Storage System Management Using Reinforcement Learning Techniques and Nonlinear Models

by

Masoud Mahootchi

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2009

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

In this thesis, modeling and optimization in the field of storage management under stochastic condition will be investigated using two different methodologies: Simulation Optimization Techniques (SOT), which are usually categorized in the area of Reinforcement Learning (RL), and Nonlinear Modeling Techniques (NMT).

For the first set of methods, simulation plays a fundamental role in evaluating the control policy: learning techniques are used to deliver sub-optimal policies at the end of a learning process. These iterative methods use the interaction of agents with the stochastic environment through taking actions and observing different states. To converge to the steady-state condition where policies and value functions do not change significantly with the continuation of the learning process, all or most important states must be visited sufficiently. This might be prohibitively time-consuming for large-scale problems. To make these techniques more efficient both in terms of computation time and robust optimal policies, the idea of Opposition-Based Learning (OBL-Type I and Type II) is employed to modify/extend popular RL techniques including Q-Learning, Q(λ), sarsa, and sarsa(λ). Several new algorithms are developed using this idea. It is also illustrated that, function approximation techniques such as neural networks can contribute to the process of learning. The state-of-the-art implementations usually consider the maximization of expected value of accumulated reward. Extending these techniques to consider risk and solving some well-known control problems are important contributions of this thesis.

Furthermore, the new nonlinear modeling for reservoir management using indicator functions and randomized policy introduced by Fletcher and Ponnambalam, is extended to stochastic releases in multi-reservoir systems. In this extension, two different approaches for defining the release policies are proposed. In addition, the main restriction of considering the normal distribution for inflow is relaxed by using a beta-equivalent general distribution. A five-reservoir case study from India is used to demonstrate the benefits of these new developments. Using a warehouse management problem as an example, application of the proposed method to other storage management problems is outlined.

Acknowledgements

This research study would not have culminated into a successful conclusion without the support of my supervisors, Professors K. Ponnambalam and H.R. Tizhoosh. I would like to express my sincere gratitude to them for valuable assistance, support and guidance during my Ph.D. education. It is a pleasure to express my deepest gratitude to the members of the examination committee, Professors P. Calamai, J. Kofman, and O. Basir, for providing editorial and critical comments. My special thanks to Professor B. Lamond, the external reviewer from the University of Laval, who provided a detailed report and helpful editorial comments. I am also grateful to my dear friend and colleague, Dr. J. Mousavi from Amirkabir University of Technology in Tehran for useful discussions regrading the reservoir problem. I would also like to convey my appreciation to the Ministry of Science, Research, and Technology of the Islamic Republic of Iran for providing the financial support through the duration of my Ph.D. education. Finally, I wish to express my love and gratitude to my beloved family, my wife and kids, for their understanding and patience through the duration of this study.

Dedication

I dedicate this thesis to my wife, Fatemeh, my lovely daughter, Yasaman, and my little son, Yahya.

Contents

List of Tables

List of Figures

Acronyms

A/D Aggregation-Decomposition

A/D-DP Aggregation Decomposition Dynamic Programming

CCP Chance-Constrained Programming

CDF Cumulative Distribution Function

DAV Distance of Action-Value functions

DB-CDF Double-Bounded Cumulative Density Function

DB-PDF Double-Bounded Probability Density Function

DC Distribution Center

DP Dynamic Programming

DPSA DP with Successive Approximation

ECDF Empirical Cumulative Distribution Function

EXR Expected Reward

FP Fletcher-Ponnambalam

FRB Fuzzy Rule-Based modeling

GA Genetic Algorithms

HDDI Hierarchical, Distributed, Dynamic, Inventory

IDP Incremental Dynamic Programming

LP Linear Programming

MAM-DP Multilevel Approximation Dynamic Programming

MDPs Markov Decision Processes

MIDP Multi-level Incremental Dynamic Programming

MLP Multi-Layer Perceptron networks

NCDF Normal Cumulative Distribution Function

NLP Nonlinear Programming

NMT Non-Linear Modeling Techniques

NN Neural Networks

OBL Opposition-Based Learning

OQLA-I Type I Opposition Q(λ)

OQLA-II Type II Opposition Q(λ)

OQLR-I Type I Opposition Q-Learning

OQLR-II Type II Opposition Q-Learning

PAP Parambikulam-Aligar Project

QLA Q(λ)

QLR Q-Learning

RL Reinforcement Learning

SDP Stochastic Dynamic Programming

SLA Sarsa(λ)

SLP Successive Linear Programming

SLR Sarsa

SOS Summation of Standard deviation

SOT Simulation-Optimization Techniques

SP Stochastic Linear Programming

SQP Successive Quadratic Programming

TD Temporal-Difference

TSP Two-Stage Stochastic Programming

Chapter 1

Introduction

Efficient storage management requires optimal policies that manage storage volumes and releases and aim to maximize benefits or minimize costs. Many of the information used in storage management optimization models are affected by stochastic uncertainty. In addition, nonlinearity, and the large scale nature of this problem makes the solution process challenging [1]. Numerous mathematical and heuristic optimization or simulation techniques have been developed in the past for storage management optimization; however, most of them suffer from excessive computational expenses. Most of these techniques have included varieties of simplifications and approximations which usually make the operating policies suboptimal in practice. Labadie [2] believes that the future prospects for implementing reservoir optimization models depends on strong connections with simulation techniques and using heuristic methods.

In the following sections, the optimization models will be described for reservoir management problems that can be used for warehousing management application as well. Some important concepts and the optimization model which is usually used for these problems will be presented. In the last section, the thesis outline and its contributions will be outlined.

1.1 Problem description

In most storage applications, the main objective of optimization is to find a policy (or an operating rule) that will optimize the productivity and the efficiency of storage operations for short, medium, or long periods under some stochastic situations described later. In most surface water resource management problems all or some combinations of the following can be simultaneously considered in the objective function:

- Hydropower generation
- Irrigation or municipal industrial water supply
- Recreational enhancement
- Water quality control
- Controlling shoreline encroachment
- Flood control
- Navigation
- Fish and wildlife enhancement

A warehouse in production planning or supply chain plays the role of buffer storage to fill the gap between supply and demand [3]. In the warehousing problem, as another example of storage management, the problem is to find a policy to minimize the total of the following costs:

1. the fixed ordering cost;
2. the transportation cost;
3. the cost of holding products;
4. the capital cost;

5. the lost-sale cost (the cost related to demands which have not been met); and

6. the cost of stockouts (excessive items or products that should be returned or sold in an open market for cheaper prices).

In other storage management applications, analogous objective functions should be considered. The main state variable in all these problems is the storage, which is bounded by minimum and maximum values. However, the terminologies, interacting variables or parameters, and definitions might be different. In this thesis, we concentrate on reservoir applications but also extend the developed stochastic models and techniques to warehousing problems in order to demonstrate that these methods are applicable to other storage applications.
In the following subsections, important terminologies and concepts which are widely used in the literature of optimization or simulation techniques for reservoir and warehousing management will be discussed.

1.1.1 Uncertainty

In optimization and simulation, some of the parameters or uncontrolled variables are considered as random variables. The characteristics of these variables need to be determined to solve the problem. Unregulated inflows, net evaporations, hydrologic, and economic parameters and system demands are often considered as random variables in reservoir management [2, 4]. Of course, in some cases these variables could be considered deterministic. For example, if the inflows of some rivers are not highly variable, their expectation can be used in the optimization or simulation model [5]. In many real-world applications of reservoir management, especially when we are dealing with power generation or supplying water needed for irrigation, the price of water is also a very important parameter whose randomness should be taken into account in the operational model.
In warehousing problems, both demand and supply should be considered stochastic; however, in most cases demand is the main random variable in the system model [3, 6].

1.1.2 Operating policy

There are two policy types in the literature of reservoir management: Open-loop and closed-loop. Based on the original definition [7, 8], a policy is called an open-loop policy if the corresponding optimization or simulation model does not consider the future uncertainty. In other words, if the optimal decisions are not functions of the system state (e.g., storage volume or previous inflow), it is called an open-loop policy; otherwise, the policy is closed-loop [2].

An operating policy/rule for N storage reservoirs and T periods in a cycle (e.g., twelve months in a year), which is usually created based on optimization or simulation algorithms, is usually categorized into one of two types [2, 9] in which $\bar{f}(t, s^t)$ is the operating policy with respect to period t and storage volume in that period, s^t. The two types are:

- **release-based**: The amount of water to be released from each individual reservoir or supplier with consideration of the current state at the time of decision-making. As it is clear in the following function, for each state-period pair, a specific release (d^t) has to be determined:

$$d^t = \bar{f}(t, s^t).$$

- **storage-based:** storage level at the end of each period of each individual reservoir given the current state at the time of decision-making. This is denoted by s^{t+1} for each state-period pair via the following function:

$$s^{t+1} = \bar{f}(t, s^t).$$

These two types of policy definitions are easily extendable to the warehousing problem in which d^t is the number of items to be ordered from either external (supplier) or internal (warehouse or retailers) sources. In this thesis, only the release-based policy is considered.

1.1.3 Performance measurement in reservoir problems

To measure the performance of reservoir operations, three different criteria are usually incorporated in objective functions [2]:

- Efficiency: This corresponds to maximizing or minimizing the summation of discounted income or cost during planning and designing periods as well as real-time operation. In stochastic problems, this criterion would be adjusted accordingly.

- Survivability: If the goal of reservoir management is to maximize the total discounted income, this criterion should guarantee that the future income is higher than minimum level considered.

- Sustainability: This can be measured by a weighted combination of reliability, resilience, and vulnerability. Reliability is related to the number of failures and indicates the average of time that the system runs without failure. Resilience is pertinent to the expectation of time distance between a step with failure and the next step without failure, indicating how fast the system recovered from the failure. Vulnerability is the expectation of time distance between two successive failures. The failure is determined based on various economic, environmental, ecological, or social criteria. In general, a reservoir with more reliability and resilience and less vulnerability is more sustainable [10]. For example, in irrigation, resilience has an important effect on crop yield, and must be considered in the objective function [11].

1.1.4 Illustration of two multi-storage applications

Depending on geography and purposes, different configurations of reservoirs are constructed. The interrelation between these reservoirs could be serial, parallel, or a combination of both (see Figure 1.1). In many real-world applications, the whole structure of the system has to satisfy a specific demand in the downstream reservoirs. It means that the releases of these reservoirs are only utilized to meet the total demand including irrigation, municipal water supply, etc. However, in some problems, in addition to the policy that these demands have to be satisfied by the downstream reservoirs, each individual reservoir upstream has to assist in meeting a portion of the whole demand. For example, there is a powerplant corresponding to each reservoir having a responsibility

to generate power. It is clear that the water released to generate hydroelectric power in each reservoir is again conducted to downstream reservoirs for other purposes [12]. It is worth mentioning that a portion of release in some reservoirs in the configuration can be assigned for other purposes because of existing water-sharing agreements between different states or governments [13].

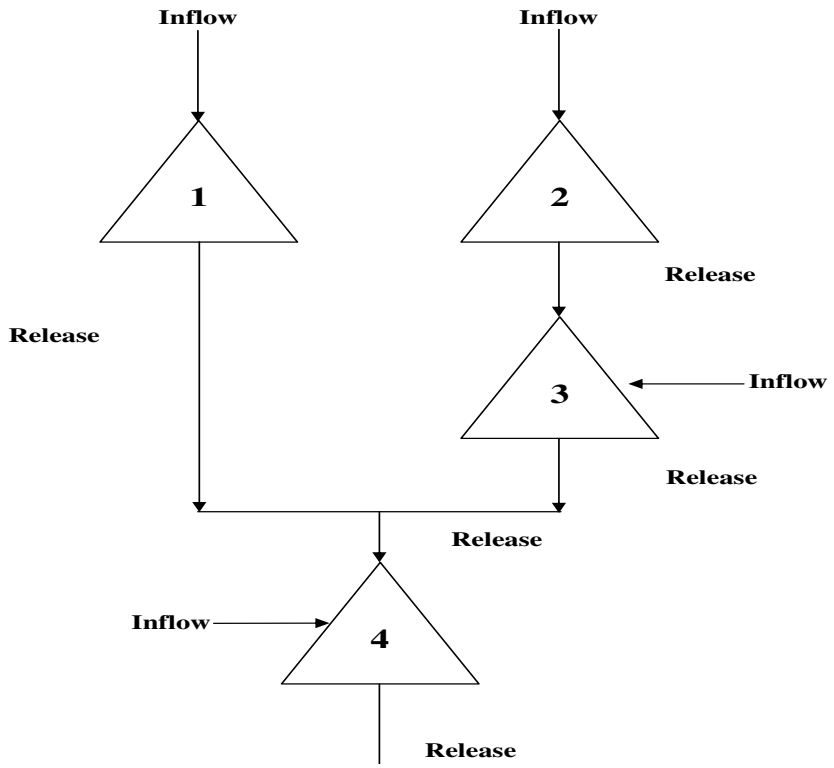

Figure 1.1: A four-reservoir configuration with demand downstream

Figure 1.2 demonstrates a schematic example of a warehousing problem in which there are three different entities: 1- Supplier, 2-Distribution Center (DC), 3- Depot or Retailer. Suppliers in the supply chain can supply the needed items to distribution centers. The distribution center, which is sometimes called the warehouse in the literature, obtains the ordered quantity from suppliers or from other distribution centers to satisfy the demands for some smaller storage spaces, called retailers or depots, in their downstream. In the real world, distribution centers could be connected to each other; however, there

is usually no connection between retailers and suppliers.

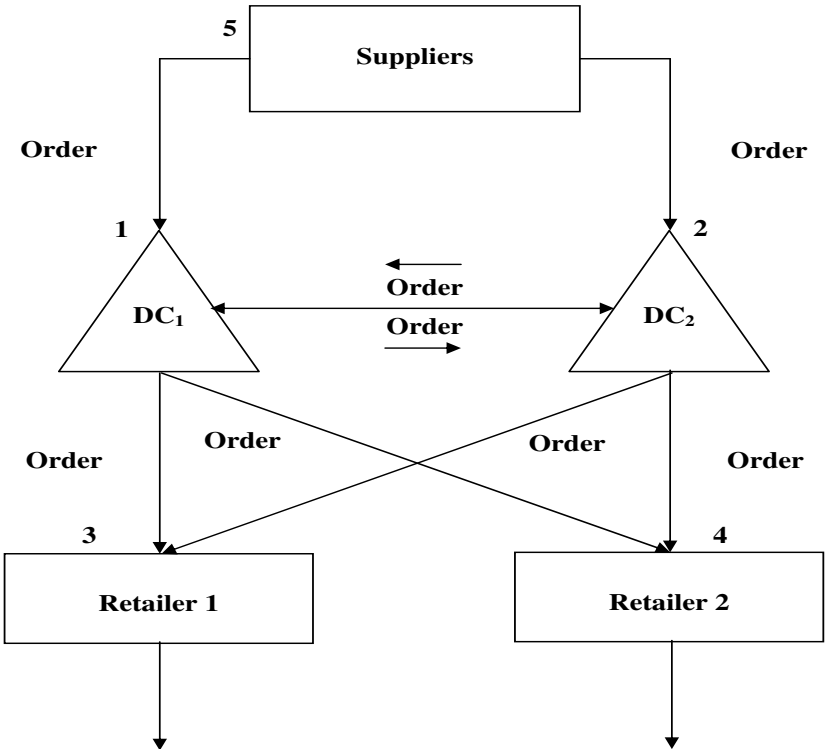

Figure 1.2: A schematic representation of a multi-echelon warehousing problem

1.2 Modeling formulation

In the following subsections, objective functions and constraints for storages and optimization models will be described.

1.2.1 Objectives

If systems under study are deterministic, a general form of the objective function in a multi-storage application can be written as [9, 12]

$$f(u^t) = \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T} f_{ij}^t(s_i^t,\ u_{ij}^t,\ D_i^t), \tag{1.1}$$

where f is the cost or revenue function which could be a function of s_i^t (the storage volume of reservoir i or the current stock of warehouse i), u_{ij}^t (the amount of release from reservoir i to reservoir j, or the number of items to be ordered by warehouse j from source i), and D_i^t (the demand given for source i in period t). N is the number of storages in the system model.

1.2.2 Constraints

The constraints in the optimization model of storage management applications can be summarized as follows:

Balance equations

These equality constraints impose the conservation of mass (in appropriate unit) with respect to inputs and outputs of storages:

$$\begin{aligned} s_i^{t+1} \;=\; s_i^t \;-\; \sum_{l=1,\ l\neq i}^{N_D} u_{il}^t \times \delta_{il} \;+\; I_i^t \;-\; \nu_i^t \;+\; \sum_{l=1,\ l\neq i}^{N_U} u_{li}^t \times \delta_{li} \\ \forall\ i \;=\; 1 \cdots N \;\;\&\;\; t \;=\; 1 \cdots T, \end{aligned} \tag{1.2}$$

where I_i^t is the amount of inflow to every storage i in period t, ν_i^t is the loss at storage i in period t, and u_{il}^t is the amount of controlled inflow from storage i to storage l in period t, N_D is the number of downstream reservoirs, and N_U is the number of upstream reservoirs. δ_{ij}, an element of the routing matrix, can be defined as follows:

$$\delta_{ij} = \begin{cases} 1 & \text{if } i^{th} \text{ storage is physically connected to } j^{th} \text{ storage} \\ 0 & \text{otherwise.} \end{cases} \tag{1.3}$$

Losses in the above formulation can be represented as a percentage of the storage level

at the end of each period. However, in many applications, it is expressed as a function of average storage in a period [4, 9].

In warehousing problems, I_i^t is input to warehouses or retailers, defined as the number of items received by source i from suppliers, which is usually considered deterministic, and $\sum_{l}^{N} u_{il}^t \delta_{il}$ can be replaced by D_i^t as a stochastic demand of source i in period t.

