of the objective function with 36 GWH compared with the constant exploration rate. Moreover, the amount of water spillage is decreased by 43% that is a much more valuable compared to the improvement in the side of objective function. As for Learning rate (α), using the function $\frac{10}{10+n}$ presents the best result comparing the other rules used in the similar function. In this setting, the water spillage is also experiencing a significant improvement with a rate of 46.7% where the objective has an increase of 39 GWH compared with the first case where the exploration and learning rate are fixed with constant value. The rest of simulations carry out in this thesis will be performed with the best parameter setting obtained in this section.

4.7 Pre-processing the training dataset

In this section, the objective is to analyze the effectiveness of proposed algorithm for selecting the sample scenarios of water inflow to train Q-factors. In this way, two datasets: historical, and synthetic dataset are considered. In this study, the synthetic dataset is generated by hydrological software called SAMS as it is shown in Figure 4.4.

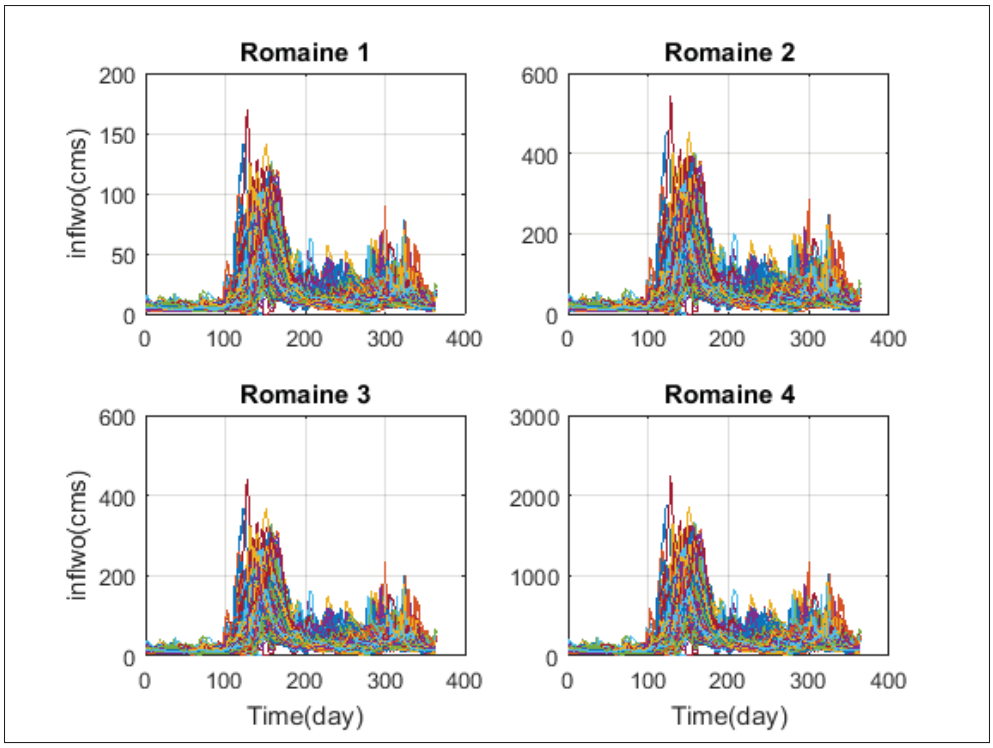


Figure 4.4 Daily water inflows generated by SAMS for a complete year

As it was introduced in the section 3.3.4, two simple strategies could be considered for selecting the training dataset. The first idea is to consider all the index of water inflow occurred in the period of operation and the other is to use the Threshold Strategy to select the most correlated sequence of water inflow. To verify the performance of the generated synthetic dataset and the proposed preprocessing strategy, the following case of experiments are executed.

1) Repeated historical series (R-HS)

2) Repeated pre-processed historical series (RP-HS)

3) Synthetic series (SS)

4) Pre-processed synthetic series (P-SS)

Here, the input dataset for the first two experiments (R-HS and RP-HS) is historical data series where they are treated with/without preprocessing strategy. In these experiments, the number of

scenarios are different due to the difference between initial dataset and preprocessing afterward. So, it is not possible to have a fair comparison between all cases at the end. To overcome this issue, a simple strategy is to repeat the dataset *m* number of times so that there could be an almost the same number of scenario. In the second group of experiments (SS and P-SS), the synthetic data is used as the initial dataset to train the Q-factor as well as being modified with the Threshold Strategy where the value of boundary (δ) is 5%.

Table 4.3 Comparison of different training datasets on single operation scenario

Method	**Scenario index**	**Objective function**	**Water spillage**	**Mean outflow**
R-HS	50041 (m=893)	2544	1348	290
RP-HS	50091 (m=263)	2548	1116.6	288
SS	50000 (m=1)	2563	1575.4	370.3
P-SS ★	49512 (m=1)	2587	753.06	328

The simulation results for cases mentioned above are provided in Table 4.3. In the first two experiment, the number of the dataset was repeated 893 and 263 number of times accordingly, as the original dataset is small to reach a convergence on training the Q-factor. On the last two simulations, the historical series of water inflow is replaced by the synthetic dataset. As there is a no limitation on the side of generating an infinite number of scenarios, we skipped the need to repeat the initial dataset over the last two simulations. Over these experiments, it can be found out that the preprocessing strategy is quite useful either using the historical series or the synthetic dataset. In the last simulation, the objective function is improved by 24 GWH compared with the first simulation where there is no modification on the training dataset. Furthermore, the water spillage is decreased with the rate of 52.1%.

Table 4.4 illustrates a comprehensive results towards the impact of different training datasets over several operational scenarios than a single one. In this results, 40 series of water inflows are sampled from the historical dataset for operation step. The criteria of this sampling is to take an average value of $\bar{\omega}_0$ of all years in the historical dataset, then consider any indexes over

Table 4.4 Extended analysis toward the impact of different training datasets

Parameters	**R-HS**	**PR-HS**	**SS**	**P-SS**
Scenario index	50041 (m=893)	50091 (m=263)	50000 (m=1)	49512 (m=1)
Mean(O.f.)	1860	1834.4	1867	1872
σ(O.f.)	233	245	311	309
Max(O.f.)	2554.7	2596.8	2770	2782
Min(O.f.)	1431.3	1434	1373	1417
Mean(spillage)	802	1243	1636	1574
Mean(outflow)	213	325	367	334

the whole year which has the same value of $\bar{\omega}_0$. One point in this simulation is that the training dataset is somehow correlated with the operation dataset for the first two experiments (R-HS and PR-HS). So, it is not possible to have a fair comparison about the impact of preprocessing strategy. However, looking at the last two results (SS and P-SS), it can be observed that preprocessing strategy can increase the performance of QL algorithm in term of objective function. This comparison is fair since the initial dataset is not correlated with the operation dataset.

4.8 Aggregated Q-learning

In the previous sections of this Chapter, we were concentrating on appropriately obtaining the best parameters for lookup table version of QL algorithm like learning rate, exploration rule, training data set, etc. In spite of improvements provided in the previous sections, the daily computed policies were experiencing a noticeable trend of fluctuation over the water outflow of reservoirs. This may not be favorable in the real-time operation as it can bring an operational cost for changing the turbine release on every day. One possible reason for the observed fluctuations associates with curse of dimensionality in the multireservoir operation problem. This will affect the performance of QL approach as it needs to reveal many Q-factors for many state and action pair. In the literature, several advance techniques are presented for approximating Q-factors so that the impact of dimensionality can be eliminated. However,

we present a simple modification in the structure of QL algorithm applied to the short-term management of hydropower system.

The idea, here, is to renew policies once the training of Q-factor for daily operation is completed. In this case, the Q-factor estimated for a state and action pair on a specific day can be considered for some days before or after of that day due to the short-term operational properties. In the proposed methodology called "$QL(weekly)$", once a policy for a specific state is observed on a day of a week, that policy could be valid for the particular state independent on the day of operation over the first week. To clarify more, if a particular state has been visited just on the first day of operation, the same policy can be considered for other days of the same week. Meanwhile, another hypothesis is made for a type of states that have been visited within several days of a week. The idea here is to consider the average policies over the week since the random variables have a stationary behavior over the weekly time step. The basic impact of this assumption is the increase in a number of Q-factors stored in the Q-table. As a result, it will make the algorithm to be more stable and reliable in term of performance.