Maximum and minimum storages

The following constraints in the reservoir management application provide flood control (or stockout control in warehousing problems) while considering some aspects of recreation or assuring a minimum level for powerplant operation.

$$s_{i,MIN}^t \leq s_i^t \leq s_{i,MAX}^t \quad \forall \; i \; = \; 1 \; \cdots \; N; \;\; t \; = \; 1 \; \cdots \; T, \tag{1.4}$$

where $s_{i,MAX}^t$ and $s_{i,MIN}^t$ are the maximum and minimum level of storage i in period t, respectively.

Maximum and minimum releases

In reservoir applications, the purpose of these constraints are to provide a suitable water quality for the existence of wildlife and fish, and to prevent floods downstream. In warehousing applications, these are the limitations for the items to be ordered,

$$R_{i,MIN}^t \leq \sum_{l=1}^{N} u_{il}^t \times \delta_{il} \leq R_{i,MAX}^t \quad \forall \; i \; = \; 1 \; \cdots \; N; \;\; t \; = \; 1 \; \cdots \; T \tag{1.5}$$

where $R_{i,MAX}^t$ and $R_{i,MIN}^t$ are the maximum and minimum releases from reservoir i or the maximum or minimum items that can be ordered from source i in period t, respectively.

1.3 The contributions and the thesis outline

The final goal in storage management applications is usually to find an optimal operating policy with respect to some physical constraints in a stochastic situation, such that the

respective objective functions are optimized. Most modeling techniques in this area use some types of simplification or estimation to cope with the stochastic situation. Moreover, the optimization approaches such as discrete dynamic programming, a well-known technique for stochastic problems, suffer from the curse of dimensionality, that is, when there are more than two or three storages, the optimization problem is impossible to solve.

The goals in this thesis are to propose 1) new learning techniques based on simulation and 2) new nonlinear models that do not suffer from curse of dimensionality when applied to multi-storage management problems in a stochastic environment.

It is also demonstrated that how to apply Reinforcement Learning (RL) techniques, as simulation-based and adaptive approaches, to storage management application. Furthermore, to speed up the process of learning, two Opposition-Based Learning (OBL) schemes will be proposed to improve four Reinforcement Learning (RL) techniques.

Although RL techniques, which could be implemented based on simulation (off-line) or interactions with the real environment (on-line), can be applied in multi-storage applications, they still suffer from the slow convergence and high deviation in results for large-scale problems. One way to make these learning techniques applicable is to use the Aggregation-Decomposition (A/D) methods proposed by Turgeon [14], by Ponnambalam and Adams [13], or by Archibald et al. [15]. The A/D technique decomposes the whole problem into different subproblems, where each subproblem is composed of only two reservoirs: the actual reservoir and an aggregated one. Ponnambalam and Adams [13] proposed a method to weight the aggregated reservoir in every subproblem by finding coefficients through simulation. To apply RL techniques using A/D in an appropriate way, we need to know these coefficients before implementing RL techniques. We propose to use the Fletcher-Ponnambalam (FP) stochastic models [16] to find the respective coefficients which can be used for aggregated reservoirs in applying RL techniques using A/D. Moreover, to make these nonlinear models more general, they will be extended to multi-storage applications with a stochastic release policy where the distribution of

inflows or demands are not normally distributed.

It is worth mentioning that the developed models and techniques will be described and experimentally verified using reservoir applications. In one part of this thesis RL techniques are also applied to a grid world problem as a popular example in RL. A model for warehouse management problems, which is analogous to reservoir management problems, is also developed.

Chapter 2 will be assigned to the background and literature review of relevant research on multi-reservoir and multi-echelon warehousing problems. In Chapter 3, simulation-based and nonlinear model-based optimization methodologies for single- and multi-storage applications, including reservoir and warehouse problems will be proposed. These methods will be applied to single and multi-reservoir applications in addition to a grid-world problem in Chapter 4. A case study with five reservoirs is also presented here. The conclusions and future work is summarized in Chapter 5.

Chapter 2

Background and Literature Review

2.1 Introduction

The complex problem of reservoir operations management can be broken down into three stages: 1) planning, 2) designing, and 3) real time operations. All three stages are fully related, such that the outputs of one stage would be utilized as inputs to another and vice versa [1]. The purpose of the planning stage is to find the optimal capacity, considering random demands and inflows. The final solution of this stage, along with inflows, are then used in the design stage to find monthly releases. Having obtained the sequence of decisions with respect to the maximization of the yield in the design stage, the operating rules are constructed and used as a guideline in real time operations to determine releases in a shorter time scale such as hourly or daily. This is schematically shown in Figure 2.1.

In general, to optimize the operations of multi-reservoirs in an uncertain situation, two types of stochastic programming are used: implicit and explicit stochastic programming. In implicit modeling, a long synthetic sequence of data corresponding to random variables such as inflow or net evaporation is generated, using forecasting methods based on existing knowledge and the history of the system. These data, as representatives of the stochastic behavior of the environment, are then considered as inputs in deterministic optimization to be solved. The main advantage of this method is that the generated

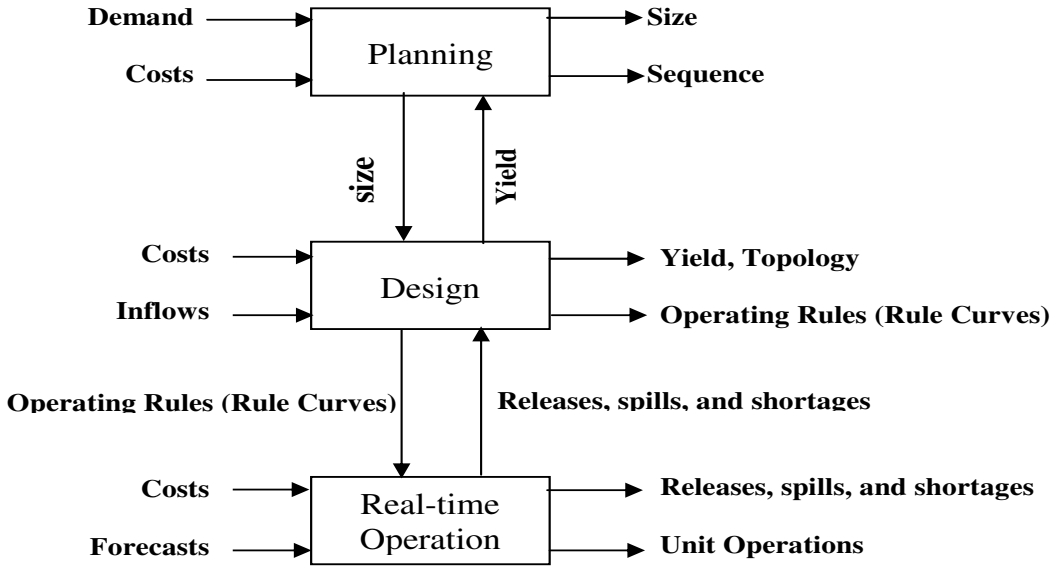

Figure 2.1: Various Operating Horizons in the management of reservoirs [1]

data would represent many aspects of stochastic relations, including spatial or temporal correlations. The main disadvantage is that the optimal operating rules, which are used to conduct the operations of the system in the real-time operations, are based on the assumed hydrologic time series [2].

In the explicit stochastic type of formulation, instead of using the synthetic data of the time series, the probability distributions of random variables are used to incorporate the stochastic nature of the environment in the model. However, in contrast to the implicit stochastic formulation, models in explicit methods may be complex, and in some methods like dynamic programming excessive computing time may be needed [17]. Figure 2.2 represents a schematic view of using both the implicit and the explicit type of programming in optimizing the operations of reservoir systems.

Many different algorithms for these two schemes have been developed: Linear Programming (LP), Nonlinear Programming (NLP), Dynamic Programming (DP) [4], and Simulation-Optimization Techniques (SOT) [18] are the major tools being used. Because reservoir management is a broad area, the survey is performed only for some of the relevant methods that are related to multi-reservoir systems.

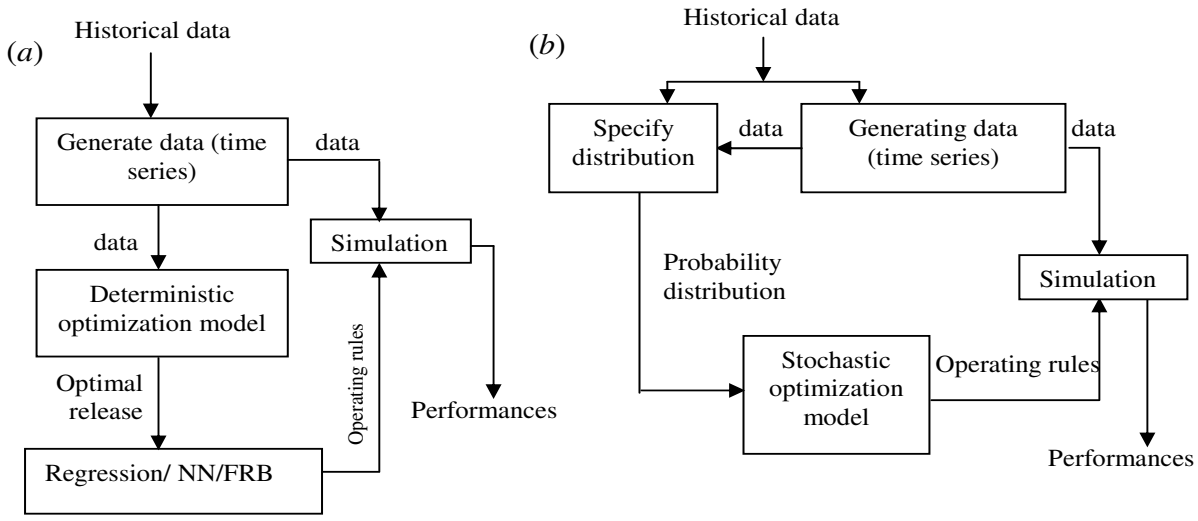

Figure 2.2: Schematic view of implicit and explicit stochastic programming (a) implicit, (b) explicit [2]

Moreover, because the reinforcement learning and nonlinear modeling techniques are used to develop the new methods in storage management, some sections at the end of this chapter will present detailed background information on relevant concepts and mathematical formulations.

2.2 Implicit stochastic approaches

In the following subsections, a few important implicit stochastic linear and nonlinear optimization methods, applied in multi-reservoir case studies, are reviewed.

2.2.1 Linear Programming (LP)

Linear programming is a powerful method, when a solution exists, it provides a global solution, and is widely used for the operation of multi-reservoir systems with linear objective functions and constraints. Draper and Adamowski [19] and Bechard et al. [20] used linear programming to find the optimal solution for the management of 17 and 22

reservoirs, respectively. However, they encountered huge computational burdens. Hiew et al. [21] applied this technique in optimizing seven reservoirs in Colorado. Crawley and Dandy [22] have applied piecewise linear approximation in a multi-reservoir configuration to maximize the yield of the system for a specific reliability, an issue considered in this thesis.

2.2.2 Nonlinear Programming (NLP)

In many applications of multi-reservoir operations planning, linear techniques are not applicable due to the existence of nonlinear and non-separable relations in objective functions and/or constraints. The feasible regions are generally non-convex. In this situation, various nonlinear algorithms such as Successive Linear Programming (SLP), Successive Quadratic Programming (SQP), and other techniques have been considered to solve the problem. If these functions are not differentiable, using nonlinear methods might be difficult [23].

Among different nonlinear methods in large-scale multi-reservoir planning, SLP has led to reasonable results in many real-world applications, both in terms of computational requirements and optimality of final solution [24, 25]. SLP uses the two first terms of Taylor's series to linearize the nonlinear objective function, while SQP uses the first three terms. In both methods, first order linearization of nonlinear constraints are considered. SQP methods required more CPU time compared to SLP [25]. Therefore, in many cases, especially when objective function and/or constraints are nearly linear in the region of interest, SLP is used.

2.2.3 Dynamic Programming (DP)

Dynamic programming (DP) is an optimization technique to cope with the nonlinear, non-convex, and discontinuous nature of problems. This method was formulated by Bellman [26]. DP breaks the larger problem into a sequence of connected smaller subproblems; only one subproblem is solved at any instant with the solution available from

the rest of the subproblems.

To apply an implicit stochastic DP in a reservoir application, a finite sequence of inflows and evaporations over all time horizons is generated. Then, a deterministic DP is implemented for this time horizon [4, 5]. The main difficulty in DP which makes it harder to apply in large-scale applications is the *curse of dimensionality*. To overcome this drawback in the context of implicit stochastic programming, some versions of Discrete Dynamic Programming (DDP) with various approximations have been proposed: 1-DP with Successive Approximation (DPSA) [27], 2-Incremental Dynamic Programming (IDP) [28], and 3-Multi-level Incremental Dynamic Programming (MIDP) [29].

The purpose of DPSA is to reduce the number of system states in every iteration of the DDP method, such that it converges to a specific solution after a finite number of iterations [27, 30]. Larson and Korsak [27, 30] used the DPSA in a standard problem composed of four reservoirs, to optimize the power generation in each reservoir. In IDP, which has been proposed to improve the solution of DPSA, the method should be started with an initial admissible trajectory over the time horizon T. However, in contrast to DPSA, the state variables corresponding to all reservoirs are active. This method has been used to tackle the four-reservoir application [31].

MIDP is the combination of IDP and DPSA. It consists of several levels, such that the number of active states in each level are different. Nopmongcol and Askew [29] applied this method in the four-reservoir case study solved by Larson [27]. All these revisions have lost some important advantages of DDP in practical applications, such as the globality of the optimal solution.

2.3 Explicit stochastic approaches

In the following subsections, a few relevant explicit stochastic optimization methods applied in multi-reservoir case studies are reviewed.

2.3.1 Stochastic Dynamic Programming (SDP)

Stochastic Dynamic Programming (SDP) is another way to directly incorporate stochasticity of system parameters into the model [26, 32]. The final solution of SDP is an operating policy including decisions for all possible combinations of states in reservoirs. As mentioned, the probability distribution of inflows directly influences the model in SDP. The state of the system is usually divided into some specific discrete values, and the recursive function, which is a function of the state and the decision taken, is updated in every iteration.

There are two ways for finding the maximum or minimum in terms of Bellman optimality and Bellman equation: value iteration, and policy iteration. The process of finding the solution in these two techniques is performed iteratively such that in each iteration two steps are taken: policy evaluation and policy improvement.

There are very few methods based on SDP in literature to cope with large-scale water resource management. Turgeon [14] proposed a method called Aggregation Decomposition Dynamic Programming (A/D-DP). This method, which has been specifically applied in some serial or parallel reservoirs, decomposes the whole reservoir system into two reservoirs: actual and aggregated reservoir. Then, a SDP is used to solve the problem with two state variables, including the state of the actual reservoir (e.g. storage of the reservoir) and the state of the aggregated reservoir, which is the combination of all states in downstream reservoirs (e.g. the summation of all storages in downstream reservoirs).

Ponnambalam and Adams [13] proposed a Multilevel Approximation Dynamic Programming (MAM-DP) method based on A/D-DP and MIDP. In this new technique, each stage could be composed of more than one reservoir as actual reservoirs. It is also applicable to different sets of arrangement of reservoirs. Although it seems that the solution of these methods is very close to the global solution, the effectiveness of these algorithms are really problem-dependent.

Value iteration in SDP

In the value iteration approach of discrete SDP, the value function ($V^t(i)$) as the maximum accumulated reward from period t to termination for given state i) is obtained based on the optimality equation proposed by Bellman [26] as:

$$V^t(i) \;=\; \max_{a\in A(i)} \left[Re_i^t(a) \;+\; \gamma \times \sum_j P_{ij}^t(a) \times V^{t+1}(j) \right], \tag{2.1}$$

where $Re_i^t(a)$, the expected immediate reward, is given by,

$$Re_i^t(a) \;=\; \sum_j P_{ij}^t(a) \;\times\; r_{ij}^t(a), \tag{2.2}$$

where $P_{ij}^t(a)$ is the probability of transition from state i to state j when an action a in period t is taken, $r_{ij}^t(a)$ is the reward function pertinent to action a for transition from state i to state j in period t, $A(i)$ is the set of admissible actions for this state, and γ is a discount factor. Equation 2.1 is solved from the back, i.e., $t : T-1, \cdots 1$. At any given time, only the problem of time t is solved while the solution from time $t+1$ to the end is available in the value function V_j^{t+1}, $\forall\ j$. This equation is also the basis for the RL techniques explained later.

2.3.2 Chance-Constrained Programming (CCP)

Chance-Constrained Programming (CCP), which was initially introduced by Charnes and Cooper [33], is a common and popular technique in the modeling of stochastic problems. Revelle et al. [34] has applied this technique in reservoir management by incorporating a linear decision rule into the formulation. In this method, the constraints of the continuity equation (balance equation) and the storage boundaries are combined using statistical information of inflow, and constitute a set of new constraints, which are completely independent of the stochastic parameter in every period. However, Stendinger and Strycharczyk [35] describe serious disadvantages of this method, especially its pessimistic solutions.

2.3.3 Stochastic programming

Stochastic Linear Programming (SP) is a specific type of stochastic programming, which is usually called two-stage stochastic linear programming with recourse function, because it models the uncertain situation in a linear fashion. Two-Stage Stochastic Programming (TSP) is another version of stochastic programming which is similar to SP, with nonlinear chance constraints and a nonlinear objective function, and may have variance terms in the objective function [36].