In Figure 4.5, the simulation result for comparing the original QL algorithm and aggregated QL into weekly time step is provided. The simulation period is selected similarly as it was mentioned in the section 4.5. The exploration and learning rate is assigned with the best value obtained in section 4.6 as well as pre-processing the training data set. The simulation results show that the aggregated QL approach presents better performance through different aspects. The first point is the elimination of fluctuations of water outflow observed in the daily QL algorithm. As it was mentioned, this is much more promising since it will result in a reduction of operational cost for changing the turbine release on each day for the industrial platform. On the other hand, the water spillage reduced significantly in the newly proposed structure where it is close to zero. This indicates that as we explore more the environment (more Q-factors are known), more reward can be obtained by finding a more close solution to the optimal one. Another point is the possible improvement of objective function where its value improved by 15 GWH over the time period of two weeks. Also, the minimum water outflow on Romaine-1 is satisfied as the water release computed is above the line in dash blue. A minor hint for

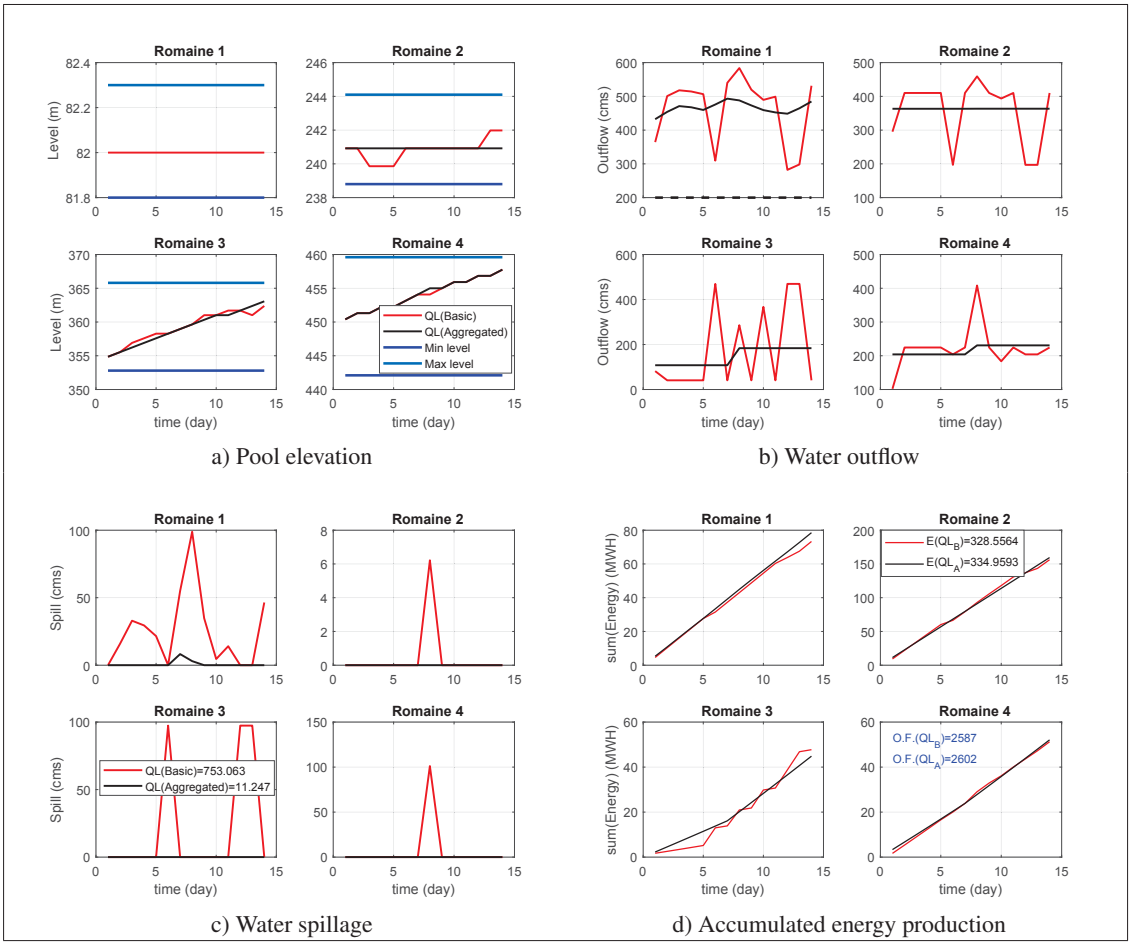


Figure 4.5 Comparison of basic and aggregated Q-learning algorithm

Figure 4.5a could be considered on the elimination of the line for "$QL(weekly)$" approach. As it was mentioned before, Romaine-1 is a run-of-the-river power plant where it does not have a capacity to store water. This point results to have a constant level during the operation step. So, the lines plotted in one strategy will be overlapped over the other one.

4.9 Q-learning with linear Q-factor approximation

As it was mentioned before, the Q-factors could be approximated by the linear combination of the feature vector. In this section, two type of simulation results are presented based on the choice of feature vector on the training step of QL approach. The first result is provided with a linear combination of state and action vectors called as "QL_L". The second version of the

result is executed by considering the same feature vector plus the water spillage with a positive coefficient called as "$QL_{L,S}$". In the $QL_{L,S}$, the optimization model will try to push the system to reduce the water spillage (considering the penalty factor is one). Hence, we can compare its consequences on the power production and objective function. Meanwhile, they will also be compared with "QL_{Lookup}" which is the results obtained by lookup table version of QL algorithm with best exploration and learning rate as well as pre-processed training dataset.

As for implementation of linear Q-factor approximation, the general simulation process is similar to the lookup table version of QL approach where it contains a training and operation step. In the training process, the exploitation step of approximated QL algorithm requires to optimize a nonlinear function since the reward function is nonlinear. In this step, the nonlinear optimization problem is solved by the interior point algorithm, provided in Matlab Optimization Toolbox. Using the obtained coefficients for linear approximated Q-factors, the water outflow for operational step is computed by solving a constraint linear programming problem. This optimization problem is also carried out by using Matlab Optimization Toolbox. The simulation setup mentioned in the section 4.5 is also considered to execute a simulation for approximated QL algorithm.

Figure 4.6 illustrates the simulation results where the approximated QL approach is compared with the lookup table version. As it can be seen, the combination of state and action as a feature vector scores badly in term of objective function comparing with the designed lookup table version. A possible reason is the fact that Q-factors will be under-fitted if a linear function is being used for their approximation. Analyzing the results obtained for " QL_L", the water outflow computed in operational step fluctuated between the minimum and maximum value of outflow. The reason here is that the solution of linear programming problem is always the extreme points of a feasible region. In our problem, since Q-factor is represented with a linear function, the solution of optimizing this linear function will always be the maximum or minimum of the feasible domain. Considering this point, the solution obtained by this feature vector is either zero or thousand for water outflow. This has also resulted in a significant increase in the amount of water spillage.

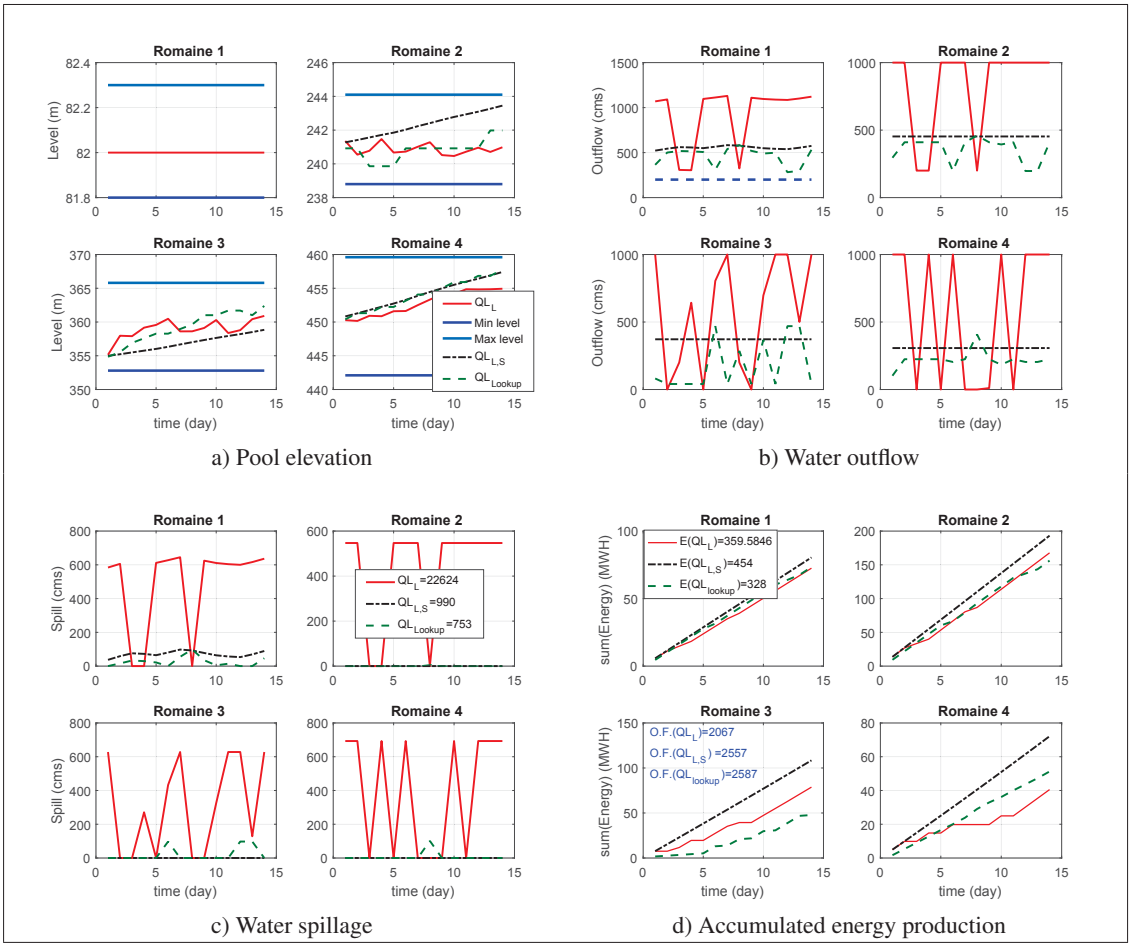


Figure 4.6 Comparison of approximated and lookup table QL algorithm

Analyzing the results provided for "$QL_{L,S}$", the inclusion of water spillage to the feature vector can be promising where the objective function could be improved. Obviously, avoiding water spillage could be beneficial for operating the multireservoir system. However, there is no guarantee that low spillage operation is the optimal solution of the main problem. Based on the Figure 4.6c, one can see that the water spillage is almost close to the zero in Romaine-2, Romaine-3 and Romaine4 as the spillage of these reservoirs are included in feature vector with a negative penalty. This means that even by using a higher penalty coefficient, an almost similar result can be obtained for this case. Exclusion of Romaine-1 is because of the fact that it is a run-of-the-river power plant (not a reservoir).

Table 4.5 Study results towards the approximation of the Q-learning algorithm

Method	Power generation	Objective function	spillage	CL	CO
QL_{Lookup}	328	2587	753	Satisfied	Satisfied
QL_L	359	2067	22624	Satisfied	Satisfied
$QL_{L,S}$	454	2557	990	Satisfied	Satisfied

Table 4.5 summarized the numerical simulation results obtained for the approximated and lookup table QL approach. In the QL_L, the objective function scores 20% less in comparison with lookup table approach. Nonetheless, adding the water spillage with a negative term could help the linear function to improve significantly in term of the objective function and the water spillage, it still cannot beat the solution obtained by lookup table version. This shows that a linear function with the introduced features is not the right candidate for approximation of Q-factors. To this end, this approach will not be studied further in the later sections.

4.10 Experimental setting (II)

In the primary simulations, we have analyzed the performance of QL algorithm considering the experimental setting described in the section 4.5 where the operation period was one particular period of a single year. To perform a solid statistical analysis of the QL approach, two simulations will be presented in this section. The first one is to verify the performance of QL algorithm over all the scenarios existed in historical series. Furthermore, another simulation will be performed on another period of the year to see how designed QL algorithm will act under different characteristics of hydrological water inflow occurred in the different period of the year. The main reason to select these periods is to cover the highest pick observed in the historical series. Although, it is also possible to perform a simulation in any period of year as it has been integrated in the designed software package. In these simulations, a similar algorithm's parameters are used as mentioned in the section 4.5 like step size rule, ε-greedy, initialization of Q-factors, and discretization of variables. As for operational dataset, the historical series of

daily water inflow observed from 1960 to 2010 are considered. Meanwhile, the training dataset will be generated by using the synthetic one.

The basic point in performing the experiments is the way that we will integrate the preprocessing strategy in the training step. As it was mentioned before, the preprocessing algorithm tries to extract the dataset that preserves a similar short-term dynamics with respect to the initial value of water inflow $\bar{\omega}_0$ of a specific year. However, the new simulation dataset contains the water inflow for several years, consequently, different values of the $\bar{\omega}_0$ are available for each year. So the question here is that which $\bar{\omega}_0$ should be selected for preprocessing strategy to generate the training dataset. One possible solution is to consider all the values of $\bar{\omega}_0$ by augmenting the $\bar{\omega}_0$ in the state variable of Q-factors as $(\bar{\omega}_0, \bar{\omega}_{t-1}, x_{i,t})$.

In each experiment, three sets of simulation results will be presented namely QL, QL(weekly), and DDP. The first results are associated with the solution computed by basic lookup table QL algorithm where the parameters are optimized and the training dataset is preprocessed. The second results are obtained by using the methodology described in section 4.8 where Q-factors are aggregated in a week. Finally, the results called "DDP" is the solution obtained by deterministic dynamic programming. The result computed by DDP can also be viewed as the perfect solution of optimization problem since the optimization performed by knowing the information of water inflow in advance. In DDP, we used the deterministic version of equation (1.1) where the value of uncertain variable ω_t is fixed with the corresponding historical series. The experimental setting used for DDP is similar to the QL approach like the discretization of variables and the final value of water to have a fair comparison between two approaches.

A key point in the operation process of each simulation is the way that we try to satisfy the constraints rather than violating them. Once we are operating the multireservoir system, sometimes we will be in a state that has never been visited in the training process. This means that there is no estimation of Q-factor in that specific state. If this situation happens, a possible solution is to return zero for water outflow. However, this will violate the minimum water outflow constraint. To avoid this violation, the zero solution will replace by the minimum re-

quired value of water outflow if there is enough water in the reservoir. If there is no water in the reservoir to release, then we have to violate this constraint and the cost function will be penalized accordingly. For the constraint on pool elevation, the water level at day t is being checked by using the water outflow computed at day t and water inflow observed at day $t-1$. Based on this estimation, a modification will be applied to the water release if the constraints on the water level are going to be violated.

4.11 Algorithm Extension

Considering the new experimental setting explained in the section 4.10, we are going to present and compare the performance of previously mentioned three approaches namely: QL, QL(weekly), and DDP. As we mentioned in the previous section, the purpose of performing these experiments is to deeply analyze how the designed approach can behave under different conditions especially different years of operation (first scenario) and different periods of operation (second scenario). As for the first scenario, we will present the extended simulation results over the generalization on operation year. Figure 4.7, 4.8 show the evolution of main reservoir's dynamics as pool elevation, water spillage, water outflow, and energy production in the period of 20^{th} of May up to 4^{th} of June. In these simulations, the initial value of pool elevation is fixed and it is provided by the industrial partner of this project.

In Figure 4.7c and Figure 4.7d, a noticeable fluctuation is observed in the water release policy computed by QL algorithm compared to the QL(weekly) algorithm. As we mentioned in the section 4.8, the QL(weekly) approach has a more estimated value of Q-factors stored in the Q-table as a result of the aggregation over a week. This will result in having more information from the environment once the algorithm tries to find an optimal policy. Another positive point can be considered on the successful satisfaction of constraints in either pool elevation and minimum water outflow. In Figure 4.7a and Figure 4.9b, there is an increasing trend in the pool elevation of the Romaine-3 and Romaine-4. As it was mentioned before, Romaine-1 is not a reservoir so that it does not possess a usable storage. This will be a problem once the Romaine-1 receive smaller water inflow than the minimum required water outflow to release.