In SP or TSP, there are two sets of decision variables. The objective function is composed of two main terms: the cost minimization(or the revenue maximization) with respect to first-stage decisions, and the expected value of cost (revenue) corresponding to the future random variable realizations [37]. The main disadvantage of this scenario-based method is that the number of constraints which include random variables will usually increase with the number of scenarios. Therefore, using this kind of modeling is limited in real-world problems [38]. Of course, this difficulty could be alleviated by using the Benders decomposition method [39], in which the original problem is divided into subproblems and a master problem which actively coordinate together to obtain the optimal policy. This method is also called dual dynamic programming.

2.3.4 Fletcher-Ponnambalam (FP) model

This method was initially developed and implemented for decision-making in single and multi-reservoir applications over the long term, by Fletcher and Ponnambalam [16, 40], in order to consider the stochastic nature of inflow. Two main assumptions were considered in the respective stochastic models: that the releases are deterministic, and that the inflows to reservoirs are normally distributed. Fletcher-Ponnambalam have extended this type of modeling for the randomized release policy (the release policy in every period depends on the previous storage which is a random variable) in a single reservoir case study [40]. Zhang and Ponnambalam [41] extended this kind of modeling to short- or medium-term operations where the initial storage volume is assumed known, and both inflow and water discharge prices are considered stochastic, to find a short-term policy

for the Lake Nipigon reservoir system in Ontario, Canada. With this technique, the probabilities of spills and deficits are embedded into the model without adding any new variables. Furthermore, in contrast to the SP method, no discretization is needed to consider the uncertainties of inflow and price. In this thesis, further explanations of this method will be presented.

2.4 Reinforcement Learning (RL) methods

The intuition behind these methods is to combine simulation with optimization in a way that they become adaptive and model-free. It means that these methods can start without any knowledge of the environment and can cope well with dynamic environments, in which the stochastic features of random variables are changing. In practice, this means that there is no need for transition probability matrix as required in the SDP method
Using Reinforcement Learning (RL) as a problem-solver dates back to the years of cybernetics and associated work in the field of psychology and statistics. In fact, the core idea of what we today call RL in engineering applications has been derived from Thorndike's work [42], which involved studying the behavior of animals in the field of psychology. He believed that actions followed by suitable results in a specific situation create a good experience for the animal, so that if this situation is repeated, it has a higher tendency to choose them. RL is actually looking for best behaviors in different situations of a dynamic system, through interacting with the environment without any explicit teacher. In other words, RL is a solution for optimal control processes in which the agent or decision-maker wants to find an optimal policy. Because the RL technique can perform using simulation, some researchers have chosen other names for RL: Simulation-based dynamic programming [18], neuro-dynamic programming [43]. Further details can be found in related books and papers [18, 43, 44, 45, 46].
Lamond and Boukhtouta [47] applied Neuro-Dynamic Programing (NDP) in a single reservoir case study for generating hydropower. They demonstrated that CPU time for the NDP method is less than the respective time for dynamic programming. Mahootchi et al. [48] applied Q-Learning as one of the RL techniques to a single-reservoir management.

Lee [49] and Lee and Labadie [50] use this learning technique to tackle a two-reservoir application in which K-means clustering is used for variables' discretization. They used all possible actions as a set of admissible actions in a two-reservoir case study. This might make the applicability of this learning method inefficient, because some actions in some states are practically impossible. Moreover, it could be computationally expensive for multi-reservoir applications. Improvement to RL techniques will be presented later in this thesis.

2.5 Important RL concepts and techniques

In the following subsections, some important concepts, in addition to four different learning methods, are described.

2.5.1 RL and supervised learning

The learning in RL is compared to other types of learning techniques, such as supervised learning, in which the learning process is based on an external teacher. In many studies, the differences between supervised and reinforcement learning has been blurred [51, 52]; Klopf [53] criticized supervised learning. He believed that adaptive behavior is being missed in supervised learning. Given this idea, Barto and Sutton [54] specified a clear boundary between these two types of learning. In general, in supervised learning, the agent tries to create the desired output in each situation; however, in RL, there are no desired outputs, and the agent selects an action from admissible actions through comparing their consequences.

2.5.2 RL components

There are four main components in any RL system: policy, reward, value function, and model. A model is optional but depends on the situations as discussed later. The policy is a mapping from state to the decision to be made. The reward is the immediate or

delayed response of the environment to the action taken by the agent. The value function, which is defined for an action-state pair, takes into account the accumulative reward from the starting point of RL. In other words, the value function, in contrast to the reward function, specifies the gain of the system for a given action-state pair after a long run. Indeed, the value functions can be calculated using the reward function accumulated by a discount factor. They can be also available either analytically or through simulation. The model component of RL determines the next state and the reward of the environment based on a mathematical function. Figure 2.3 illustrates a schematic perspective of RL.

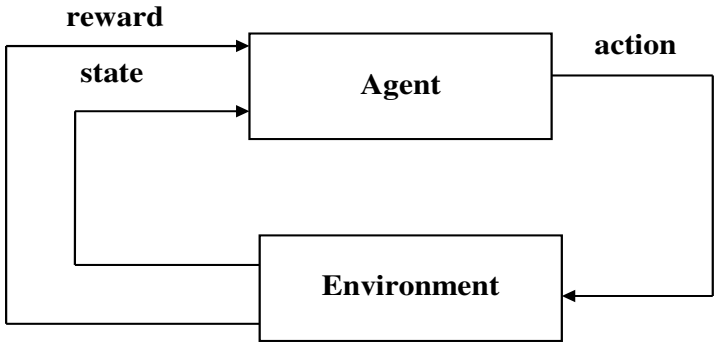

Figure 2.3: The schematic view of Reinforcement Learning

2.5.3 Policies for taking action

The policy of taking action is actually a trade-off between exploration (taking an action randomly) and exploitation (taking the best action), which leads to four common action policies in the literature: random, greedy, ϵ-greedy, and Softmax [18, 45].

In the random policy, there is no action preference. This means that the probability of choosing any action is the same and equals to $\frac{1}{|A(i)|}$, where $A(i)$ is the set of admissible actions for state i, and $|A(i)|$ is denoted as the number of actions in this set.

In the greedy policy, the agent has to pick the best one among all admissible actions in each iteration, with respect to the last estimate of the action-value function for all admissible actions in the respective state.

In the ϵ-greedy policy, greedy actions are chosen most of the time; however, once in a

while, the agent tries to choose an action in the set of admissible actions with probability of $\epsilon/|A(i)|$ where ϵ is the probability of taking non-greedy actions.
In contrast to the greedy policy, the softmax policy derived from the Boltzman's distribution can be defined in which the proportion of exploration versus exploitation changes as the process of learning continues. In this policy, the probabilities of choosing actions in every iteration are based on the following formula:

$$P(a) = \frac{e^{Q(i,a)/\tau}}{\sum_{b \in |A(i)|} e^{Q(i,b)/\tau}}, \tag{2.3}$$

where τ is a positive number called temperature and $Q(i, a)$ is the action-value function for a given state i when the action a is taken. When the temperature is high, the probabilities of taking all actions are the same, while a low temperature causes higher probabilities of actions with high action-value functions.

2.5.4 Q-Learning method

Q-Learning has been derived from the formulation of Stochastic Dynamic programming (SDP) [18, 32, 44]. The value function in SDP is substituted with an action-value function, which is a value defined for every pair of action-states, instead of every state in the value function. This method uses the Robbins-Monro algorithm [55] to estimate action-value functions pertinent to all possible pairs of action-state in a step-by-step fashion after realizing sufficient observations [43].
The criterion, which the agent has to consider for taking action, is the accumulated rewards or the value function. The value function here has the same meaning as in the conventional method of SDP with two main parts: immediate reward and accumulated reward. Because in RL the learning process is implemented by direct interaction with the environment, value functions have to be updated after each interaction. This is similar to asynchronous dynamic programming, in which the latest updated value functions of states can be considered for updating the value functions of other states in the same iteration.
To find an updated value function based on equation 2.1, all admissible actions in the

respective state and period must be tested with respect to this equation, and the best value is chosen as a new value function. Therefore, we can introduce another terminology called action-value function, $Q(i,a)$. This value demonstrates the expected accumulated reward when a decision maker starts from state i and takes action a. Using these new values, the formulation of SDP in Equation 2.1 can be written as follows [32, 45]:

$$\begin{aligned} Q^t(i,a) &= Re_i^t(a) + \gamma \sum_j P_{(ij)}^t(a) Q^{t+1}(j,b^*), \\ &= \sum_j p_{ij}^t(a) \times \left(r_{ij}^t(a) + \gamma \times \max_{b \in A(j)} Q^{t+1}(j,b) \right), \\ &= E[r_{ij}^t(a) + \gamma \max_{b \in A(j)} Q^{t+1}(j,b)], \end{aligned} \tag{2.4}$$

where E is the expectation operator, $r_{ij}^t(a)$ is the immediate reward in period t when the action a is taken for transition from state i to state j, and b and b^* are admissible actions and the best one with respect to the next state j, respectively. This is the same formulation for SDP, except that it is expressed in terms of the action-value function.
Let us explain the algorithm of Robbins-Monro [55] for calculating the average of a sequence of data iteratively. Suppose we have a sequence of data X_1, X_2, X_3, $\cdots$ X_n, X_{n+1}. If the average of the first n input is denoted by $\bar{X}_n$, and the $(n+1)^{th}$ observation is X_{n+1}, the average of the new sequence of data in terms of the current average and the new occurrence is given by:

$$\bar{X}_{n+1} = \bar{X}_n + \frac{1}{n+1} \times (X_{n+1} - \bar{X}_n) = \frac{\sum_{i=1}^{n+1} X_i}{n+1}. \tag{2.5}$$

Based on this rule, it is simple to find a new formulation as follows:

$$Q^t(i,a) := Q^t(i,a) + \tfrac{1}{NOV(i,a)} \times [r_{ij}^t(a) + \gamma \max_{b \in A(j)} Q^{t+1}(j,b) - Q^t(i,a)], \tag{2.6}$$

where $NOV(i,a)$ is the number of observations of the action-state pair (i, a). We can

also substitute $\frac{1}{NOV(i,a)}$ with α, called the learning parameter:

$$Q^t(i,a) \;:=\; Q^t(i,a) \;+\; \alpha \;\times\; [r_{ij}^t(a) \;+\; \gamma \max_{b \in A(j)} Q^{t+1}(j,b) \;-\; Q^t(i,a)]. \tag{2.7}$$

This is called the Q-Learning method, developed by Watkins [44], in which the transition probabilities are removed. In other words, this is a model-free algorithm in which the transition probabilities are not used for updating the action values. Moreover, Q-Learning is an off-policy method in RL in which the behavior policy, the policy which is used to generate behavior during the learning process, is different from the estimation policy, the policy which is updated or evaluated and supposed to be considered as an operating policy in the real-time decision-making [45]. It should be noted that one iteration in Q-Learning and other learning methods explained later starts from an action-state pair (i,a) and obtains a new action-state pair (j,b) from where the next iteration will start. Moreover, an episode comprises a certain number of iterations that should be determined before the learning process is started.

2.5.5 Sarsa method

Sarsa is an on-policy Temporal-Difference (TD) learning technique in RL, which means that the same action policy π used for taking action in the current state i (the policy that generates behavior) is used for choosing the action in the next state j in order to update the respective action-value function in one iteration [45]. Therefore, there is a transition from one action-state pair (i,a) to another action-state pair (j,a') in every iteration. However, in Q-learning the transition is defined from one action-state pair (i,a) to the next state j in which the greedy action is chosen for updating $Q^t(i,a)$. The general formulation of sarsa is given as follows:

$$Q^t(i,a) \;\;:=\;\; (1-\alpha)Q^t(i,a) + \alpha[r_{ij}^t(a) \;+\gamma Q^{t+1}(j,a')], \tag{2.8}$$

where a and a' are two actions chosen from the action policy π.

2.5.6 Sarsa(λ) method

Sarsa(λ), which is an on-policy learning technique like sarsa, uses an eligibility trace (The trace forms the memory parameters pertinent to eligible action-state pairs whose corresponding action-value functions can be updated [45].) The parameter λ, along with the discount factor γ, plays a decaying role in the eligibility trace to put more weight on those states which have been recently visited. The eligibility trace as a function of state, s, and action, a, should be incremented for the state visited and the action taken in the current iteration, and updated with $\gamma\lambda$ for all action-state pairs after editing the action-value functions. This process of learning should be iteratively performed using these eligibility traces, and continued until the convergence criteria are satisfied [45]. Analogous to sarsa, this technique uses a quintuple consisting of an action a, in current state i, observing reward r, and next state j, and taking another action a'. The initial values of action-value functions, $Q(i, a)$, and eligibility traces, $e(i, a)$, should be set to zero at the start of the learning. One iteration of the sarsa method is demonstrated in Algorithm 1 [45].

2.5.7 Q(λ) method

Q(λ) is an off-policy RL method in which the behavior, or the exploratory policy, is different from the estimation policy, which is evaluated or improved during the learning process [44]. The value of eligibility traces are updated if the next action is the same as that obtained by the greedy policy; otherwise, the updating process is only performed for the action taken in that iteration, as in Q-Learning, and the eligibility traces for all action-state pairs set to zero. In other words, the eligibility traces are exponentially smoothed in every iteration except when the next action is not greedy in which all eligibility traces set to zero

Algorithm 1 One iteration of sarsa(λ) for the action-state pair (i, a)

1: find Temporal-Difference (TD) error as

$$\Delta = r_{ij}^t(a) + \gamma Q^t(j, a') - Q^t(i, a)$$

2: set the eligibility trace for this pair

$$e(i, a) = e(i, a) + 1$$

3: perform for all action-state pairs (l, b) ($\forall \;\; l \in \{\text{all possible states}\} \;\; \& \;\; b \in A(l)$)

3-1) update $Q^t(l, b)$

$$Q^t(l, b) := Q^t(l, b) + \alpha \times \Delta \times e(l, b)$$

3-2) update eligibility traces

$$e(l, b) = \gamma \times \lambda \times e(l, b)$$

4: $(i, a) \;\; \leftarrow \;\; (j, a')$

2.6 Opposition-Based Learning (OBL)

Opposition-Based Learning (OBL) schemes, which were introduced by Tizhoosh [56, 57], could be suitable approaches to speed up the process of learning in RL. It has been also shown that using this scheme in, for example, Genetic Algorithms (GA), Neural Networks (NN), and Reinforcement Learning (RL), can generally speed up the training process [56, 58, 59, 60]. The intuition behind this learning methodology is to use the inherent oppositional relationships in the system to update the agent's knowledge more frequently. However, the efficiency of OBL is generally problem-dependent.

In most search algorithms, such as GA and learning techniques such as NN, we usually start from a single or a set of random points in the domain of input variables, and continue with some other operations based on a specific criterion until an optimal or near-optimal solution has been found. If the starting points are close enough to a optimal solution, we might converge to the optimum within a few iterations; however, finding a good starting solution is hard. A higher rate of convergence in these applications is essential, and an

ongoing research area in literature. The opposition corresponding to a point could be defined in two different ways: type I and type II [61]. In type I, the opposite point is calculated based on lower and upper bounds of the search interval:

$$\breve{y} = y^{max} + y^{min} - y, \tag{2.9}$$

where $\mathbf{y}=\{y_1, y_2, \cdots y_m\}$, is a vector of m input features, $\mathbf{y}^{max}=\{y_1^{max}, y_2^{max}, \cdots yx_m^{max}\}$ and $\mathbf{y}^{min}=\{y_1^{min}, y_2^{min}, \cdots y_m^{min}\}$ are two vectors of corresponding lower and upper bounds for each feature respectively, and $\breve{\mathbf{y}}$ is the opposite vector of $\mathbf{y}$. Suppose we are going to find the optimum solution for a monotonic function $f(\cdot)$ in one dimensional problem and a specific domain by two different approaches: 1) take a random point y_1 and evaluate it along with its opposite, $\breve{y}_1$, simultaneously using the respective evaluation function; 2) take the second random point y_2 and compare its evaluation with the evaluation of the first random point in every iteration. Based on the theorem called Central Opposition for a one-dimensional space and proved by Rahnamayan et al. [62], the probability of closeness of the opposite $\breve{y}_1$ to the optimal solution y_o (assuming that there is at least one optimal solution) is higher than the probability pertinent to the second random variable y_2 (all three variables y_1, y_2, and $\breve{y}_1$ are considered simultaneously).

$$Pr\left(|\breve{y}_1 - y_o| < |y_2 - y_o|\right) > Pr\left(|y_2 - y_o| < |\breve{y}_1 - y_o|\right). \tag{2.10}$$

This mathematical result demonstrates that type I opposition may guide the agent or learner to reach the optimal solution faster. The authors [62] also provide experimental results confirming the theorem.

Another version of the opposition, called the type II opposition scheme, utilizes the value of the objective function or approximation functions to find the opposite point [61]. In other words, in contrast to the type I scheme in which the opposites are calculated using the variable domain y, in type II, they are computed via the evaluation of function $f(y)$. Moreover, to enjoy the advantage of Type II opposition-based learning, function $f(y)$ is not needed to be monotonic. The way of finding opposite action is mathematically given in the following equation [61]:

$$\breve{y} \in \{\breve{y}_i | \breve{f}(\breve{y}_i) = \max_j \breve{f}(y_j) + \min_j \breve{f}(y_j) - \breve{f}(y_i)\}, \tag{2.11}$$

where $\check{f}(y)$ is the value function or the approximate value function in point y, and $min\check{f}(y)$ and $max\check{f}(y)$ are the lower and upper bounds of the corresponding function, respectively. For example, at the early stage of the learning process, there are only a few approximations, which might not be accurate enough to be considered as value functions. However, Equation 2.11 assists the agent to use this partial knowledge to extract the opposites. It should be noted that as the learning continues, the agent might find different opposites for a similar situation.