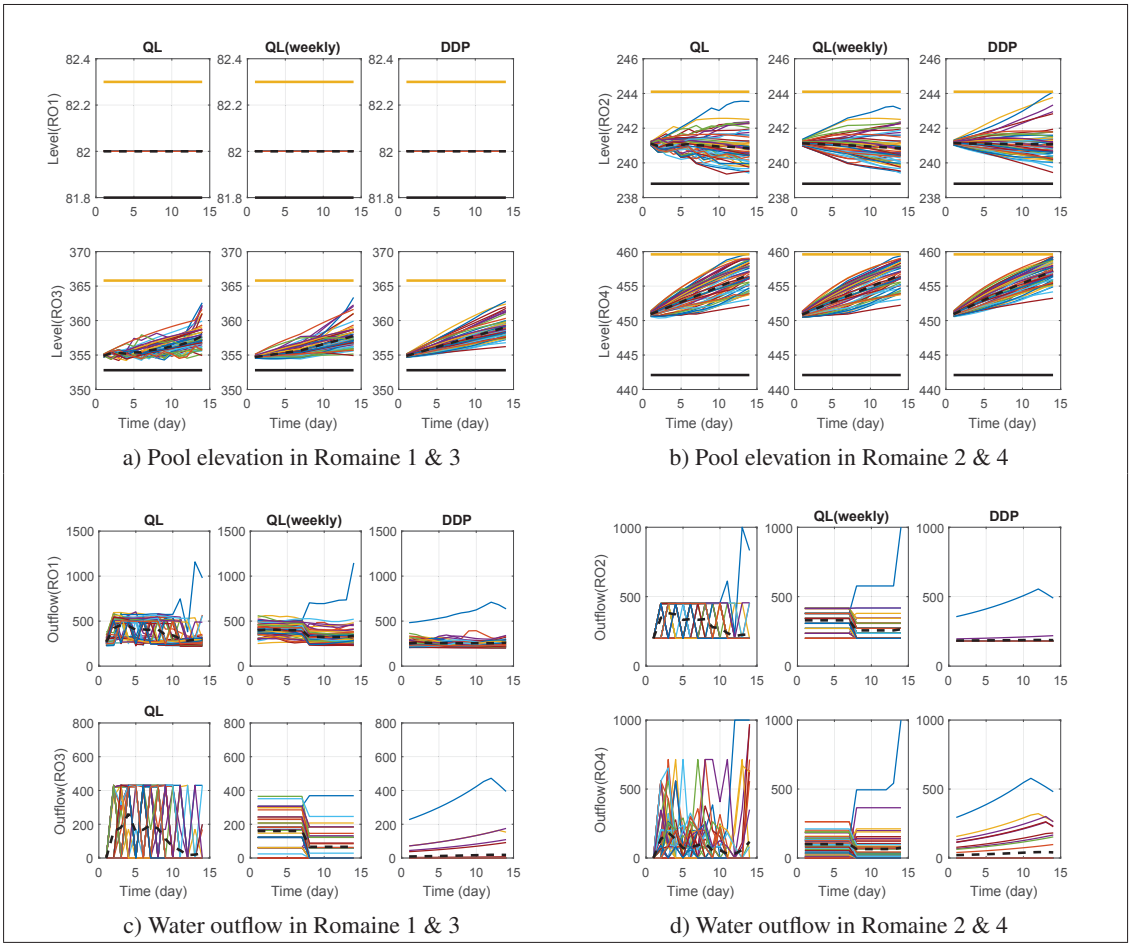
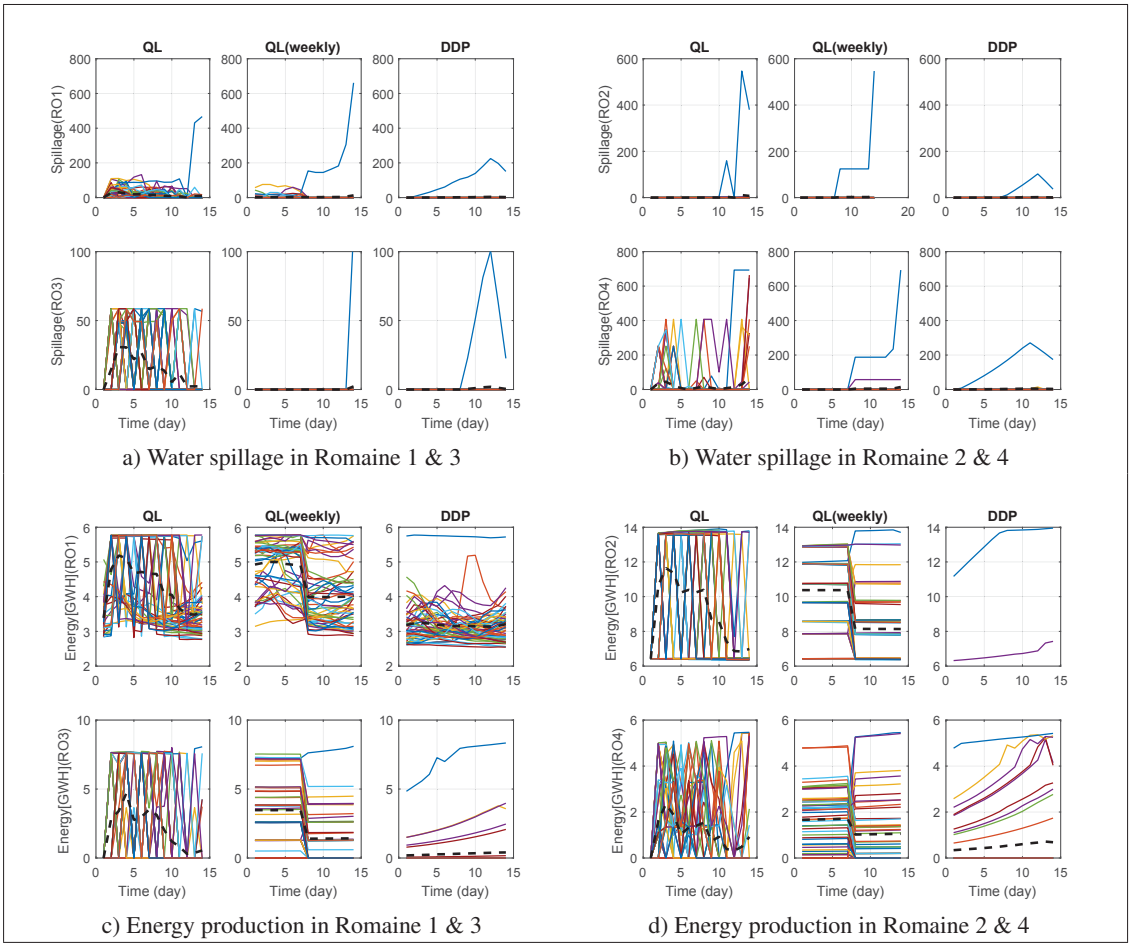


Figure 4.7 Extended simulation result (I) in the period of 20^{th} of May to 4^{th} of June from 1960 to 2010

In this situation, the rest of water required to satisfy the constraint should be provided by the upstream reservoir. As it can be seen from Figure 4.9b, the pool elevation of the Romaine-2 is increasing constantly compared with Romaine-3 and Romaine-4 which is the result of satisfying the minimum water outflow on Romaine-1. In Figure 4.7a and Figure 4.9b, lines in black and orange correspond to the minimum and maximum level of each reservoir, respectively.

Figure 4.8a and Figure 4.8b show the water spillage computed for three approaches as well. It can be easily seen that the water spillage of QL(weekly) is lower than the QL approach. This is another advantage of having more estimation of Q-factors stored in Q-tables. Aside from these simulation results, we are also interested in evaluating the impact of integrating a



a) Water spillage in Romaine 1 & 3

b) Water spillage in Romaine 2 & 4

c) Energy production in Romaine 1 & 3

d) Energy production in Romaine 2 & 4

Figure 4.8 Extended simulation result (II) in the period of 20^{th} of May to 4^{th} of June from 1960 to 2010

different form of the cost function in the process of training in the QL algorithm. In this case, the objective function is modified by integrating the water spillage with a negative coefficient so that it expected to see a reduction in the value of spill.

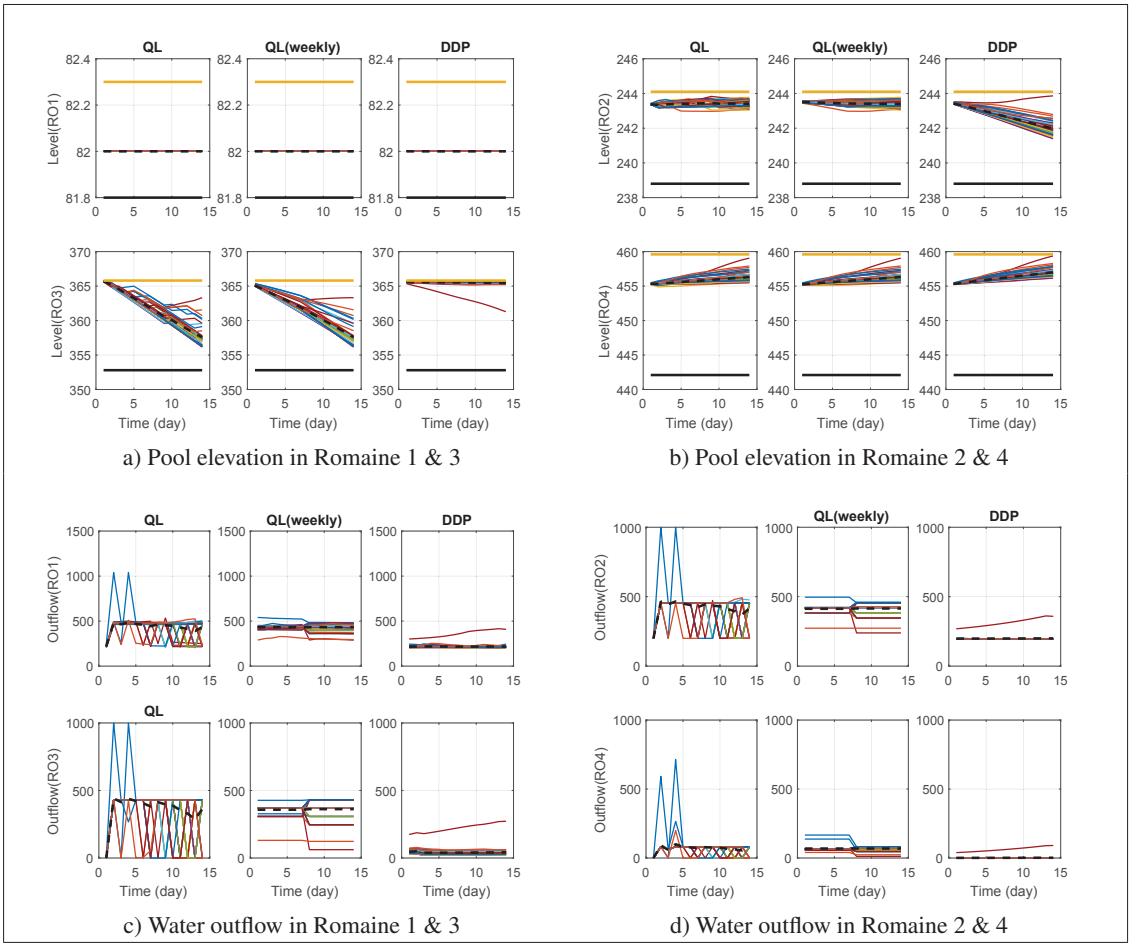
Table 4.6 summarized the numerical results of the experiments performed above plus two additional simulations. These two simulations called $\text{QL}_{daily,spillage}$ and $\text{QL}_{weekly,spillage}$ are a similar version of QL and QL(weekly) approach, respectively, where the cost function is penalized with water spillage. Analyzing the information, it can be observed that integration of water spillage in the cost function results in having less loss of water, while the objective function does not experience a noticeable improvement. According to the results, the QL(weekly)

approach without water spillage presents the best results with 2092 GWH. The value reported as an objective function is the summation of the energy production on all time period and the value of water at the end of horizon. Compared to the DDP, the QL(weekly) scores close in term of objective function, and this shows that the QL approach has been implemented efficiently on this problem. One can also see that the $QL_{daily,spillage}$ and DDP approach scores close on objective function while $QL_{daily,spillage}$ spills more. The potential reason is the water spillage does not have so much impact on the short-term operation of the multireservoir system due to the structure of the system and statistical properties of natural water inflow. It is important to note that the values of energy production, objective function (O. f.) and water spillage in Table 4.6 are reported by the mean value.

Table 4.6 Comparison of Q-learning algorithm with surrogate form of objective function in the period of spring

Method	**Energy**	**O. f.**	**Min-outflow**	**Level-violation**	**Water spillage**
QL_{daily}	224.3	2076	Satisfied	Satisfied	1207
QL_{weekly}	236.4	2091	Satisfied	Satisfied	612.4
$QL_{daily,spillage}$	232.2	2077	Satisfied	Satisfied	1040.2
$QL_{weekly,spillage}$	246.2	2090	Satisfied	Satisfied	573.24
DDP	139.4	2106	Satisfied	Satisfied	79.17

The second simulation scenario is to assume another period of the operation in the year. Here, the new period is selected from 15^{th} of October to 4^{th} of November, assuming the experimental setting explain in the section 4.10. The reason to select this period is the high incoming water inflow to the reservoir observed in the historical series. The challenge of operating in this period can be considered in the high level of water in reservoirs, so, even a medium size water inflow could increase the risk of flood. Figure 4.9, 4.10 shows the simulation results carried out during the period of autumn.



a) Pool elevation in Romaine 1 & 3

b) Pool elevation in Romaine 2 & 4

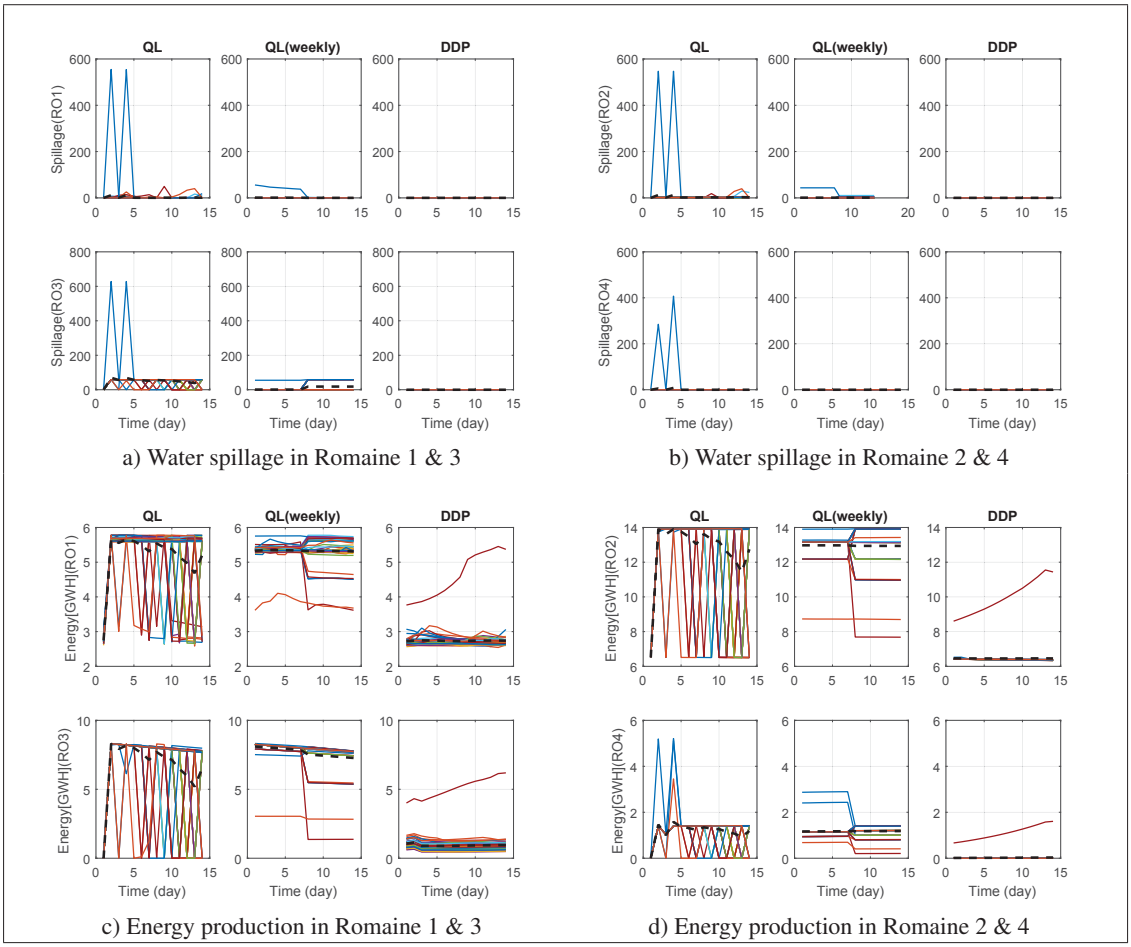
c) Water outflow in Romaine 1 & 3

d) Water outflow in Romaine 2 & 4

Figure 4.9 Extended simulation result (I) in the period of 15^{th} of October till 4^{th} of November from 1960 to 2010

Figure 4.9a and Figure 4.9b illustrate the result for the pool elevation of all reservoirs during the operation step. The first positive point is that all the water level of all reservoirs are in the boundary, so all the constraints on water level has been satisfied. In Figure 4.9b, there is a trend of emptying the Romaine-3 reservoir's storage. This pattern is mostly because of satisfying the constraint on minimum water outflow of Romaine-1 than violating the constraint. The QL(weekly) approach tries to keep the highest level of water in all reservoirs by smoothing the water release policy. As a result, the QL(weekly) approach still performs well in comparison with the QL algorithm as well as being close to the perfect solution provided by DDP. Figure 4.9c and Figure 4.9d show the computed release policies on all reservoirs. As it can be seen, the QL approach possesses fewer fluctuations in the water outflow policies compared

to the simulation scenario performed in the spring. Moreover, the QL(weekly) successfully removes most of the fluctuations from water release policies.



a) Water spillage in Romaine 1 & 3

b) Water spillage in Romaine 2 & 4

c) Energy production in Romaine 1 & 3

d) Energy production in Romaine 2 & 4

Figure 4.10 Extended simulation result (II) in the period of 15^{th} of October till 4^{th} of November from 1960 to 2010

Figure 4.10a and Figure 4.10b show the trend of water spillage computed during the operation step. It can be observed that QL(weekly) avoids spilling the water in a good quantity. Table 4.7 summarizes the simulation results obtained in the autumn period. A remarkable point in these set of simulations is the impact of penalizing the water spillage in the cost function. It can be observed that the water spillage is reduced significantly by the QL(weekly), while integrating the water spillage in the cost function of the QL(weekly,spillage) does not result in an improvement in either water spillage or objective function. One possible reason could be

considered in the structure of the selected multireservoir system. In the selected case study, the reservoirs are located in series, so the water spillage of the upstream reservoir will be the water inflow for downstream. Considering this point, avoiding the water spillage does not always result in an optimal behavior of the multireservoir operation problem. Furthermore, the natural water inflow will also have an impact on the optimal operation of the multireservoir system. Comparing the result, QL(weekly) possesses the highest score of the objective function with the value of 2287 GWH.