2.7 Risk consideration in RL techniques

In a stochastic environment where the Markovian assumption is reasonable, the main target in all learning methods is usually to maximize or minimize the expected value of a predefined objective function; however, this decision-making does not consider risk [63]. Bessa [64] developed a method by using two-pass SDP, in which the risk is considered. Heger [63] presented an expected value-variance criterion to consider this fact in the objective function as a risk minimization technique in SDP, where all transition probabilities are known. He developed a method similar to Q-Learning called $\hat{Q}$-Learning which used a *minmax* criterion to find the action-value functions. The risk in $\hat{Q}$-Learning is much more important than the expected value criterion, such that the optimal policy would be a conservative policy at the end, which is not always favoured by decision makers. This way of considering risk is called the *worst-case control*, and its respective stationary policy is called a risk-avoiding policy [65], which means that the agent considers the most unlikely accumulated reward for determining the action-value function. It usually results in a very pessimistic average of benefit or cost at the end of the learning process [65].
There are other techniques, called *risk-sensitive control*, in which the risk is embedded into the objective function. The ccBeta method which is presented by Pendrith and Ryan [66] falls into this category.
Neuneier and Mihatsch [65] presented another risk-sensitive control framework. They created a new variable called χ^k for weighting the accumulated reward in different iterations.

Variance-Penalized RL is another way of considering risk in the RL methods [67, 68]. Geibel [67] also uses a risk-penalized method and represents its applicability in a grid world problem. In this technique, a new variance term with a risk-aversion parameter should be added to the standard formulation of the action-value function in different RL methods. The variance, like the other expected-value term, should be trained during the learning process. Sato and Kobayashi [68] developed a method for training the variance of action-value functions in finance applications as a penalty term in the objective function. The optimality equations cannot be applied for this new objective function and some appropriate assessment should be performed to determine a better policy in each episode or iteration (e.g., using simulation). This technique, like other risk-sensitive techniques, is used for finance applications which might make the generalization of the risk consideration limited. In this thesis, we develop an alternative method for including risk.

2.8 Two-stage Stochastic Programming (TSP)

To model a storage management problem with two stochastic variables, namely, inflow and water discharge price based on Two-stage Stochastic Programming (TSP), we should define scenarios' corresponding probabilities to cover different possible situations of these two variables in the future:

$$\begin{aligned}
&Scenario\ 1: && \acute{C}^{t,1}, && \acute{I}^{t,1} && \Rightarrow\ P^{t,1},\\
&Scenario\ 2: && \acute{C}^{t,2}, && \acute{I}^{t,2} && \Rightarrow\ P^{t,2},\\
&\quad\vdots && \quad\vdots && \quad\vdots && \quad\vdots\\
&Scenario\ N: && \acute{C}^{t,N_1}, && \acute{I}^{t,N_1} && \Rightarrow\ P^{t,N1},
\end{aligned}$$

where $\acute{C}^{t,l}$ and $\acute{I}^{t,l}$ are the price and inflow considered for the l^{th} scenario during period t, respectively, $P^{t,l}$ is the probability of l^{th} scenario of inflow in period t, and N_1 is the number of scenarios. The number of scenarios is chosen based on the sensitivity of the

solution. At the beginning, N_1, for example, is small (e.g., N_1=30) and then N_1 is increased until the solution does not change significantly.

In a single reservoir application, only the continuity constraints that include stochastic inflows are substituted with a set of new constraints based on the existing scenarios for every period t as follows:

$$s^{t+1} - s^t + u^t + SF^{t,1} - SU^{t,1} = \acute{I}^{t,1},$$

$$\begin{aligned} s^{t+1} - s^t + u^t + SF^{t,2} - SU^{t,2} &= \acute{I}^{t,2}, \\ \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots & \\ s^{t+1} - s^t + u^t + SF^{t,N_1} - SU^{t,N_1} &= \acute{I}^{t,N_1}, \end{aligned}$$

Here, u^t is the release in period t, and $SF^{t,l}$ and $SU^{t,l}$ are second-stage decisions corresponding to either shortage or surplus, respectively, that may occur for the l^{th} scenario inflow. We assume that the corresponding cost for shortfall, CSF^t, is equal to C_s (C_s is the benefit for keeping high storages, a term defined as zeroth order approximation of a power generation function, as described in [9]). The cost for surplus, CSU^t, is equal to $E(c^t)$, the expected value of price corresponding to releases. Hence, the objective function in SP is given by

$$\begin{aligned} &\max_{u^t, s^t} \{f^t(s^t, u^t)\} = \\ &\max_{u^t, s^t} \left\{ \sum_{t=1}^{T} \{E(C^t) \times u^t + C_s \times s^t - \sum_{l=1}^{N1} P_l^t (CSF^t \times SF^{t,l} + CSU^t \times SU^{t,l})\} \right\}. \end{aligned} \tag{2.12}$$

The obvious disadvantage of this method is the large increases in the size of the problem with increase in the number of random variables and periods.

2.9 Fletcher-Ponnambalam (FP) model

A new continuity equation for a single-reservoir case study based on indicator functions is given by [40]

$$\begin{aligned} s^t = \{s^{t-1} + \bar{I}^t + \eta^t - u^t\} 1_{[s^t_{min}, s^t_{max}]}(\hat{s}^t) + \{s^t_{min}\} 1_{(-\infty, s^t_{min})}(\hat{s}^t)\} \\ + \{s^t_{max}\} 1_{(s^t_{max}, \infty)}(\hat{s}^t)\}, \end{aligned} \tag{2.13}$$

where $\bar{I}^t$ is the natural mean of inflow, η^t is a zero-mean random component, and $1_{[.]}(\cdot)$ are binary indicator functions or random variables which are defined as follows:

$$\begin{aligned} &1_{[s^t_{min},\ s^t_{max}]}(\hat{s}^t) = 1 \quad \text{if} \quad s^t_{min} \le \hat{s}^t \le s^t_{max}, \\ &1_{[s^t_{min},\ s^t_{max}]}(\hat{s}^t) = 0 \quad \text{otherwise}, \end{aligned} \tag{2.14}$$

$$\begin{aligned} &1_{(-\infty,\ s^t_{min})}(\hat{s}^t) = 1 \quad \text{if} \quad -\infty < \hat{s}^t < s^t_{min}, \\ &1_{(-\infty,\ s^t_{min})}(\hat{s}^t) = 0 \quad \text{otherwise}, \end{aligned} \tag{2.15}$$

$$\begin{aligned} &1_{(s^t_{max},\ \infty)}(\hat{s}^t) = 1 \quad \text{if} \quad s^t_{max} < \hat{s}^t < \infty, \\ &1_{(s^t_{max},\ \infty)}(\hat{s}^t) = 0 \quad \text{otherwise}. \end{aligned} \tag{2.16}$$

It should be noted that Equation 2.13 can be written in a different way using minimum and maximum operator as follows:

$$s^t \ = \ min\left\{s^t_{max}, \ \ max\left(s^t_{min}, \ \ s^{t-1} + I^t + \eta^t - u^t\right)\right\}. \tag{2.17}$$

Conceptually, there is no difference between the equations 2.13 and 2.17; however, the expected values of indicator functions in the Equation 2.13 represent the probabilities of containment, shortage, and spill, respectively, and this form is used in this thesis
By taking expectations of Equation 2.13, using the stochastic release policy ($u^t = k^t + s^{t-1}$ in which k^t is the deterministic part of release in period t), the first and second moments of storage (given below) are substituted for mass balance and storage boundaries constraints in the reservoir model. This model can be solved by constrained nonlinear optimization

algorithms. These new constraints based on the use of normal distribution are as follows [40]:

$$\begin{aligned} E(s^t) &= (I^t - k^t) \cdot \left\{ \tfrac{1}{2} \left[erf\left(\tfrac{[s^t_{max} - (\bar{I}^t - k^t)]}{[2Var(\eta^t)]^{1/2}} \right) - erf\left(\tfrac{[s^t_{min} - (I^t - k^t)]}{[2Var(\eta^t)]^{1/2}} \right) \right] \right\} \\ &- (\tfrac{[Var(\eta^t)]^{(1/2)}}{\sqrt{2\pi}}) \cdot \left\{ \exp\left(-\tfrac{1}{2} \tfrac{[s^t_{max} - (\bar{I}^t - k^t)]^2}{Var(\eta^t)} \right) - \exp\left(-\tfrac{1}{2} \tfrac{[s^t_{min} - (\bar{I}^t - k^t)]^2}{Var(\eta^t)} \right) \right\} \\ &+ (s^t_{min}) \cdot \left\{ \tfrac{1}{2} \left[1 + erf\left(\tfrac{[s^t_{min} - (\bar{I}^t - k^t)]}{[2Var(\eta^t)]^{1/2}} \right) \right] \right\} \\ &+ (s^t_{max}) \cdot \left\{ \tfrac{1}{2} \left[1 - erf\left(\tfrac{[s^t_{max} - (\bar{I}^t - k^t)]}{[2Var(\eta^t)]^{1/2}} \right) \right] \right\}, \end{aligned} \tag{2.18}$$

where the term in the first set of curly brackets estimates the probability of containment, the term in the third set of brackets estimates the probability of deficit, and the term in the fourth set of brackets estimates the probability of spill.

$$\begin{aligned} E(s^t_i)^2 = {} & [(\bar{I}^t - k^t)^2 - Var(\eta^t)] \cdot \left\{ \tfrac{1}{2} \left[\; erf\left(\tfrac{[s^t_{max} - (\bar{I}^t - k^t)]}{[2Var(\eta^t)]^{1/2}} \right) - erf\left(\tfrac{[s^t_{min} - (\bar{I}^t - k^t)]}{[2Var(\eta^t)]^{1/2}} \right) \; \right] \right\} \\ & + 2(\bar{I}^t - k^t) \cdot (\tfrac{-[Var(\eta^t)]^{(1/2)}}{\sqrt{2\pi}}) \left\{ \exp\left(-\tfrac{1}{2} \tfrac{[s^t_{max} - (\bar{I}^t - k^t)]^2}{Var(\eta^t)} \right) - \exp\left(-\tfrac{1}{2} \tfrac{[s^t_{min} - (\bar{I}^t - k^t)]^2}{Var(\eta^t_i)} \right) \right\} \\ & - \tfrac{[Var(\eta^t)]^{(1/2)}}{\sqrt{2\pi}}) \cdot \left\{ [s^t_{max} - (\bar{I}^t - k^t)] \cdot \exp\left(-\tfrac{1}{2} \tfrac{[s^t_{max} - (\bar{I}^t - k^t)]^2}{Var(\eta^t)} \right) \right. \\ & \left. - [s^t_{min} - (\bar{I}^t - k^t)] \cdot \exp\left(-\tfrac{1}{2} \tfrac{[s^t_{min} - (\bar{I}^t - k^t)]^2}{Var(\eta^t)} \right) \right\} \\ & + (s^t_{min})^2 \cdot \left\{ \tfrac{1}{2} \left[1 + erf\left(\tfrac{[s^t_{min} - (\bar{I}^t - k^t)]}{[2Var(\eta^t)]^{1/2}} \right) \right] \right\} + (s^t_{max})^2 \cdot \left\{ \tfrac{1}{2} \left[1 - erf\left(\tfrac{[s^t_{max} - (\bar{I}^t - k^t)]}{[2Var(\eta^t)]^{1/2}} \right) \right] \right\}, \end{aligned} \tag{2.19}$$

Here, erf is the error function which can be computed by the following equation:

$$\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt. \tag{2.20}$$

Since both stochastic variables, namely the price (explicitly) and the inflow (implicitly via storage volume), have influence on determining the value of objective function in FP,

the objective function can be presented as follows [41]:

$$\max_{E(s^t),u^t} \left\{ f^t(E(s^t),u^t) \right\} = \max_{E(s^t),u^t} \left\{ \sum_{t=1}^{T} (E(C^t) \times u^t + C_s \times E(s^t)) \right\}. \quad (2.21)$$

In this thesis, extension of this method for multi-storage systems will be presented.

2.10 Warehousing problem

The purpose of the warehousing optimization problem is to achieve an optimal order policy, the quantity of products to be ordered or bought from a supplier outside of the defined system or other storage spaces, such that it minimizes the total cost.
This is analogous to what is considered in reservoir management; however, the following interpretations should be established:

1. The decision in reservoir management is defined as the amount of water to be released that should be released; however, it is the quantity of items to be ordered in the warehousing problem. In other words, the decision is an output entity with a negative sign in the mass balance equation for reservoir applications, but it is an input entity with a positive sign in the respective warehousing balance equation.

2. Stockouts in warehouse can be interpreted as spillage in reservoir management. It can be manipulated by decision makers to be returned to suppliers or conducted to other storage spaces (other warehouses); however, the spillage in reservoir application is out of control which might be considered as available water in other reservoirs or cause some losses due to floods downstream.

3. Lost-sale in warehousing process is equivalent to shortage in reservoir management. The release in this situation should be revised with respect to the boundary condition of minimum storage level; however, the ordered quantity in warehousing does not change when the lost-sale situation happens.

4. The set of reservoirs in a configuration are usually all the same: that is, they can receive stochastic inflows from sources outside the defined system, and can potentially satisfy the external demands (i.e., the demands from outside of the defined system) in their downstream. However, in the warehousing problem, there exist distribution centers (usually called warehouses in the literature) which can be fully connected to each other. These storage centers obtain the ordered quantity from some external sources (suppliers) or other distribution centers. They satisfy the demands for smaller storage spaces called retailers or depots in their downstream (i.e., there is usually no connection between retailers and suppliers). In some problems, transshipments between retailers are allowed (e.g., a retailer can sell stockouts or its existing products to other retailers). Moreover, the stochastic demands that should be characterized (i.e., finding the distribution function, mean, or standard deviation) for the optimization model are specified in the lowest level (retailers level). However, the features of the random demands for distribution centers should be calculated in terms of demand at the retailers, which are physically connected to these distribution centers.

We only focus on the literature review of multi-echelon warehousing management problems in this section. Chen and Ghosh [69] introduced Hierarchical, Distributed, Dynamic, Inventory (HDDI) management based on simulation techniques in which there is no limitation for the number of warehouses and retailers. The expected values of the normally-distributed retailer levels are considered as actual demands. In this system, there are bi-directional flows between all neighbor retailers and one single warehouse, and orders from other warehouses only occur in the emergency (shortage) situations. This kind of simulation could be very complex and time consuming where the number of entities are enormous. Gupta and Maranas [70] developed deterministic and stochastic tactical logistics models for decision making in a intermediate time-step (e.g., between 1 and 2 years) to satisfy the demands for inventory management. They believe that considering all parameters as constant values is too far from reality. Moreover, the optimal decision policy obtained through these optimization models cannot respond well to what happens

in the real world, which is usually dynamic and stochastic. They divided the whole tactical stochastic models into two main categories: scenario-based approach (implicit) and distribution-based approach (explicit). They used the first approach under a normality assumption to model the stochastic situation in order to find the optimal operational policy in supply chain networks composed of six different entities or sites (warehouses or plants). Kljajic et al. [3] applied a simulation technique by considering several different strategies in a real-world case study to find the optimal operational strategy. The historical data which are supposed to be a suitable representation for real situation are used to generate sufficient samples for implementing the simulation. Chen and Lee [71] models a warehousing scheduling problem with a multi-objective function in which both demand and price are stochastic variables. The scenario-based approach is used to handle the stochasticity of demand and fuzzy variables are set for prices. The ultimate model is a mixed-integer nonlinear programming problem with normally-distributed stochastic variables. Monthatipkul and Yenradee [6] developed a mixed-integer programing model for a supply chain management application with one warehouse and multiple identical retailers, called Inventory/Distribution Plan, in which the mean and the standard deviation of demand are used in constructing the stochastic models. In this formulation, the constraint of safe stock using the demand variance is a crucial factor. The models developed later on also consider this aspect.

2.11 Summary

In this chapter, relevant research articles, including a few different optimization or simulation techniques applied for multi-reservoir or multi-echelon warehousing problems, have been reviewed. Many similar methods are utilized to tackle both storage applications in the stochastic situation in which some of them relied on simplifications of stochastic variables (e.g., using only the mean of stochastic variables to cover the stochasticity). Moreover, most of these techniques cannot cope well in large-scale real-world problems. In the next chapter, we introduce improvements to RL techniques and new nonlinear models for storage management problems.

Chapter 3

Simulation-Based Methods and Nonlinear Models

3.1 Introduction

In this chapter, we will introduce selected Reinforcement Learning (RL) techniques to be used in storage management problems. Furthermore, several nonlinear models which have been originally developed by Fletcher and Ponnambalam [16] are extended for optimal operations optimization of single and multi-reservoir systems. The Gaussianity assumption in these models is relaxed and two different stochastic release policies are proposed.

3.2 RL techniques for storage management

In the following subsections, we will explain how to apply RL techniques in single- and multi-storage management applications.

3.2.1 Taking actions and making decisions

The constraints in the optimization model for storage management, including boundary conditions for storages and releases explained in section 1.2, play the key role in determining the set of admissible actions and the possible states. States are usually the storage volumes of reservoir, or the current stock in a warehousing problem, at the beginning of each time period. When the agent takes an action a^t as a target amount of water or items to be ordered from the set of admissible actions using an action policy (e.g., ϵ-greedy), it should wait to receive some signals as feedback from the environment. These signals are the immediate reward and the next storage volume. The reward is determined by the reward function which is usually given as a part of the problem definition. However, the next state, given s^t, will be determined using the balance equation in a single reservoir application as:

$$s^{t+1} = s^t + I^t - \nu^t - a^t. \tag{3.1}$$

Inflow to reservoir, I^t, and the evaporation, ν^t, from it are random variables, and hence the next period storage value is random. It is clear that the agent takes an action only based on the current state, without having any information about the future. Therefore, the amount of release to be chosen or the amount of items to be ordered is not necessarily the same thing that actually occurs in reality. To make a distinction between these two concepts, the following definitions are assumed:

1. Release taken ($a = u$): the amount of water to be released
2. Release occurred (ϑ): the release that actually occurs due to physical constraints

As the environment is stochastic, the respective immediate reward, which is usually calculated based on the release which occurred and the end-of-period storage, might vary for the same action-state pair observed in different iterations.