Table 4.7 Comparison of the Q-learning algorithm with surrogate form of objective function simulated in the autumn

Method	**Energy**	**O. f.**	**Min-outflow**	**Level-violation**	**Water spillage**
QL_{daily}	359.4	2266	Satisfied	Satisfied	793.1
QL_{weekly}	380	2287	Satisfied	Satisfied	155.3
$QL_{daily,spillage}$	359.5	2263	Satisfied	Satisfied	683.5
$QL_{weekly,spillage}$	380.5	2285	Satisfied	Satisfied	145.3
DDP	142.3	2325	Satisfied	Satisfied	0

A the end of this section, a complete comparison between the best solutions obtained in previous experiments over two period of spring and autumn are compared with the solution of DDP approach. Figure 4.11 illustrates the simulation results as a comparison between DDP and QL(weekly) approach. The horizontal axis called scenario is the year of operation on which DDP and QL(weekly) approaches are applied to solve the multireservoir operation problem. In addition, three variables are depicted as score of objective function (O.f.), energy production, and the water spillage. As it can be seen, the objective function scores close in the simulation carried out in the spring. However, the results obtained for autumn period is less variant compared with the solution obtained by DDP and the mean value of objective function is not as close as the spring period. The possible reason can be considered on the statistical properties of water inflow in specific period of year plus the impact of initial value of water level. The energy shown in the second row is value of power production computed in any year of opera-

tion. Similar to the trend of objective function, It can be observed that there is a comparable gap between the result obtained in the autumn period compared with the spring one.

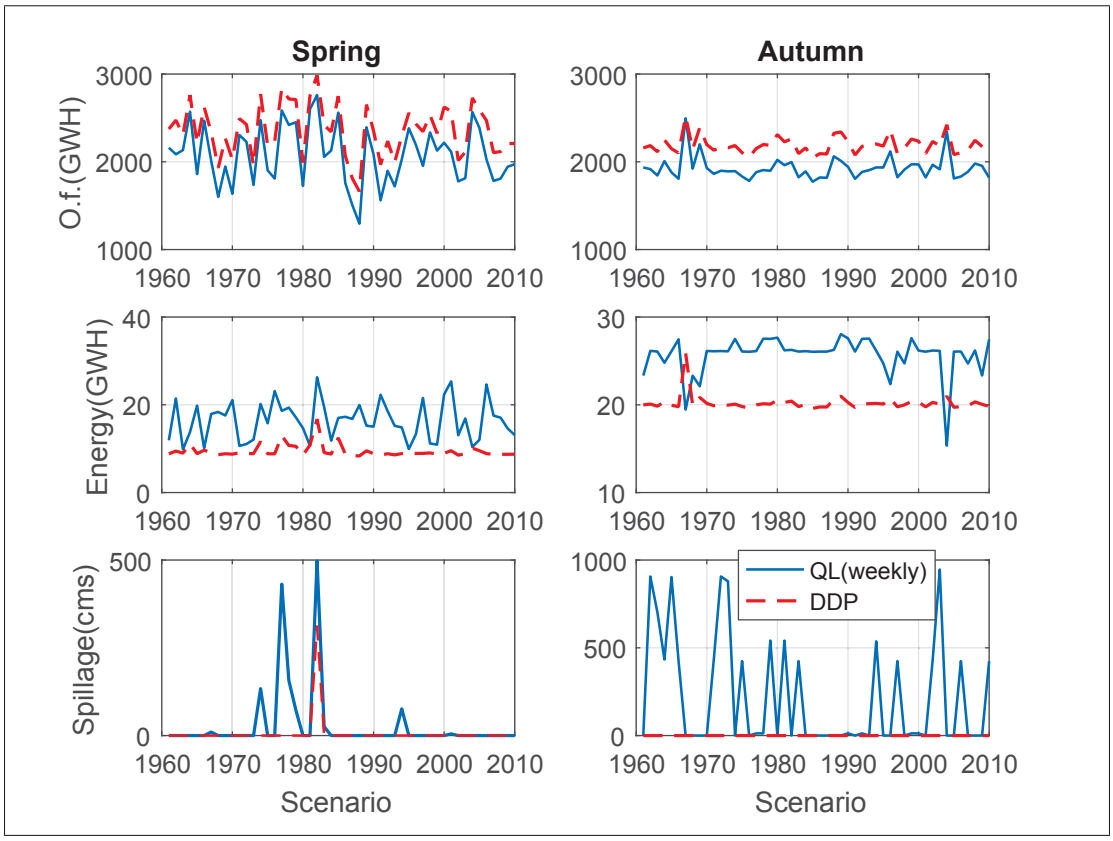


Figure 4.11 Comparison of Q-learning approach and deterministic dynamic programming (perfect solution).

Towards the end of this section, the last point is the computational cost of designed QL approach. In the presented lookup table version, the training time of QL algorithm with 30000 scenarios is approximately 8 minutes. This time running time will increase almost linearly with respect to the number of scenario.

4.12 Sensitivity Analysis

In this section, the QL algorithm will be analyzed more by scaling down the value of the water at the end of horizon through our optimization process. This can also be interpreted as the

sensitivity analysis of QL algorithm with respect to different initializations of Q-factor. Recalling from section 3.3.2, the final value of water is approximated based on the water storage of each reservoir, given the amount of energy can be produced based on the specific storage. The function has been written as:

$$\begin{aligned} V(x_{i,t}) &= H(s_{i,t}, u_{i,t}^{turbine}) \\ &\approx C_i \times s_{i,t} \end{aligned} \tag{4.4}$$

The above function can be scaled down by reformulating the function as below:

$$V'(x_{i,t}) \approx \frac{C_i \times s_{i,t}}{100} \tag{4.5}$$

Considering the new function for the value of the water at the end of horizon, we redo the simulations for Aggregated QL and DDP in the period of autumn. This will show that how the newly tested QL algorithm (QL') will perform with respect to the new perfect solution (DDP'). Table 4.8 summarizes the simulation results obtained for sensitivity analysis in the autumn period. It can be observed that the QL algorithm scores %1.63 less comparing with the solution obtained by DDP where we used the original function for the value of water at the end of horizon. Considering the QL algorithm with scaled down function for final value of the water, the objective function scores %1.51 less which is almost similar trend by considering the algorithm optimization with the original function. The increase of the water spillage in the new experiments is the result of decreasing the value of water in the reservoir.

4.13 Summary

In this Chapter, the QL algorithm was implemented on a large-scale multireservoir operation problem where the objective is to maximize the power production over the short time horizon. In the first step, a simulator was designed to compute the reservoir dynamics especially the power production value. Using the designed simulator, several simulations were carried out

Table 4.8 Comparison of the Q-learning algorithm with under different value of the water at the end of horizon

Method	**Energy**	**O. f.**	**Min-outflow**	**Level-violation**	**Water spillage**
QL_{weekly}	380	2287	Satisfied	Satisfied	155.3
QL'_{weekly}	392.2	2277	Satisfied	Satisfied	423.3
DDP	142.3	2325	Satisfied	Satisfied	0
DDP$'$	250.4	2312	Satisfied	Satisfied	11.4

to verify the performance of QL approach. The simulation results showed that the answers provided for research questions in the Chapter 3 were promising and indicated the potential capability of the QL algorithm to overcome a large-scale multireservoir operation planning problem. One of the interesting improvement was the impact of intelligently selecting the training dataset on the performance of basic QL algorithm. The second key fact is the further improvement obtained by renewing daily optimized policies by a heuristic approach called as aggregated QL. From these two simulations, it can be concluded that the QL can be improved significantly where some basic details could be selected appropriately. At the last step, a linear function approximation technique was integrated to approximate the Q-value function. The use of this function approximation technique resulted in handling a larger state space problem and gave the flexibility to solve the optimization problem at hand. In spite of its simplicity, the result illustrated that a linear function is not the right candidate to preserve the feature of the problem, hence present an appropriate approximation.