3.2.2 Admissible actions

In multireservoir problems, for each time period, agent should choose an action among candidate actions, which are called possible actions. In other words, possible actions may be defined as a set of all discrete values of the control variable (release or the amount of items to be ordered) which are within the range between minimum and maximum release volumes (demands) for reservoir i, ($R^t_{i,min}$ and $R^t_{i,max}$). Given the maximum number of discrete release volumes in each period for every reservoir i as G_i, one may determine the set of discrete possible actions as follows:

$$\text{All possible actions} = \left\{ a^t_{i,g_i} | a^t_i = R^t_{min} + \frac{(R^t_{max} - R^t_{min})}{G_i - 1}(g_i - 1) \right\}, \quad \text{for } i = 1, \cdots, N \quad \text{and} \quad g_i = 1, \cdots, G_i, \tag{3.2}$$

where a^t_{i,g_i} is the $g_i th$ possible action in reservoir i in period t. In the case of having multiple outlets from one storage, possible actions for each outlet should be separately determined with respect to the maximum and minimum release volumes (demand) of that outlet.

The agent may pick an action from this set at each time period based on one action policy. In problems with small action-state spaces, extended simulation helps to explore all possible actions at different states. However, the convergence may be quite slow and perhaps intractable in multi-reservoir systems with numerous state variables. Furthermore, some of those actions are physically infeasible. In order to eliminate these infeasible actions from the set of admissible actions and to make this set more compact, they can be determined when inflow (demand) is fixed in the balance equation (Equation 1.2), for example using the historical data. Depending on whether the lower or upper bound of the inflow (demand) bound is active, the set of admissible actions associated with any storage volume (stock) may fall into two types: 1) optimistic (e.g., considering maximum inflow and minimum evaporation in Equation 1.2), or 2) pessimistic (e.g., considering minimum inflow and maximum evaporation in Equation 1.2). In multi-reservoir applications, in addition to precipitation and runoff flowing to each reservoir, the releases from other reservoirs should be considered. Algorithm 2 demonstrates the process of

finding the admissible action in these two schemes in multi-reservoir cases which has to be applied from the most upstream reservoir to the downstream ones.

3.2.3 Optimal policy and updating action-value function

An agent takes an action using an action policy which is built on action-value functions. After receiving the reward from the environment, the agent updates the action-value function related to the previous state and takes a new action. For the starting point, arbitrary values should be considered for all action-value functions $Q(i,a)$ (e.g., $Q(i,a) = 0$, for all states i, and actions a). In each iteration at least one value should be updated. Assuming that the agent observes storage $s^t = i$ and takes action a^t, based on the balance equation (Equation 1.2) and the value of the stochastic variables, the next storage becomes $s^{t+1} = j$ (in the case of violating the maximum or the minimum level of storage volume in the next period, $j = s_{max}^{t+1}$ or $j = s_{min}^{t+1}$). The action-value functions are updated as follows:

$$Q^t(i,a) := Q^t(i,a) + \alpha \ \times [r_{ij}^t(a) + \gamma \max_{b \in A(j)} Q^{t+1}(j,b) - Q^t(i,a)] \tag{3.3}$$

where $r_{ij}^t(a)$ is the immediate reward when action a is taken for transition from state i to state j in period t.

Although the transition from one storage to another is continuous and is controlled by the balance equation (Equation 1.2), there is no guarantee that the current or next storage is exactly one of the discrete values. An easy way to tackle this problem is to find the closest discrete value to the current or next storage. However, this seems to be error-prone because the respective storage only partially belongs to this state. To increase the accuracy, we might use a linear or nonlinear interpolation. To interpolate linearly, firstly, we have to find two successive values i and $i+1$, which are the closest discrete values to the current storage. Finally, the degrees of closeness of the storage to these boundaries can be simply computed (e.g., w_1 and w_2 in a single-reservoir case study such that w_1+w_2=1). Algorithm 3 demonstrates the process of updating action-value functions in a single reservoir case study, which is easily extendable to multi-reservoir cases.

Algorithm 2 : Finding admissible actions in reservoir management

1: Compute minimum and maximum inflow, $I^t_{i,min}$ and $I^t_{i,max}$, and minimum and maximum evaporation, $\nu^t_{i,max}$ and $\nu^t_{i,min}$ for all reservoirs ($i = 1 \cdots N$)

$$\text{pessimistic} \begin{cases} P(I^t_i \le I^t_{i,min}) &= \epsilon_1, \\ P(\nu^t_i \le \nu^t_{i,max}) &= 1 - \epsilon_2, \end{cases}$$

$$\text{optimistic} \begin{cases} P(I^t_i \le I^t_{i,max}) &= 1 - \epsilon_1, \\ P(\nu^t_i \le \nu^t_{i,min}) &= \epsilon_2, \end{cases}$$

where ϵ_1 and ϵ_2 are two parameters in [0,1] which have to be determined before the learning process is started

2: Find all possible actions using Equation 3.2

3: In an optimistic scenario, find the end-of-period storages using all possible actions

$$\begin{cases} s^{t+1}_i &= \bar{s}^t_{i,j_i} + I^t_{i,max} - \nu^t_{i,min} - a^t_{i,g_i} + \sum_{l=1,l\neq i}^{N} u^{*t}_{li} \times \delta_{li}, \\ u^{*t}_{li} &= max\{a^t_{l,g_{i'}} | s^{t+1}_l \ge s^{t+1}_{l,min},\ \bar{s}^t_{l,j_l}\}, \\ & \quad \text{for } i = 1, \cdots, N \ \ , \ \ j_i = 1, \cdots, J_i, \ \ g_i = 1, \cdots, G_i \end{cases}$$

where $\bar{s}^t_{i,j_i}$ is j^{th}_i discrete value of storage in reservoir i, J_i is the number of discretizations for storage level in reservoir i, u^{*t}_{li} is the maximum release flowing to reservoir i from reservoir l in period t, and s^{t+1}_i is the end-of-period storage in reservoir i.

4: In a optimistic scenario, find admissible actions

$$\begin{aligned} A(i, j_i) &= \{a^t_{i,g_i} | s^{t+1}_i \ge s^{t+1}_{i,min}, \bar{s}^t_{i,j_i}\}, \\ & \quad \text{for } i = 1 \ \cdots \ N, \ \ j_i \ = \ 1 \cdots, J_i, \ \ g_i \ = \ 1 \ \cdots G_i. \end{aligned}$$

5: In a pessimistic scenario, perform steps 3, 4 after changing $I^t_{i,max}$ and $\nu^t_{i,min}$ to $I^t_{i,min}$ and $\nu^t_{i,max}$; respectively, and change maximization to minimization for finding $u^{*t}_{l,i}$.

Algorithm 3 : Updating action-value functions using interpolation (single-reservoir)

1: Take the closest discrete value of storage to the actual current storage s_i^t during period t

2: Compute the next storage volume using the balance equation (Equation 1.2) and other boundary constraints

3: Find two successive discrete values which are the closest values to the next storage (j and $j+1$)

4: Find w_1 and w_2 using using proper interpolation for s^{t+1}

5: Perform interpolation for the next storage

$$\begin{aligned} Q_z^t &= Q^t(i,a) + \alpha[r_{ij}^t(a) + \gamma \max_{b \in A(j')} Q^{t+1}(j',b) - Q^t(i,a)], \\ & \quad j' = (j-1) + z \ \ \& \ \ z = 1, 2 \end{aligned}$$

6: Update the action-value function

$$Q^t(i,a) = \sum_{z=1}^{2} w_z \times Q_z^t.$$

7: Perform interpolation for the current storage s^t. Consider i and $i+1$ as two consecutive boundaries for the current storage and w_1' and w_2' as proper weights

$$\begin{aligned} Q_{zz'}^t &= Q^t(i',a) + \alpha w_{z'}'[r_{ij}^t(a) + \gamma \max_{b \in A(j')} Q^{t+1}(j',b) - Q^t(i',a)], \\ & \quad i' = (i-1) + z', \ \ j' = (j-1) + z, \ \ z = 1,\ 2 \ \ \& \ \ z' = 1,\ 2. \end{aligned}$$

8: Update action-value function

$$Q^t(i',a) = \sum_{z=1}^{2} w_z \times Q_{zz'}^t \qquad \text{for} \ \ z' = 1, 2.$$

As seen in Algorithm 3, because the new learning rate ($\alpha \times w'$) is smaller than α, it might slow the learning process; however, it is likely to lead to more accurate results than the situation in which only one action-value function is updated.

3.3 Opposition in storage management

RL techniques are some learning approaches in which a single or multiple agents could be trained through interaction with stochastic or deterministic environments, such that an optimal or near-optimal policy can be extracted. The most advantageous aspect of these techniques is their model-free basis which makes them very attractive and useful in real-world and on-line training applications. However, to converge to a steady state [44], all states and actions should be infinitely (sufficiently) visited. This may take too much time in real-world applications. Therefore, the question may arise how to achieve an optimal solution with fewer interactions. Different opposition-based learning schemes might be a useful answer to this question.

3.3.1 Opposite action/state in RL techniques

The following assumptions for applying the Opposition-Based Learning (OBL) are evident:

- the immediate or delayed reward is usually extracted from the problem definition and called a reward function.
- the transition function (balance equation in 1.2) in which some of components are random variables is given as a priori system knowledge.
- there is no dependency between the action taken and the values realized for stochastic parameters in every iteration. This means that the agent is capable of determining what happens (meaning what is the next state) if it chooses a different action

(e.g., opposite action) after realizing the values of the respective random variables in every iteration.

In RL techniques in which the updating process is performed based on the action taken and the state (current or next state), the opposite could be defined for these two elements. Two kinds of opposition can be considered for opposite action $\breve{a}$ or opposite state $\breve{s}$ as follows [57]:

1. **Type I opposite:** Opposite action $\breve{a}$ and opposite state $\breve{s}$ are defined based on their corresponding domain boundaries

$$\breve{a} = a_{max} + a_{min} - a, \tag{3.4}$$

$$\breve{s} = s_{max} + s_{min} - s, \tag{3.5}$$

where s_{min}, s_{max} are the minimum and maximum value of state variable, and a_{min} and a_{max} are the minimum and maximum value of admissible action, respectively.

2. **Type II opposite:** Opposite action $\breve{a}$ and opposite state $\breve{s}$ are defined using the action-value functions [57]:

$$\breve{a} \in \{\breve{a}|Q^t(i,\breve{a}) \approx \max_b Q^t(i,b) + \min_b Q^t(i,b) - Q^t(i,a)\}, \tag{3.6}$$

$$\breve{s}_i \in \{s_j|\min_{s_j}(\rho_{(s_i,s_j)})\}, \tag{3.7}$$

where $Q_{(i,a)}$ is an evaluation for the expected value of the accumulated reward if the agent starts with state i, takes action a, and follows a specific policy. ρ is a criterion showing the degree of similarity between current state and other states, calculated as follows:

$$\rho_{(s_i,s_j)} = 1 - \frac{\sum_{k\in A(i)} |Q(s_i,a_k) - Q(s_j,a_k)|}{\sum_{k\in A(i)} max\{Q(s_i,a_k), Q(s_j,a_k)\}}, \tag{3.8}$$

For example, if ρ equals to one for two different states s_i and s_j, it indicates that they can be classified as equivalent states. To find the opposite state $\breve{s}$ using

Equation 3.7, the minimum ρ with respect to all states should be considered. It is obvious that in the case of zero denominator in Equation 3.8 the opposite state cannot be well defined.

3.3.2 A new type II opposition learning

With respect to type II opposition, the action-value functions play a substantial role in finding the opposite action and state. However, the majority of action-value functions have zero value at the very beginning of the learning process or may have not been observed a sufficient number of times. Therefore, the opposites are not accurate enough. To increase the accuracy of finding the actual opposite action/state, some kind of function approximation, such as a feed-forward Multi-Layer Perceptron networks (MLP), Fuzzy Rule-Based modeling (FRB), or any other approach to function approximation could be used. In this type of opposition mining, regular learning is performed for some limited time in order to obtain initial information about the behavior of the system and specifically action-value functions. The information related to action-state pairs which are visited during learning can be employed as sample data for training the function approximations. Using other action-state pairs as testing data, new knowledge or a set of new action-value functions are extracted. In other words, we can introduce an *"opposite agent"* which is not actually taking any action. The knowledge provided by the function approximation can be used to create a set of new action-value functions which then are used for finding the opposite actions/states.

Another key point for the function approximation is how to choose the training data from the whole set of action-state pairs and their respective action-value functions. There is a simple way in which we can find an average over all observations related to action-state pairs at each time period. This value could be a basis for determining the set of training data, such that all pairs observed equal to or greater than this limit are considered as training data at that time. However, this limit might create some errors if, for example, the policy used to take an action is ϵ-greedy with a small ϵ, or completely greedy. Based on the nature of these policies, only a few actions might be taken most of the time, and

the rest are only picked a few times or not at all. In other words, the average limit might be biased toward the action-state pairs which have been visited many times. Therefore, there is a considerable possibility of missing some important information pertinent to other action-state pairs.

One simple way to tackle this problem is to store the percentage of changes for all action-value functions in every iteration. The reinforcement agent could initialize all these values to one at the beginning of the learning process and then keep tracking the changes for all action-value functions between two consecutive iterations, as illustrated in the following equation:

$$p^t_{Q(i,a)} = \frac{|Q^t_k(i,a) - Q^t_{k-1}(i,a)|}{Q^t_{k-1}(i,a)}, \qquad \text{if} \quad Q^t_{k-1}(i,a) \neq 0 \tag{3.9}$$

where $p^t_{Q(i,a)}$ is the percentage of change for the action-value function related to action-state pair (i,a) during period t and $Q^t_k(i,a)$ is the action-value function for that pair in the k^{th} iteration of the learning process.

In general, the action-state pairs which are more observed during the learning process can be chosen as training data. Therefore, taking an average over all these values could be a good limit to separate the training and testing data. Of course, there is a possibility that the percentage values corresponding to some action-state pairs become higher than one. This implies that some of the action-value functions have significantly changed in some iterations, which might indicate that they are only visited few times. These values could be considered as outliers based on some criteria and eliminated from the respective set used for computing the average. Figure 3.1 illustrates this scheme.

After obtaining the output of the approximation section, the appropriate method should be employed to generate new knowledge (new evaluation functions) for all action-state pairs (e.g., consider the actual action value functions for the training data and approximates ones for the testing data or use all approximate one).

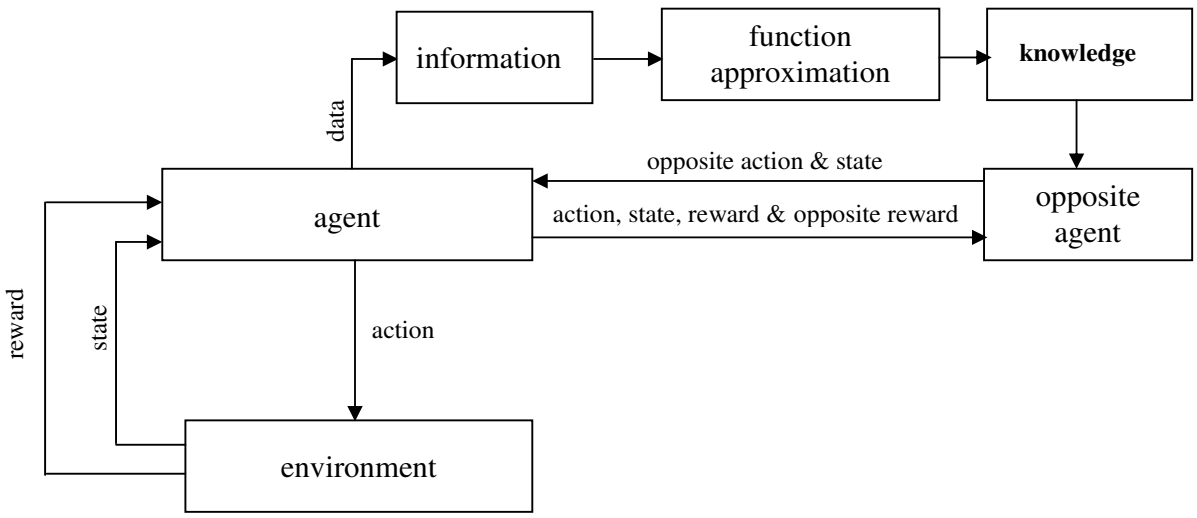

Figure 3.1: Type II opposition mining using function approximation

3.3.3 Opposition-based Q-Learning

The agent in Q-Learning updates its knowledge after each interaction with the environment using the signals it receives (immediate reward, r) and depending on the new situation (state, s_{t+1} or j), it makes another admissible decision (taking action, a) based on the new information acquired (e.g., new action-value functions, $Q(i,a)$ and updated policy, π). After receiving the reward and observing the new state, the agent can easily obtain the information pertinent to the stochastic parameters. Since the reward is usually a function of action, state, and the stochastic parameters, the agent can calculate the opposite reward $\breve{r}$ when the opposite action is chosen. The opposite state $\breve{j}$ could be derived using the transition function as well (e.g., the balance equation in 1.2).

The analogous updating step for action-state pair could be implemented for the opposite action-state pair as follows:

$$Q^t(i,\breve{a}) \;:=\; (1-\breve{\alpha})Q^t(i,\breve{a}) + \breve{\alpha}[r_{ij}^t(\breve{a}) \;+\gamma \max_{b\in A(\breve{j})} Q^{t+1}(\breve{j},b)], \tag{3.10}$$

where $\breve{\alpha}$ is the learning parameter for the opposite action.

The question may arise why only opposite action or opposite state should be tried, when we could consider all admissible actions or possible current states within the above

assumptions and update all respective action-value functions. The answer is simple: it is computationally expensive (in most real cases even impossible) to consider all actions or states in every iteration. Moreover, the combination of some actions and states with the specific value of stochastic parameter which occurred in that iteration would rarely happen in reality, and it might, therefore, cause some errors in the assessment of value function. We would actually like to use less updating of action-value functions in each iteration with the maximum effect on speeding up the learning process. In other words, we are interested in using the negative correlation between both action and opposite action, or state and opposite state, to discover something new which can be embedded into the current system knowledge in addition to those which have been experienced via observations in all iterations.

3.3.4 Opposition-based sarsa

Sarsa is an on-policy technique in RL, in which the agent follows the same policy π in the next iteration. In other words, the agent uses the behavior policy to make a decision and the same policy for improving the action value functions.

Suppose we want to develop the opposition version of the Sarsa technique based on the action and its opposite. Two opposite actions should be created: one is related to the action in the current state and another should be pertinent to the action for the next state where the opposite action is considered for the current state. The opposite action for the current state is easily computable because the agent has access to the action taken in the current state. However, it does not know what is the actual action for the opposite of the next state because it only takes an action for the next state. To tackle this problem, the agent should utilize some knowledge regarding the action taken in the next state. If this action is greedy, the actual action in the opposite of the next state will also be greedy. Otherwise, the same action can be considered as an actual action for this state if the set of admissible actions for both states is the same. In the case of different sizes, normalization can be performed to find the actual action for the opposite of the next sate.

Algorithm 4 Finding opposite actions in on-policy learning methods for ϵ-greedy policy

1: Set the admissible action for all possible states

2: Take action a in state $i = s^t$ using the ϵ-greedy policy and observe next state $j = s^{t+1}$ and immediate reward $r_{ij}^t(a)$

3: Take action a' (using the ϵ-greedy policy) in state j and store the action number, k'

4: Find the opposite action $\breve{a}$ (type I or type II opposite) for current state i

5: Find the opposite of the next state $\breve{j}$ when choosing $\breve{a}$ for i using transition function (e.g., the balance equation in 1.2)

$$\breve{j} = \hat{g}(i, \breve{a})$$

6: Find the greedy action for the next state

$$a'^* = arg \max_{b \in A(j)} Q^{t+1}(j, b)$$

7: If $a' = a'^*$, find the greedy action, $\hat{a}'$, for $\breve{j}$

$$\hat{a}' = arg \max_{b \in A(\breve{j})} Q^{t+1}(\breve{j}, b),$$

8: If $a' \neq a'^*$, find the following ratio

$$\Gamma = \frac{k'}{|A(j)|}$$

9: Find $\breve{\Gamma}$ as follows

$$\breve{\Gamma} = \Gamma \times |A(\breve{j})|$$

10: Find the closest action number in the set of admissible action, $A_{\breve{j}}$, to $\breve{\Gamma}$ and consider as $\hat{a}'$

11: Use $\hat{a}'$ and one of type I or type II opposition schemes to find the opposite action, $\breve{a}'$, for $\breve{j}$

3.3.5 Opposition-based sarsa(λ) and Q(λ)

These two techniques in RL use eligibility traces as basic mechanisms in an RL technique to update more action-value functions in every iteration. The parameter λ, along with discount factor γ, play a decaying role in the eligibility traces to put more weight on those states which have been recently visited.

In order to make these two techniques computationally efficient in a problem with large state spaces, we could augment these methods with an additional condition on the value of eligibility trace. That is, the step of updating for every action-state pair is performed if $e(i, a) > \epsilon_1$ in which ϵ_1 is a positive small value; otherwise, the respective value of $e(i, a)$ should be set to zero and deleted from the eligibility trace. In other words, the updating process is only performed for those action-state pairs in which the corresponding eligibility traces have values larger than zero or a small predefined value, say ϵ.

To develop the opposition version of Sarsa(λ) and Q(λ), we have to define a new eligibility trace related to opposite action or state, $\breve{e}(i, a)$. After finding the opposite action or state using Algorithm 4, the processes of setting the eligibility trace and updating the action value function should be repeated for the opposite action or state with respect to all elements in the opposite eligibility trace. There is a possibility that some action-value functions are updated multiple times in one iteration, because those action-state pairs are the members of both regular and opposite eligibility traces with values larger than ϵ_1. We assume that the process of updating is initially performed for action and state and then continued with the opposites. The complete process of Sarsa(λ) and Q(λ) for opposite action is represented in Algorithm 5. The analogous method can be performed for the opposite state.

3.4 New approaches for risk consideration

Mahootchi et al. [72] developed a new method to find the variance of the discounted accumulated reward for Q-Learning in an off-line learning and demonstrated its applicability in a single reservoir case study. In the following, we are going to extend this new

Algorithm 5 Opposition-Based Q(λ) and Sarsa(λ) for opposite action

1: Initialize $Q^t(i,a)$, $e(i,a)=0$, $\breve{e}(i,a)=0$, and ϵ_1 = small positive value

2: Initialize the eligibility traces matrix and its opposite, $\underline{e}(\cdot)$, $\underline{\breve{e}}(\cdot)$

3: Start an episode with an initial state, $i=s^t$,

4: Take action a, observe $j=s^{t+1}$, $r_{ij}^t(a)$ and choose another action a' with policy π

5: **Q(λ)** : find $a^*=arg\max_{b\in A(j)} Q^{t+1}(j,b)$
Sarsa(λ): consider $a^*=a'$

6: Calculate $\Delta=r_{ij}^t(a)+\gamma Q^{t+1}(j,a^*)-Q^t(i,a)$

7: Set $e(i,a)=e(i,a)+1$

8: Append a new column including action, state, and the value of eligibility trace to $\underline{e}(\cdot)$ or update the value of respective eligibility trace

9: Perform for all i and a in $\underline{e}(\cdot)$:

9-1) $Q^t(i,a):=Q^t(i,a)+\alpha\Delta e(i,a)$

9-2) **Q(λ)** : $e(i,a)=\gamma\lambda e(i,a)$ if $a'=a^*$
$e(i,a)=0$; otherwise
Sarsa(λ): $e(i,a)=\gamma\lambda e(i,a)$

9-3) if $e(i,a)>\epsilon_1$, update $e(i,a)$; otherwise, delete the respective column from $\underline{e}(\cdot)$

10: Choose the opposite action $\breve{a}$ for i, find the next state $\breve{j}$ using transition function, find opposite action $\breve{a}'$ using Algorithm 4, and find the opposite reward $\breve{r}$

11: **Q(λ):** find $\breve{a}^*=arg\max_{b\in A(\breve{j})} Q^{t+1}(\breve{j},b)$
Sarsa(λ): consider $\breve{a}^*=\breve{a}'$

12: $\breve{\Delta}=r_{ij}^t(\breve{a})+\gamma Q^{t+1}(\breve{j},\breve{a}^*)-Q^t(i,\breve{a})$

13: Set $\breve{e}(i,\breve{a})=\breve{e}(i,\breve{a})+1$

14: Add opposite action, state, and the value of eligibility trace to $\underline{\breve{e}}(\cdot)$ or update it

15: Perform for all i and a in $\underline{\breve{e}}(\cdot)$:

15-1 $Q^t(i,a):=Q^t(i,a)+\alpha\breve{\Delta}\breve{e}(i,\breve{a})$

15-2) **Q(λ)** : $\breve{e}(i,a)=\gamma\lambda\breve{e}(i,a)$ if $\breve{a}'=\breve{a}^*$,
$\breve{e}(i,a)=0$ otherwise
Sarsa(λ): $\breve{e}(i,a)=\gamma\lambda\breve{e}(i,a)$

15-3) if $\breve{e}(i,a)>\epsilon_1$, update $\breve{e}(i,a)$; otherwise, delete the respective column from $\underline{\breve{e}}(\cdot)$

16: Set $a=a'$ and $s=s'$ and go to the next step of the episode

17: Perform the process for all episodes

methodology to the mentioned RL methods for off- and on-line learning. Recall that Q-Learning is a way to approximate the expected value of accumulated reward for different action-state pairs based on the Robbins-Monro algorithm [55]. Using this algorithm, we are able to update the expected value of random variable in an iterative way with every new observation. Therefore, in equation $Q^t(i,a) = E[r_{ij}^t(a) + \max_{b \in A(j)} Q^{t+1}(j,b)]$, the term inside E can be considered as random, so that the general formulation of Q-Learning is created. Now we can use the same algorithm to establish a new formulation called squared-action-value function, $Q2(i,a)$. This new equation approximates the expected value of $[r_{ij}^t(a) + \max_{b \in A(j)} Q^{t+1}(j,b)]^2$ as follows:

$$Q2^t(i,a) \;:=\; Q2^t(i,a) + \alpha \left([r_{ij}^t(a) + \max_{b \in A(j)} Q^{t+1}(j,b)]^2 - Q2^t(i,a) \right). \tag{3.11}$$

Having these values in every step of the learning process gives the opportunity for the agent to know something about the variance of accumulated reward, such that they can be utilized to find the policy with risk consideration or *risk-sensitive policy* [65]. In this thesis, the standard deviation is used as a measure of risk because minimizing the standard deviation minimizes the risk. This is a well-known approach, for example, as in [36]. The standard deviation for every action-state pair can be easily approximated in every iteration:

$$\sigma^t(i,a) = \sqrt{Q2^t(i,a) - Q^t(i,a)^2}. \tag{3.12}$$

The reinforcement agent uses a greedy or an ϵ-greedy policy to take an action in every interaction with the environment. The agent actually updates its knowledge in terms of both action-value and squared-action-value functions. After converging to the stationary situation, the following formulation is considered to find the optimal policy for every state i in period t with respect to expected value of accumulated reward/punishment and its variance simultaneously:

$$a^{*t} \;=\; \arg\max_a \{Q^t(i,a) - \beta \sigma^t(i,a)\}, \tag{3.13}$$

where β is called the "risk-aversion factor". Indeed, the value of this factor changes how influential risk is compared to the average accumulated reward in deriving the operating

policy.

Of course, this is an off-line training because it uses simulation with some synthetic data, generated by forecasting methods or existing historical data, as training data.

If the on-line training is implemented (real-time decision-making), a policy such as greedy or ϵ-greedy should be constructed based on both the action-value function $Q^t(i,a)$ and its variance $(\sigma^2)^t(i,a)$ (or the standard deviation $\sigma^t(i,a)$) in every iteration. For example, if the agent would like to take or choose a greedy action in a ϵ-greedy policy, it should make a decision based on a new evaluation function $(Q')^t(i,a)$ defined as follows:

$$(Q')^t(i,a) \quad = \quad Q^t(i,a) - \beta\sigma^t(i,a). \tag{3.14}$$

In many real-world applications, off-line learning can be performed first to obtain some initial knowledge about the system, and then on-line learning using this knowledge can be started with real data [73, 74]. Q-Learning, Sarsa, Q(λ), and Sarsa(λ) in an off-line and on-line scheme with consideration of standard deviation of accumulated reward, are illustrated in Algorithms 6-7.

3.5 FP model with stochastic releases, Approach 1

As explained in section 2.3, there are two main assumptions pertinent to original FP [16] models for reservoir application in a stochastic situation:

1. The optimization model is constructed based on the open-loop policy in which the release is deterministic.

2. inflows into the reservoir are assumed to be normally distributed.

The first assumption can be quite unrealistic in many situations. For instance, in the case of water reservoir management, when the reservoir is almost full, operators would prefer to release more water than they normally do to prevent a possible flood from happening downstream causing damage. Or, when the amount of stored water is in short supply, operators should consider hedging (that is, releasing less water than normal). Therefore,

Algorithm 6 Q-Learning/Sarsa with variance consideration

1: Initialize $Q^t(i,a)$, $Q2^t(i,a)$

2: Set the value of risk-aversion factor β

3: Determine admissible action A for all states

4: Start an episode with an initial state $i = s$

5: Compute evaluation functions for the current state i

$$\text{off-line:} \quad (Q')^t(i,b) = Q^t(i,b) \qquad \forall b \in A(i)$$
$$\text{on-line:} \quad (Q')^t(i,b) = \{Q^t(i,b) - \beta\sqrt{[Q2^t(i,b) - Q^t(i,b)^2]}\} \quad \forall b \in A(i)$$

6: Take an action a in state i using policy π (e.g., ϵ-greedy policy with respect to $(Q')^t(i,a)$ for all $a \in A(i)$) and observe r and $j = s^{t+1}$

7: **Sarsa:** Choose action a' in state j using policy π (e.g., ϵ-greedy policy with respect to $(Q')^t(j,a)$ for all $a' \in A(j)$)

8: Update the following functions

Q-Learning

$$\begin{aligned} Q^t(i,a) &:= Q^t(i,a) + \alpha[r_{ij}^t(a) + \gamma Q^{t+1}(j, arg\max_{b \in Aj}(Q')^{t+1}(j,b)) - Q^t(i,a)] \\ Q2^t(i,a) &:= Q2^t(i,a) + \alpha\{[r_{ij}^t(a) + \gamma Q^{t+1}(j, arg\max_{b \in Aj}(Q')^{t+1}(j,b))]^2 - Q2^t(i,a)\} \end{aligned}$$

Sarsa

$$\begin{aligned} Q^t(i,a) &:= Q^t(i,a) + \alpha[r_{ij}^t(a) + Q^{t+1}(j,a') - Q^t(i,a)] \\ Q2^t(i,a) &:= Q2^t(i,a) + \alpha\{[r_{ij}^t(a) + Q^{t+1}(j,a')]^2 - Q2^t(i,a)\} \end{aligned}$$

9: Set $i = j$, $a = a'$ for Sarsa, and go to the next step of the episode

10: Perform the process for the next episode

11: The operating policy with risk consideration in every iteration can be approximated using Equation 3.13

Algorithm 7 Q(λ)/Sarsa(λ) with variance consideration

1: Initialize $Q^t(i,a)$, $Q2^t(i,a)$, $e(i,a)=0$, $\underline{e}(\cdot)$, and ϵ = small value

2: Set the value of risk-aversion factor β

3: Determine admissible actions $A(\cdot)$ for all states

4: Start an episode with an initial state $i=s$,

5: Compute evaluation functions for the current state i

$$\text{off-line:} \quad (Q')^t(i,b) = Q^t(i,b) \qquad \forall b \in A(i)$$
$$\text{on-line:} \quad (Q')^t(i,b) = \{Q^t(i,b) - \beta\sqrt{[Q2^t(i,b) - Q^t(i,b)^2]}\} \quad \forall b \in A(i)$$

6: Take action a in state i using policy π (e.g., ϵ-greedy policy with respect to $(Q')^t(i,a)$ for all $a \in A(i)$) and observe r^t and $j = s^{t+1}$

7: Choose action a' in state j using policy π (e.g., ϵ-greedy policy with respect to $(Q')^{t+1}(i,a)$ for all $a' \in A(j)$)

8: **Q(λ)** : find $a^* = arg\max_{b \in A(j)} (Q')^{t+1}(j,b)$
Sarsa(λ): consider $a^* = a'$

9: $\Delta = r_{ij}^t(a) + \gamma Q^{t+1}(j,a^*) - Q^t(i,a)$

10: $\Delta_2 = [r_{ij}^t(a) + \gamma Q^{t+1}(j,a^*)]^2 - Q^t(i,a)$

11: Set $e(i,a) = e(i,a) + 1$

12: Append a new column including action, state, and the value of trace to $\underline{e}(\cdot)$ or update the value of the respective eligibility trace

13: Perform for all i and a in matrix $\underline{e}(\cdot)$:

13-1) $Q^t(i,a) := Q^t(i,a) + \alpha\Delta e(i,a)$

13-2) $Q2^t(i,a) := Q2^t(i,a) + \alpha\Delta_2 e(i,a)$

13-3) **Q(λ):** $e(i,a) = \gamma\lambda e(i,a)$ if $a' = a^*$
$e(i,a) = 0$ otherwise
Sarsa(λ): $e(i,a) = \gamma\lambda e(i,a)$

13-4) if $e(i,a) > \epsilon$, update $e(i,a)$; otherwise, delete the respective column from $\underline{e}(\cdot)$ (the matrix including all eligibility traces bigger than ϵ)

14: Set $i = j$ and $a = a'$ go to the next step of the episode

15: Perform the process for the next episode

16: The operating policy with risk consideration in every iteration can be approximated using Equation 3.13

the optimum release policy should be a function of current storage, which makes the release stochastic because the storage is a random variable. Therefore, the new stochastic release in the optimization model can be introduced as follows:

$$u_{ij}^t = k_{ij}^t + s_i^{t-1}, \tag{3.15}$$

where u_{ij}^t is the total release and k_{ij}^t is the deterministic part, both from reservoir i to reservoir j in period t, and s_i^{t-1} is the end-of-period storage in period $t-1$.

The extension of this enhanced FP model [40] to multi-reservoirs for the case of stochastic releases is developed first for a two-reservoir problem with serial and parallel configuration (Figure 3.2), and will then be generalized for multi reservoirs.