CONCLUSION AND RECOMMENDATIONS

The study of large-scale water basin system is often so complicated to be handled as a result of the uncertainty (water inflow). Reinforcement learning (RL) in particular Q-learning approach is presented to solve the stochastic optimization problem of operating multireservoir systems. Based on the literature, the stochastic dynamic programming is a powerful approach in dealing with nonlinear, and stochastic reservoirs optimization problems. However, this approach has been limited in term of performance as it requires a precise information for transition probability matrix (curse of modeling) and the solution for huge number of state variable (curse of dimensionality). The purpose of this study is to provide an optimization package by the use of QL algorithm for reliable and realistic management of multireservoir system so that it can increase a potential aspect of the practical implementation. In summary, QL can present two major advantages in solving the large-scale multireservoir control problem:

1) QL can handle the curse of dimensionality. It uses forward recursion instead of backward recursion used by DP and SDP to solve the Bellman equation. This will avoid the demand for solving the problem in all possible realization of system state variable, therefore, the computational time required for solving the problem is being reduced as a result of less number of stages. Furthermore, there could be a possibility in the use of function approximation methods in order to decrease more the curse of dimensionality.

2) QL can eliminate the curse of modeling. It does not required to know either the exact transition functions or the transition probability matrix of the system. This will result in the capability of solving the MDP without knowing the exact model of the system.

This thesis has adapted the QL approach to tackle a stochastic large-scale short-term multireservoir operation problem. The main objective was to enhance operating rules so that the

maximum expected value of energy production could be achieved under the presence of random water inflows to the reservoirs.

The overall conclusions made by this study can be briefly outlined in the followings:

- The QL algorithm was successfully used to solve a large-scale multireservoir operation problem. The designed optimization package was able to run for a set of operating years under a specific time horizon (14 days for the selected case study);
- The proposed model was implemented and applied to Hydro-Quebec multireservoir complex on the Romaine river basin, north of the municipality of Havre-Saint-Pierre on the north shore of the St. Lawrence;
- The designed QL model does not require either predetermined transition probabilities of inflow or post-procedure to obtain the operational rules;
- The parameters of QL algorithm ware analyzed and calibrated based on their performance;
- A better selection criteria was provided to generate the training dataset where the results showed that it can outperform the conventional strategy;
- The simulation results demonstrated that the model was able to provide appropriate answers to the short-term planning problem including What are the optimal system control decisions in term of water release? What is the effective objective function to maximize the hydroelectric power production?
- A linear function is not the right candidate to approximate the Q-factors since it is not able to preserve all features should be provided by the original function of Q-factors.

Future Work

There are a couple of ways and directions in which the methodologies, improved in this thesis, could be further improved and expanded. The following is a possible list of future related research direction that could be carried out to extend and improve this work:

- Investigate the use of pattern recognition to select more correlated training dataset;
- Use parallel processing for running the model to speed up the learning process and to decrease CPU time;
- Investigate other RL techniques including Temporal difference as well as newly introduced deep learning algorithm that could potentially integrate the planning and learning process;
- Establish an appropriate function approximation approach to have an accurate estimation of the value function. Neural networks are a possible alternative to the currently employed linear function;
- Provide an in-detailed model of the system including regional transmission limits, electricity demands, market price; so that more realistic solution can be obtained for the system.

BIBLIOGRAPHY

Abolpour, B., Javan, M. & Karamouz, M. (2007). Water allocation improvement in river basin using Adaptive Neural Fuzzy Reinforcement Learning approach. *Applied Soft Computing*, 7(1), 265 - 285. doi: https://doi.org/10.1016/j.asoc.2005.02.007.

Barros, M., Tsai, F., Yang, S. L., Lopes, J. & Yeh, W. (2003). Optimization of Large-Scale Hydropower System Operations,. *Journal of Water Resources Planning and Management*, 129(3), 178–188.

Bellman, R. (1954). The theory of dynamic programming. *Bull. Amer. Math. Soc.*, 60(6), 503–515. Consulted at https://projecteuclid.org:443/euclid.bams/1183519147.

Bertsekas, D. P. & Tsitsiklis, J. N. (1995, Dec). Neuro-dynamic programming: an overview. *Proceedings of 1995 34th IEEE Conference on Decision and Control*, 1, 560-564 vol.1. doi: 10.1109/CDC.1995.478953.

Bhattacharya, B., Lobbrecht, A. H. & Solomatine, D. P. (2003). Neural Networks and Reinforcement Learning in Control of Water Systems. *Journal of Water Resources Planning and Management*, 129(6), 458-465. doi: 10.1061/(ASCE)0733-9496(2003)129:6(458).

Can, E. & Houck, M. (1984). Real-Time Reservoir Operations by Goal Programming. *Journal of Water Resources Planning and Management*, 110(3), 297–309.

Castelletti, A. (2002). Reinforcement learning in the operational management of a water system. *MODELING AND CONTROL IN ENVIRONMENTAL ISSUES*.

Castelletti, A., Galelli, S., Restelli, M. & Soncini-Sessa, R. (2010). Tree-based reinforcement learning for optimal water reservoir operation. *Water Resources Research*, 46(9), n/a–n/a. doi: 10.1029/2009WR008898. W09507.

Chandrasekaran, M., Muralidhar, M., Krishna, C. & Dixit, U. (2010). Application of soft computing techniques in machining performance prediction and optimization: a literature review. *The International Journal of Advanced Manufacturing Technology*, 46(5-8), 445-464.

Chaves, P. & Kojiri, T. (2007). Stochastic Fuzzy Neural Network: Case Study of Optimal Reservoir Operation. *Journal of Water Resources Planning and Management*, 133(6), 509-518. doi: 10.1061/(ASCE)0733-9496(2007)133:6(509).

Côté, P. & Leconte, R. (2016). Comparison of Stochastic Optimization Algorithms for Hydropower Reservoir Operation with Ensemble Streamflow Prediction. *Journal of Water Resources Planning and Management*, 142(2), 04015046.

Deka, P. C. & Chandramouli, V. (2009). Fuzzy Neural Network Modeling of Reservoir Operation. *Journal of Water Resources Planning and Management*, 135(1), 5-12.

Faber, B. & Stedinger, J. (2001). Reservoir optimization using sampling {SDP} with ensemble streamflow prediction (ESP) forecasts. *Journal of Hydrology*, 249(1–4), 113 - 133. doi: http://dx.doi.org/10.1016/S0022-1694(01)00419-X.

Fayaed, S. S., El-Shafie, A. & Jaafar, O. (2013). Integrated Artificial Neural Network (ANN) and Stochastic Dynamic Programming (SDP) Model for Optimal Release Policy. *Water Resources Management*, 27(10), 3679–3696. doi: 10.1007/s11269-013-0373-5.

George, K. (1967). Finding Reservoir Operating Rules. *Journal of the Hydraulics Division*, 93(6), 279–322.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning* (ed. 1st). Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Heitsch, H. & Römisch, W. (2003). Scenario Reduction Algorithms in Stochastic Programming. *Computational Optimization and Applications*, 24(2), 187–206. doi: 10.1023/A:1021805924152.

Heydari, M., Othman, F. & Qaderi, K. (2015). Developing Optimal Reservoir Operation for Multiple and Multipurpose Reservoirs Using Mathematical Programming. *Mathematical Problems in Engineering*, 2015(–), 1–11.

Hiew, K., Labadie, J. W. & scott, L. (1989). Optimal Operational Analysis of the Colorado-Big Thompson Project. *in Computerized Decision Support Systems foe Water Managers*, –(–), 632–646.

Hınçal, O., Altan-Sakarya, A. B. & Metin Ger, A. (2011). Optimization of Multireservoir Systems by Genetic Algorithm. *Water Resources Management*, 25(5), 1465–1487.

Huang, W., Harboe, R. & Bogardi, J. J. (1991a). Testing Stochastic Dynamic Programming Models Conditioned on Observed or Forecasted Inflows. *Journal of Water Resources Planning and Management*, 117(1), 28-36. doi: 10.1061/(ASCE)0733-9496(1991)117:1(28).

Huang, W., Harboe, R. & Bogardi, J. J. (1991b). Testing Stochastic Dynamic Programming Models Conditioned on Observed or Forecasted Inflows. *Journal of Water Resources Planning and Management*, 117(1), 28-36.

Jacobson, H. & Mayne, Q. (1970). *Differential Dynamic Programming*. New York: Elsevier.

Johnson, S. A., Stedinger, J. R., Shoemaker, C. A., Li, Y. & Tejada-Guibert, J. A. (1993). Numerical Solution of Continuous-State Dynamic Programs Using Linear and Spline Interpolation. *Operations Research*, 41(3), 484-500.

Kaelbling, L. P., Littman, M. L. & Moore, A. W. (1996). Reinforcement Learning: A Survey. *J. Artif. Int. Res.*, 4(1), 237–285. Consulted at http://dl.acm.org/citation.cfm?id=1622737.1622748.