Recall that, in the FP model developed for deterministic releases, the inflow has been

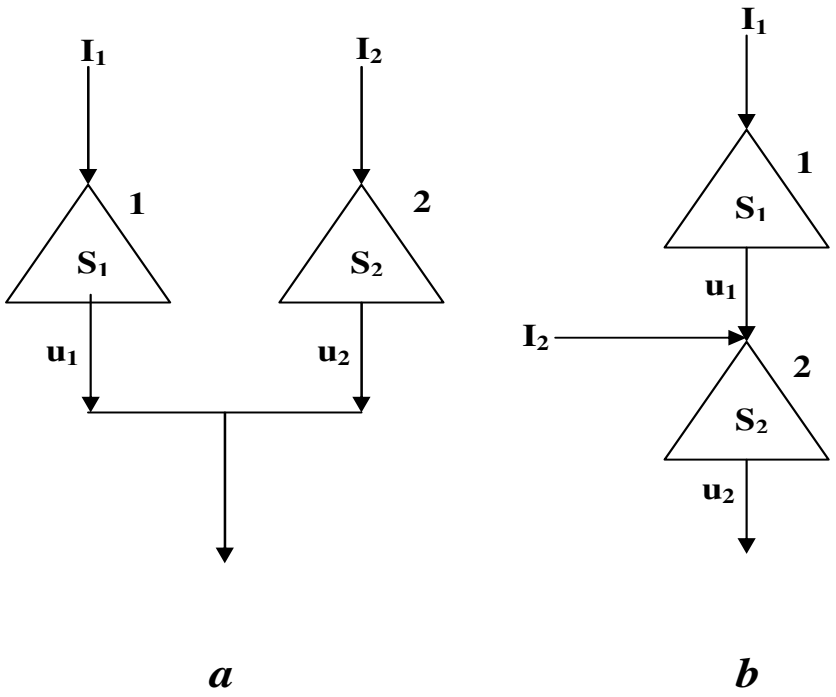

Figure 3.2: Two-reservoir case study a)parallel, b)serial

divided into two components, one for the natural mean of inflow ($\bar{I}_i^t$), and an another for zero-mean random component (η_i^t).

Let us start with the easiest configuration, namely, the two parallel reservoirs in Part a of Figure 3.2. The formulation which represents the dynamics of reservoirs uses the mass balance equations, the respective indicator functions, which embody the reliability and

the failure probabilities, and the stochastic release in Equation 3.15. It can be written as the following:

$$\begin{aligned} s_i^t &= \{\bar{I}_i^t + \eta_i^t - k_{ii}^t\} 1_{[s_{i,min}^t, s_{i,max}^t]}(\hat{s}_i^t) + \{s_{i,min}^t\} 1_{(-\infty, s_{i,min}^t)}(\hat{s}_i^t) \\ &+ \{s_{i,max}^t\} 1_{(s_{i,max}^t, \infty)}(\hat{s}_i^t), \end{aligned} \tag{3.16}$$

where

s_i^t : the end-of-period storage level of reservoir i in period t,

k_{ii}^t : the deterministic part of the total release from reservoir i to all downstream reservoirs (a decision variable in the optimization problem),

$s_{i,min}^t$: the minimum storage level of i^{th} reservoir in period t,

$s_{i,max}^t$: the maximum storage level of i^{th} reservoir in period t,

$\hat{s}_i^t$: the end-of-period available water
$(\hat{s}_i^t = s_i^{t-1} + \bar{I}_i^t + \eta_i^t - (k_{ii}^t + s_i^{t-1}) = \bar{I}_i^t + \eta_i^t - k_{ii}^t)$,

$1_{[\ ,\ ]}$: the indicator functions as zero-one random variables pertinent to reliability and failure conditions.

Indicator functions for the two parallel reservoirs configuration are set to one or zero using the following rules

$$\begin{aligned} 1_{[s_{i,min}^t, s_{i,max}^t]}(\hat{s}_i^t) &= 1 \quad \text{if } s_{i,min}^t \le \hat{s}_i^t \le s_{i,max}^t \ \text{ or } \ s_{i,min}^t - \bar{I}_i^t + k_{ii}^t \le \eta_i^t \le s_{i,max}^t - \bar{I}_i^t + k_{ii}^t, \\ 1_{[s_{i,min}^t, s_{i,max}^t]}(\hat{s}_i^t) &= 0 \quad \text{otherwise}, \\ 1_{(-\infty, s_{i,min}^t)}(\hat{s}_i^t) &= 1 \quad \text{if } \eta_i^t < s_{i,min}^t - \bar{I}_i^t + k_{ii}^t, \\ 1_{(-\infty, s_{i,min}^t)}(\hat{s}_i^t) &= 0 \quad \text{otherwise}, \\ 1_{(s_{i,max}^t, \infty)}(\hat{s}_i^t) &= 1 \quad \text{if } \eta_i^t > s_{i,max}^t - \bar{I}_i^t + k_{ii}^t, \\ 1_{(s_{i,max}^t, \infty)}(\hat{s}_i^t) &= 0 \quad \text{otherwise}. \end{aligned}$$

The first and the second moment equations for storage of each reservoir, respectively, are:

$$\begin{aligned} E(s_i^t) &= E\left[\{\bar{I}_i^t + \eta_i^t - k_{ii}^t\} 1_{[s_{i,min}^t, s_{i,max}^t]}(\hat{s}_i^t)\right] + \{s_{i,min}^t\} E\left[1_{(-\infty, s_{i,min}^t)}(\hat{s}_i^t)\right] \\ &+ \{s_{i,max}^t\} E\left[1_{(s_{i,max}^t, \infty)}(\hat{s}_i^t)\right], \end{aligned} \tag{3.17}$$

$$\begin{aligned} E(s_i^t)^2 &= E\left[\{(\bar{I}_i^t - k_{ii}^t)^2 + (\eta_i^t)^2 + 2\eta_i^t(\bar{I}_i^t - k_i^t)\}1_{[s_{i,min}^t, s_{i,max}^t]}(\hat{s}_i^t)\right] \\ &+ \{(s_{i,min}^t)^2\}E\left[1_{(-\infty, s_{i,min}^t)}(\hat{s}_i^t)\right] + \{(s_{i,max}^t)^2\}E\left[1_{(s_{i,max}^t, \infty)}(\hat{s}_i^t)\right]. \end{aligned} \tag{3.18}$$

Since the new $\hat{s}_i^t$ is independent from the previous storage level s_i^{t-1} (i.e., storage level does not influence future inflow, a reasonable assumption for most reservoirs), the expected value of indicator functions can be easily written as

$$\begin{aligned} E\left[1_{[s_{i,min}^t, s_{i,max}^t]}(\hat{s}_i^t)\right] &= Pr(s_{i,min}^t - \bar{I}_i^t + k_{ii}^t \le \eta_i^t \le s_{i,min}^t - \bar{I}_i^t + k_i^t) \\ &= \int_{s_{i,min}^t - \bar{I}_i^t + k_i^t}^{s_{i,max}^t - \bar{I}_i^t + k_{ii}^t} f_{I_i^t}(\eta_i^t)\, d\eta_i^t, \end{aligned} \tag{3.19}$$

$$E\left[1_{(-\infty, s_{i,min}^t)}(\hat{s}_i^t)\right] = Pr(\eta_i^t < s_{i,min}^t - \bar{I}_i^t + k_{ii}^t) = \int_{-\infty}^{s_{i,min}^t - \bar{I}_i^t + k_{ii}^t} f_{\eta_i^t}(\eta_i^t)\, d\eta_i^t, \tag{3.20}$$

$$E\left[1_{(s_{i,max}^t, \infty)}(\hat{s}_i^t)\right] = Pr(\eta_i^t > s_{i,max}^t - \bar{I}_i^t + k_{ii}^t) = \int_{s_{i,max}^t - \bar{I}_i^t + k_{ii}^t}^{\infty} f_{\eta_i^t}(\eta_i^t)\, d\eta_i^t. \tag{3.21}$$

For the time being, it is assumed that the inflows are normally and independently distributed (the normality assumption will be relaxed later in Section 3.7); therefore, the first and the second moments for both parallel reservoirs are analogous and are similar to the ones in [40] and can be written as

$$\begin{aligned} E(s_i^t) &= (\bar{I}_i^t - k_{ii}^t) \cdot \left\{\tfrac{1}{2}\left[erf\left(\tfrac{[s_{i,max}^t - (\bar{I}_i^t - k_{ii}^t)]}{[2Var(\eta_i^t)]^{1/2}}\right) - erf\left(\tfrac{[s_{i,min}^t - (\bar{I}_i^t - k_{ii}^t)]}{[2Var(\eta_i^t)]^{1/2}}\right)\right]\right\} \\ &- \left(\tfrac{[Var(\eta_i^t)]^{(1/2)}}{\sqrt{2\pi}}\right) \cdot \left\{\exp\left(-\tfrac{1}{2}\tfrac{[s_{i,max}^t - (\bar{I}_i^t - k_{ii}^t)]^2}{Var(\eta_i^t)}\right) - \exp\left(-\tfrac{1}{2}\tfrac{[s_{i,min}^t - (\bar{I}_i^t - k_{ii}^t)]^2}{Var(\eta_i^t)}\right)\right\} \\ &+ (s_{i,min}^t) \cdot \left\{\tfrac{1}{2}\left[1 + erf\left(\tfrac{[s_{i,min}^t - (\bar{I}_i^t - k_{ii}^t)]}{[2Var(\eta_i^t)]^{1/2}}\right)\right]\right\} \\ &+ (s_{i,max}^t) \cdot \left\{\tfrac{1}{2}\left[1 - erf\left(\tfrac{[s_{i,max}^t - (\bar{I}_i^t - k_{ii}^t)]}{[2Var(\eta_i^t)]^{1/2}}\right)\right]\right\}, \end{aligned} \tag{3.22}$$

$$
\begin{aligned}
E(s_i^t)^2 = \\
&[(\bar{I}_i^t - k_{ii}^t)^2 - Var(\eta_i^t)] \cdot \left\{ \frac{1}{2} \left[\ erf\left(\frac{[s_{i,max}^t - (\bar{I}_i^t - k_{ii}^t)]}{[2Var(\eta_i^t)]^{1/2}} \right) - erf\left(\frac{[s_{i,min}^t - (\bar{I}_i^t - k_{ii}^t)]}{[2Var(\eta_i^t)]^{1/2}} \right) \ \right] \right\} \\
&+2(\bar{I}_i^t - k_{ii}^t) \cdot \left(\frac{-[Var(\eta_i^t)]^{(1/2)}}{\sqrt{2\pi}} \right) \left\{ \exp\left(-\frac{1}{2} \frac{[s_{i,max}^t - (\bar{I}_i^t - k_{ii}^t)]^2}{Var(\eta_i^t)} \right) - \exp\left(-\frac{1}{2} \frac{[s_{i,min}^t - (\bar{I}_i^t - k_{ii}^t)]^2}{Var(\eta_i^t)} \right) \right\} \\
&- \frac{[Var(\eta_i^t)]^{(1/2)}}{\sqrt{2\pi}}) \cdot \left\{ [s_{i,max}^t - (\bar{I}_i^t - k_{ii}^t)] \cdot \exp\left(-\frac{1}{2} \frac{[s_{i,max}^t - (\bar{I}_i^t - k_{ii}^t)]^2}{Var(\eta_i^t)} \right) \right. \\
&\qquad \left. - [s_{i,min}^t - (I_i^t - k_{ii}^t)] \cdot \exp\left(-\frac{1}{2} \frac{[s_{i,min}^t - (\bar{I}_i^t - k_{ii}^t)]^2}{Var(\eta_i^t)} \right) \right\} \\
&+ (s_{i,min}^t)^2 \cdot \left\{ \frac{1}{2} \left[1 + erf\left(\frac{[s_{i,min}^t - (\bar{I}_i^t - k_{ii}^t)]}{[2Var(\eta_i^t)]^{1/2}} \right) \right] \right\} + (s_{i,max}^t)^2 \cdot \left\{ \frac{1}{2} \left[1 - erf\left(\frac{[s_{i,max}^t - (\bar{I}_i^t - k_{ii}^t)]}{[2Var(\eta_i^t)]^{1/2}} \right) \right] \right\},
\end{aligned}
\tag{3.23}
$$

where $Var(\eta_i^t)$ is the variance of inflow in reservoir i during period t and erf is the error function. As observed in the parallel configuration, the moments for both reservoirs are exactly the same as in the single reservoir problem, because there is no physical relation between these two reservoirs.

However, in the serial configuration (Part *b* of Figure 3.2), the moments depend on the release flowing from the upstream reservoir. In other words, if this release, which is called u_{12}^t in our formulations, is considered based on the release policy (i.e., $u_{12}^t = k_{12}^t + s_1^{t-1}$), the end-of-period storage in the second reservoir (s_2^t) will depend on two stochastic variables, which include the previous storage of the first reservoir (s_1^{t-1}) and the net inflow into the second reservoir ($\bar{I}_2^t + \eta_2^t$). This fact is shown in Equation 3.24 using the mass conservation equation for the second reservoir.

$$
s_2^t = s_2^{t-1} + \bar{I}_2^t + \eta_2^t + u_{12}^t - (k_{22}^t + s_2^{t-1}) = \bar{I}_2^t + \eta_2^t + k_{12}^t + s_1^{t-1} - k_{22}^t. \tag{3.24}
$$

It should be noted that the spillage and shortage situations which affect the amount of release flowing to the downstream reservoir are not considered in u_{12}^t.

The moments for the upstream-most reservoirs are the same as reservoirs in parallel configuration; however, we have to approximate the release to the downstream reservoir,

analogous to what Fletcher and Ponnambalam have done in [16] for multi-reservoir problem, with deterministic releases.

Let us start with the expected value of indicator functions pertinent to the second reservoir when the storage is within the bounds and then extend it to other indicator functions:

$$E\left[1_{[s^t_{2,min},s^t_{2,max}]}(\hat{s}^t_2)\right] = Pr\left\{s^t_{2,min} - \bar{I}^t_2 - u^t_{12} + k^t_{22} \le \eta^t_2 \le s^t_{2,min} - \bar{I}^t_2 - u^t_{12} + k^t_{22}\right\}$$

$$= Pr\left\{s^t_{2,min} - \bar{I}^t_2 - k^t_{12} - s^{t-1}_1 + k^t_{22} \le \eta^t_2 \le s^t_{2,min} - \bar{I}^t_2 - k^t_{12} - s^{t-1}_1 + k^t_{22}\right\}$$

$$= \int_{s^t_{1,min}}^{s^t_{1,max}} \int_{s^t_{2,min}-\bar{I}^t_2-k^t_{12}-s^{t-1}_1+k^t_{22}}^{s^t_{2,max}-\bar{I}^t_2-k^t_{12}-s^{t-1}_1+k^t_{22}} f_{(\eta^t_2,s^{t-1}_1)}(\eta^t_2, s^{t-1}_1)\, d\eta^t_2 ds^{t-1}_1. \tag{3.25}$$

We can reasonably assume that the end-of-period storage in period $t-1$ for the first reservoir is independent from the natural inflow into the second reservoir, and the above equation can be therefore expressed as:

$$E\left[1_{[s^t_{2,min},s^t_{2,max}]}(\hat{s}^t_2)\right] = \int_{s^t_{1,min}}^{s^t_{1,max}} \left[\int_{s^t_{2,min}-\bar{I}^t_2-k^t_{12}-s^{t-1}_1+k^t_{22}}^{s^t_{2,max}-\bar{I}^t_2-k^t_{12}-s^{t-1}_1+k^t_{22}} f_{\eta^t_2}(\eta^t_2)\, d\eta^t_2\right] f_{s^{t-1}_1}(s^{t-1}_1)\, ds^{t-1}_1$$

$$= E\left\{Pr[s^t_{2,min} - \bar{I}^t_2 - k^t_{12} - s^{t-1}_1 + k^t_{22} \le \eta^t_2 \le s^t_{2,max} - \bar{I}^t_2 - k^t_{12} - s^{t-1}_1 + k^t_{22}]\right\}. \tag{3.26}$$

Using the first order approximation of Taylor's series for deriving the above expected value leads to the following equation

$$E\left[1_{[s^t_{2,min},s^t_{2,max}]}(\hat{s}^t_2)\right] \approx$$

$$Pr[s^t_{2,min} - \bar{I}^t_2 - k^t_{12} - E(s^{t-1}_1) + k^t_{22} \le \eta^t_2 \le s^t_{2,max} - \bar{I}^t_2 - k^t_{12} - E(s^{t-1}_1) + k^t_{22}]. \tag{3.27}$$

This process can be repeated for other indicator functions pertinent to the second reservoir

$$E\left[1_{(-\infty,s_{2,min}^t)}(\hat{s}_2^t)\right] \approx Pr[\eta_2^t < s_{2,min}^t - \bar{I}_2^t - k_{12}^t - E(s_1^{t-1}) + k_{22}^t], \tag{3.28}$$

$$E\left[1_{(s_{2,max}^t,\infty)}(\hat{s}_2^t)\right] \approx Pr[\eta_2^t > s_{2,max}^t - \bar{I}_2^t - k_{12}^t - E(s_1^{t-1}) + k_{22}^t]. \tag{3.29}$$