Kelman, J., Stedinger, J. R., Cooper, L. A., Hsu, E. & Yuan, S.-Q. (1990). Sampling stochastic dynamic programming applied to reservoir operation. *Water Resources Research*, 26(3), 447–454. doi: 10.1029/WR026i003p00447.

Kim, T., Choi, G. & Heo, J.-H. (2009). *Inflow Forecasting for Real-Time Reservoir Operation Using Artificial Neural Network*.

Labadie, J. W. (1998). Reservoir system optimization models. *Water Resources Update Journal*, 107(–), 83–110.

Labadie, J. W. (2004a). Reservoir system optimization models. *Journal of Water Resources Planning and Management-ASCE*, 130(2), 93–111.

Labadie, J. W. (2004b). Optimal Operation of Multireservoir Systems: State-of-the-Art Review. *Journal of Water Resources Planning and Management*, 130(2), 93-111. doi: 10.1061/(ASCE)0733-9496(2004)130:2(93).

Lamond, B. F. (2003). Stochastic Optimization Of A Hydroelectric Reservoir Using Piecewise Polynomial Approximations. *INFOR: Information Systems and Operational Research*, 41(1), 51-69.

Larsen, C. T., Doorman, G. L. & Mo, B. (2015, June). Evaluation of scenario reduction methods for stochastic inflow in hydro scheduling models. *PowerTech, 2015 IEEE Eindhoven*, pp. 1-6.

Larson, R. (1965). Dynamic programming with reduced computational requirements. *IEEE Transactions on Automatic Control*, 10(2), 135-143.

Lee, J.-H. & Labadie, J. W. (2007). Stochastic optimization of multireservoir systems via reinforcement learning. *Water Resources Research*, 43(11), n/a–n/a. doi: 10.1029/2006WR005627. W11408.

Loganathan, G. & Bhattacharya, D. (1990). Goal-Programming Techniques for Optimal Reservoir Operations. *Journal of Water Resources Planning and Management*, 116(6), 820–838.

Mariano-Romero, C. E., Alcocer-Yamanaka, V. H. & Morales, E. F. (2007). Multi-objective optimization of water-using systems. *European Journal of Operational Research*, 181(3), 1691 - 1707. doi: http://dx.doi.org/10.1016/j.ejor.2006.08.007.

Mousavi, S. J., Ponnambalam, K. & Karray, F. (2005). Reservoir Operation Using a Dynamic Programming Fuzzy Rule–Based Approach. *Water Resources Management*, 19(5), 655–672. doi: 10.1007/s11269-005-3275-3.

Mousavi, S. J., Karamouz, M. & Menhadj, M. B. (2004). Fuzzy-State Stochastic Dynamic Programming for Reservoir Operation. *Journal of Water Resources Planning and Management*, 130(6), 460-470. doi: 10.1061/(ASCE)0733-9496(2004)130:6(460).

Murray, D. M. & Yakowitz, S. J. (1979). Constrained differential dynamic programming and its application to multireservoir control. *Water Resources Research*, 15(5), 1017–1027.

Nandalal, K. D. W. & Bogardi, J. J. (2013). *Dynamic Programming Based Operation of Reservoirs: Applicability and Limits*. –: Part of International Hydrology Series.

Pennanen, T. & Koivu, M. (2002). Integration quadratures in Discretization of Stochastic Programming. *E-Print Series, http://www.speps.info, 2002.*, –(–), –.

Pereira, M. V. F. & Pinto, L. M. V. G. (1991). Multi-stage stochastic optimization applied to energy planning. *Mathematical Programming*, 52(1), 359–375. doi: 10.1007/BF01582895.

Piekutowski, M., Litwinowicz, T. & Frowd, R. J. (1994). Optimal short-term scheduling for a large-scale cascaded hydro system. *IEEE Transactions on Power Systems*, 9(2), 805-811.

Powell, W. B. (2011). *Approximate Dynamic Programming*. Hoboken, New Jersey: John Wiley & Sons, Inc.

Rani, D. & Moreira, M. M. (2010). Simulation–Optimization Modeling: A Survey and Potential Application in Reservoir Systems Operation. *Water Resources Management*, 24(6), 1107–1138. doi: 10.1007/s11269-009-9488-0.

Reis, L. F. R., Walters, G. A., Savic, D. & Chaudhry, F. H. (2005). Multi-Reservoir Operation Planning Using Hybrid Genetic Algorithm and Linear Programming (GA-LP): An Alternative Stochastic Approach. *Water Resources Management*, 19(6), 831–848. doi: 10.1007/s11269-005-6813-0.

Russell, S. O. & Campbell, P. F. (1996). Reservoir Operating Rules with Fuzzy Programming. *Journal of Water Resources Planning and Management*, 122(3), 165-170. doi: 10.1061/(ASCE)0733-9496(1996)122:3(165).

Saad, M., Turgeon, A. & Stedinger, J. R. (1992). Censored-data correlation and principal component dynamic programming. *Water Resources Research*, 28(8), 2135–2140. doi: 10.1029/92WR00896.

Saad, M., Bigras, P., Turgeon, A. & Duquette, R. (1996). Fuzzy Learning Decomposition for the Scheduling of Hydroelectric Power Systems. *Water Resources Research*, 32(1), 179–186. doi: 10.1029/95WR02971.

Séguin, S., Fleten, S.-E., Côté, P., Pichler, A. & Audet, C. (2017). Stochastic short-term hydropower planning with inflow scenario trees. *European Journal of Operational Research*, 259(3), 1156 - 1168. doi: https://doi.org/10.1016/j.ejor.2016.11.028.

Shawwash, Z. K., Siu, T. K. & Russell, S. O. D. (2000). The B.C. Hydro short term hydro scheduling optimization model. *IEEE Transactions on Power Systems*, 15(3), 1125–1131.

Singh, S., Jaakkola, T., Littman, M. L. & Szepesvári, C. (2000). Convergence Results for Single-Step On-Policy Reinforcement-Learning Algorithms. *Machine Learning*, 38(3), 287–308. doi: 10.1023/A:1007678930559.

Suen, J.-P. & Eheart, J. W. (2003). Evaluation of Neural Networks for Modeling Nitrate Concentrations in Rivers. *Journal of Water Resources Planning and Management*, 129(6), 505-510. doi: 10.1061/(ASCE)0733-9496(2003)129:6(505).

Sutton, R. S. & Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. Cambridge, Massachusetts: MIT Press.

Tarim, S. A., Manandhar, S. & Walsh, T. (2006). Stochastic Constraint Programming: A Scenario-Based Approach. *Constraints*, 11(1), 53–80. doi: 10.1007/s10601-006-6849-7.

Tejada-Guibert, J. A., Alberto, J., Johnson, S. A. & Stedinger, J. R. (1993). Comparison of two approaches for implementing multireservoir operating policies derived using stochastic dynamic programming. *Water Resources Research*, 29(12), 3969–3980. doi: 10.1029/93WR02277.

Tejada-Guibert, J. A., Alberto, J., Johnson, S. A. & Stedinger, J. R. (1995). The Value of Hydrologic Information in Stochastic Dynamic Programming Models of a Multireservoir System. *Water Resources Research*, 31(10), 2571–2579. doi: 10.1029/95WR02172.

Tizhoosh, H. R. & Ventresca, M. (2008). *Oppositional Concepts in Computational Intelligence*. Heidelberg: Springer-Verlag Berlin Heidelberg.

Turgeon, A. (1980). Optimal operation of multireservoir power systems with stochastic inflows. *Water Resources Research*, 16(2), 275–283. doi: 10.1029/WR016i002p00275.

Turgeon, A. (2005). Solving a stochastic reservoir management problem with multilag autocorrelated inflows. *Water Resources Research*, 41(12), n/a–n/a. doi: 10.1029/2004WR003846. W12414.

Turgeon, A. (1987). An application of parametric mixed-integer linear programming to hydropower development. *Water Resources Research*, 23(3), 399–407. doi: 10.1029/WR023i003p00399.

Wardlaw, R. & Sharif, M. (1999). Evaluation of Genetic Algorithms for Optimal Reservoir System Operation. *Journal of Water Resources Planning and Management*, 125(1), 25-33. doi: 10.1061/(ASCE)0733-9496(1999)125:1(25).

Wen, C.-G. & Lee, C.-S. (1998). A neural network approach to multiobjective optimization for water quality management in a river basin. *Water Resources Research*, 34(3), 427–436. doi: 10.1029/97WR02943.

Yeh, W. W.-G. (1985). Reservoir Management and Operations Models: A State-of-the-Art Review. *Water Resources Research*, 21(12), 1797–1818. doi: 10.1029/WR021i012p01797.