Therefore, the related moments for the second reservoir can be written as follows, where $K_{22}^t = k_{12}^t + E(s_1^{t-1}) - k_{22}^t$,

$$\begin{aligned}
&E(s_2^t) = \\
&\left[\bar{I}_2^t + K_{22}^t\right] \cdot \left\{\tfrac{1}{2}\left[erf\left(\tfrac{[s_{2,max}^t - (\bar{I}_2^t + K_{22}^t)]}{[2Var(\eta_2^t)]^{1/2}}\right) - erf\left(\tfrac{[s_{2,min}^t - (\bar{I}_2^t + K_{22}^t)]}{[2Var(\eta_2^t)]^{1/2}}\right)\right]\right\} \\
&-\left(\tfrac{[Var(\eta_2^t)]^{(1/2)}}{\sqrt{2\pi}}\right) \cdot \left\{\exp\left(-\tfrac{1}{2}\tfrac{[s_{2,max}^t - (\bar{I}_2^t + K_{22}^t)]^2}{Var(\eta_2^t)}\right) - \exp\left(-\tfrac{1}{2}\tfrac{[s_{2,min}^t - (\bar{I}_2^t + K_{22}^t)]^2}{Var(\eta_2^t)}\right)\right\} \\
&+(s_{2,min}^t) \cdot \left\{\tfrac{1}{2}\left[1 + erf\left(\tfrac{[s_{2,min}^t - (\bar{I}_2^t + K_{22}^t)]}{[2Var(\eta_2^t)]^{1/2}}\right)\right]\right\} + (s_{2,max}^t) \cdot \left\{\tfrac{1}{2}\left[1 - erf\left(\tfrac{[s_{2,max}^t - (\bar{I}_2^t + K_{22}^t)]}{[2Var(\eta_2^t)]^{1/2}}\right)\right]\right\},
\end{aligned} \tag{3.30}$$

$$\begin{aligned}
&E(s_2^t)^2 = \\
&\left[(\bar{I}_2^t + K_{22}^t)^2 - Var(\eta_2^t)\right] \cdot \left\{\tfrac{1}{2}\left[\ \ erf\left(\tfrac{[s_{2,max}^t - (\bar{I}_2^t + K_{22}^t)]}{[2Var(\eta_2^t)]^{1/2}}\right) - erf\left(\tfrac{[s_{2,min}^t - (\bar{I}_2^t + K_{22}^t)]}{[2Var(\eta_2^t)]^{1/2}}\right)\ \ \right]\right\} \\
&+2(\bar{I}_2^t + K_{22}^t) \cdot \left(\tfrac{-[Var(\eta_2^t)]^{(1/2)}}{\sqrt{2\pi}}\right) \left\{\exp\left(-\tfrac{1}{2}\tfrac{[s_{2,max}^t - (\bar{I}_2^t + K_{22}^t)]^2}{Var(\eta_2^t)}\right) - \exp\left(-\tfrac{1}{2}\tfrac{[s_{2,min}^t - (\bar{I}_2^t + K_{22}^t)]^2}{Var(\eta_2^t)}\right)\right\} \\
&-\tfrac{[Var(\eta_2^t)]^{(1/2)}}{\sqrt{2\pi}}) \cdot \left\{[s_{2,max}^t - (\bar{I}_2^t + K_{22}^t)] \cdot \exp\left(-\tfrac{1}{2}\tfrac{[s_{2,max}^t - (\bar{I}_2^t + K_{22}^t)]^2}{Var(\eta_2^t)}\right)\right. \\
&\left.-[s_{2,min}^t - (\bar{I}_2^t + K_{22}^t)] \cdot \exp\left(-\tfrac{1}{2}\tfrac{[s_{2,min}^t - (\bar{I}_2^t + K_{22}^t)]^2}{Var(\eta_2^t)}\right)\right\} \\
&+(s_{2,min}^t)^2 \cdot \left\{\tfrac{1}{2}\left[1 + erf\left(\tfrac{[s_{2,min}^t - (\bar{I}_2^t + K_{22}^t)]}{[2Var(\eta_2^t)]^{1/2}}\right)\right]\right\} + (s_{2,max}^t)^2 \cdot \left\{\tfrac{1}{2}\left[1 - erf\left(\tfrac{[s_{2,max}^t - (\bar{I}_2^t + K_{22}^t)]}{[2Var(\eta_2^t)]^{1/2}}\right)\right]\right\}.
\end{aligned} \tag{3.31}$$

As can be seen in Equation 3.30 or 3.31, the only difference in the moment equations in terms of mathematical notation compared to what is in equations 3.22 or 3.23 for parallel reservoirs is that k is replaced by K.

It can be demonstrated that, if $N-1$ reservoirs are physically connected to another reservoir (e.g., reservoir N in Figure 3.3), the moments for that reservoir will be written in the same way as equations 3.30 and 3.31 by setting the variable K as follows:

$$\begin{aligned} K^t_{NN} &= [k^t_{1N)} + E(s^{t-1}_1)] + [k^t_{2N)} + E(s^{t-1}_2)] + \cdots + [k^t_{(N-1)N} + E(s^{t-1}_{N-1})] - k^t_{NN} \\ &= \sum_{j=1}^{N-1} [k^t_{jN} + E(s^{t-1}_j)] - k^t_{NN}. \end{aligned} \tag{3.32}$$

In this formula, the capital K related to reservoir N consists of two components: the

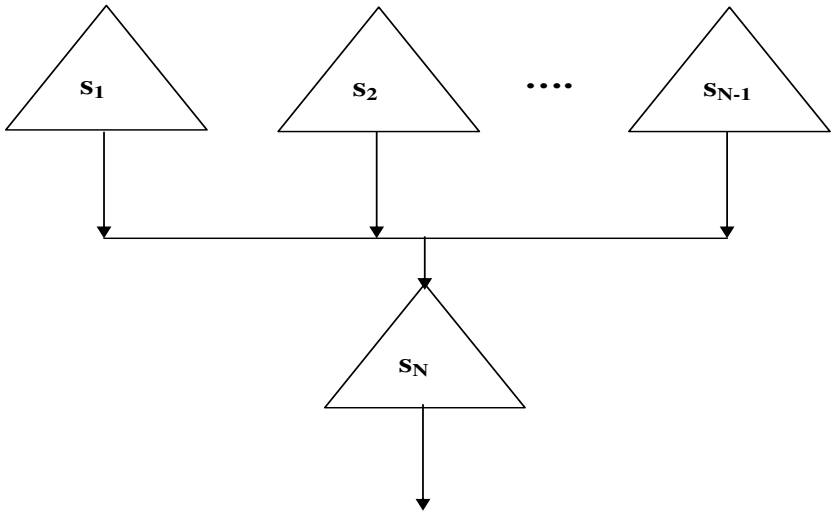

Figure 3.3: N physically connected reservoirs to one reservoir

deterministic part of release for this reservoir (k^t_{NN}), and the total water released into this reservoir from others ($\sum_{j=1}^{N-1}(k^t_{jN} + E(s^{t-1}_j))$, based on the release-route matrix explained later. In latter sections, a new approximation approach is suggested for these combined releases.

For the sake of notation, it is assumed that all reservoirs are physically connected to each other; therefore, given N reservoirs in the system model, all possible releases, and deterministic parts of releases can be introduced as two different matrices denoted as U

and K, respectively, whose elements can be represented as follows:

$$\begin{cases} u_{ij}^t : &= u_{ij}^t \times \delta_{ij}, \\ k_{ij}^t : &= k_{ij}^t \times \delta_{ij}, \end{cases} \quad \forall \;\; i,j = 1, \cdots N \tag{3.33}$$

It is also necessary to find the percentage of water released from the upstream reservoir to each downstream reservoir. The respective percentage of flows can be approximated using the following equation:

$$\psi_{ij}^t = \begin{cases} \frac{k_{ij}^t + E(s_i^t)}{k_{ii}^t + E(s_i^t)} & \text{if} \quad k_{ii}^t + E(s_i^t) \neq 0 \\ \frac{\delta_{ij}}{[\sum_j^N \delta_{ij}] - 1} & \text{if} \quad k_{ii}^t + E(s_i^t) = 0 \;\; \& \;\; i \neq j \;\; \& \;\; \delta_{ij} \neq 0 \\ 1 & \text{if} \quad k_{ii}^t + E(s_i^t) = 0 \;\; \& \;\; i = j \\ 0 & \text{otherwise} \end{cases} . \tag{3.34}$$

For example, in the case of two serial reservoirs, because the total water released from the first reservoir is available as an input into the second reservoir, $\psi_{12}^t = 1$ for all periods. The objective function in storage management is usually a function of release and the end-of-period storage ($z = f(u^t, s^t)$). These two variables in the new model are random variables, and their expected values need to be considered in the objective function. The expected value of water released $E(u)$ can be approximated as

$$E(u_{ij}^t) = k_{ij}^t + E(s_i^{t-1}). \tag{3.35}$$

The deficit or spillage situation are not considered here. This assumption will be relaxed in later sections.

The lower and upper bounds of the releases are:

$$R_{ij,min}^t \leq u_{ij}^t \leq R_{ij,max}^t \quad \Rightarrow \quad R_{ij,min}^t \leq k_{ij}^t + s_i^{t-1} \leq R_{ij,max}^t, \tag{3.36}$$

where $R_{ij,min}^t$ and $R_{ij,max}^t$ are the minimum and maximum releases flowing from reservoir i to j in period t, respectively.

As mentioned above, the release in our new model is stochastic because it is a function of previous storage, which is a random variable. Chance-constrained programming is commonly used to tackle these bounds in optimization problems where the distribution

of the respective random variables is known. This method, therefore, cannot be used for the stochastic release in our system model, because the distribution of storage is unknown. One way to control the releases to be within their maximum and minimum values is to substitute the release in Equation 3.36 with its expected value and change the lower and upper bounds (Equation 3.37):

$$R^t_{ij,min} + h^t_i \leq E(u^t_{ij}) \leq R^t_{ij,max} - h^t_i. \tag{3.37}$$

$E(u^t_{i,j})$ can be calculated by using Equation 3.35, and h^t_i is a constant value which can be substituted with the standard deviation of releases or storage calculated using Equation 3.35 as follows:

$$h^t_i = \sqrt{E(s^t_i)^2 - [E(s^t_i)]^2} = \sigma(s^t_i), \tag{3.38}$$

A set of boundary conditions should be set for the expected value of storage levels $E(s^t_i)$ and the deterministic part of releases (k^t_{ij}) similarly:

$$s^t_{i,min} + h^t_i \leq E(s^t_i) \leq s^t_{i,max} - h^t_i, \tag{3.39}$$

$$R^t_{ij,min} - s^t_{i,max} \leq k^t_{ij} \leq R^t_{ij,max} - s^t_{i,min}. \tag{3.40}$$

The constraint for balancing the total outflow with outflows to other reservoirs can be written as:

$$k^t_{ii} + E(s^{t-1}_i) - \sum_{j=1,j\neq i}^{N} [k^t_{ij} + E(s^{t-1}_i)] = 0. \tag{3.41}$$

In the above model for multi-reservoir applications, the deficit and spillage situations have not been directly considered in finding the expected value of the release. A more precise approximation in which these two situations are incorporated in the formulation of the expected value of release is proposed in the next section. The corresponding FP optimization model is described in the case study application in chapter 4

3.6 FP model with stochastic releases, Approach 2

In the first approach in the previous section, the expected value of release has been substituted with $k^t_{ij} + E(s^{t-1}_i)$, which we call the Upstream Mean Releases with lower

accuracy. The following two choices would be better approximations, in which case we refer to them as the Upstream Mean Releases with higher accuracy (the releases in this method are calculated using various probabilities):

1. where spilled water does not flow to other reservoirs, the release is:

$$u_{ij}^t = \begin{cases} k_{ij}^t + s_i^{t-1} & \text{if } L \le \eta_i^t \le U, \\ k_{ij}^t + s_i^{t-1} & \text{if } \eta_i^t > U, \\ \psi_{ij}^t \times \left[(-s_{i,min}^t + s_i^{t-1} + \bar{I}_i^t + \eta_i^t + \sum_{l=1 \ \& \ l \neq i}^{N} u_{li}^t \right] & \text{if } \eta_i^t < L, \end{cases} \tag{3.42}$$

where ψ_{ij}^t is the percentage value obtained from Equation 3.34 for every release from reservoir i to reservoir j, L and U are lower and upper bounds which can be written as:

$$\begin{cases} L &= s_{i,min}^t - \bar{I}_i^t + k_{ii}^t - \sum_{l=1 \ l \neq i}^{N} u_{li}^t, \\ U &= s_{i,max}^t - \bar{I}_i^t + k_{ii}^t - \sum_{l=1 \ l \neq i}^{N} u_{li}^t. \end{cases} \tag{3.43}$$

Using the first order approximation of Taylor's series, the expected value of release can be expressed as:

$$\begin{aligned} E(u_{ij}^t) &= [k_{ij}^t + E(s_i^{t-1})] \times \left\{ \int_{s_{i,min}^t - \bar{I}_i^t + k_{ii}^t - \sum_{l=1 \ l \neq i}^{N} E(u_{li}^t)}^{\infty} f_{\eta_i^t}(\eta_i^t) \, d\eta_i^t \right\} \\ &+ \psi_{ij}^t \times \left\{ \left[-s_{i,min}^t + E(s_i^{t-1}) + \bar{I}_i^t + \sum_{l=1, l \neq i}^{N} [E(u_{li}^t)] \right] \right. \\ &\times \left[\int_{-\infty}^{s_{i,min}^t - \bar{I}_i^t + k_{ii}^t - \sum_{l=1 \ l \neq i}^{N} E(u_{li}^t)} f_{\eta_i^t}(\eta_i^t) \, d\eta_i^t \right] \\ &+ \left. \int_{-\infty}^{s_{i,min}^t - \bar{I}_i^t - k_{ii}^t + \sum_{l=1 \ l \neq i}^{N} E(u_{li}^t)} \eta_i^t f_{\eta_i^t}(\eta_i^t) \, d\eta_i^t \right\}. \end{aligned} \tag{3.44}$$

2. When spills from upstream reservoirs are additional inflows, the release is:

$$u_{ij}^t = \begin{cases} k_{ij}^t + s_i^{t-1} & \text{if } L \le \eta_i^t \le U, \\ \psi_{ij}^t \times \left[-s_{i,max}^t + s_i^{t-1} + \bar{I}_i^t + \eta_i^t + \sum_{l=1 \ \& \ l \neq i}^{N} u_{li}^t \right] & \text{if } \eta_i^t > U, \\ \psi_{ij}^t \times \left[-s_{i,min}^t + s_i^{t-1} + \bar{I}_i^t + \eta_i^t + \sum_{l=1 \ \& \ l \neq i}^{N} u_{li}^t \right] & \text{if } \eta_i^t < L. \end{cases} \quad . \quad (3.45)$$

Using the first order approximation of Taylor's series, the expected value of release can be expressed as:

$$\begin{aligned} E(u_{ij}^t) = {} & \left[k_{ij}^t + E(s_i^{t-1})\right] \times \left\{ \int_{s_{i,min}^t - \bar{I}_i^t + k_{ii}^t - \sum_{l=1 \ l \neq i}^{N} E(u_{li}^t)}^{s_{i,max}^t - \bar{I}_i^t + k_{ii}^t - \sum_{l=1 \ l \neq i}^{N} E(u_{li}^t)} f_{\eta_i^t}(\eta_i^t)\, d\eta_i^t \right\} \\ & + \psi_{ij}^t \times \left\{ \left[-s_{i,min}^t + E(s_i^{t-1}) + \bar{I}_i^t + \sum_{l=1, l \neq i}^{N} E(u_{li}^t) \right] \right. \\ & \times \left[\int_{-\infty}^{s_{i,min}^t - \bar{I}_i^t + k_{ii}^t - \sum_{l=1 \ l \neq i}^{N} E(u_{li}^t)} f_{\eta_i^t}(\eta_i^t)\, d\eta_i^t \right] \\ & \left. + \int_{-\infty}^{s_{i,min}^t - \bar{I}_i^t + k_{ii}^t - \sum_{l=1 \ l \neq i}^{N} E(u_{li}^t)} \eta_i^t f_{\eta_i^t}(\eta_i^t)\, d\eta_i^t \right\} \\ & + \psi_{ij}^t \times \left\{ \left[-s_{i,max}^t + E(s_i^{t-1}) + \bar{I}_i^t + \sum_{l=1, l \neq i}^{N} E(u_{li}^t) \right] \right. \\ & \times \left[\int_{s_{i,max}^t - \bar{I}_i^t + k_{ii}^t - \sum_{l=1 \ l \neq i}^{N} E(u_{li}^t)}^{\infty} f_{\eta_i^t}(\eta_i^t)\, d\eta_i^t \right] \\ & \left. + \int_{s_{i,max}^t - \bar{I}_i^t + k_{ii}^t - \sum_{l=1 \ l \neq i}^{N} E(u_{li}^t)}^{\infty} \eta_i^t f_{\eta_i^t}(\eta_i^t)\, d\eta_i^t \right\}. \end{aligned} \quad (3.46)$$

If spillage causes some losses (e.g., destruction or flood) downstream, the expected value of benefit or cost might be calculated instead as in Appendix B.

The first moment for the second reservoir in two serial reservoirs (Part b of Figure 3.2) is derived using these new estimations for the expected value of the releases (Equations 3.44 and 3.46) in Appendix C.

The expected value of release should be initially derived for the most-upstream reservoirs where the natural inflow is the only stochastic input variable, and there are no releases coming from others. In other words, before setting the equations (i.e., constraints and the related terms in the objective function) for every reservoir, the expected value of releases pertinent to those reservoirs which start upstream of that reservoir should be first calculated.

In the above optimization models, the release policy pertinent to every reservoir only depends on its storage level, as a random variable with unknown distribution ($u_{ij}^t = k_{ij}^t + s_i^t$). There is another approach to create the stochasticity in the release policy such that the storage levels of all directly-connected upstream reservoirs to every reservoir are involved in the decision-making, in addition to its own storage. This extension of the FP model will be highlighted in Appendix D.

Until now, in the two approaches of the FP model, inflows have been assumed to be normally-distributed. In the next section, this assumption will be relaxed by using a generalized distribution called the Double-Bounded Probability Distribution Function (DB-PDF) [75].

3.7 FP model with non-Gaussian distribution

In this section, a generalized beta-equivalent distribution developed by Kumaraswamy [75] is substituted for the Gaussian distribution. Before getting into the detail of this method, a brief introduction of this distribution is presented in the next section.

A summary of Double-Bounded Probability Density Function (DB-PDF)

Kumaraswamy [75] suggested a beta-equivalent distribution function which he called the Double-Bounded Probability Density Function (DB-PDF). This is now more commonly

called the Kumaraswamy distribution [76]. This function is able to consider different varieties of shapes of distributions for random variables with lower and upper bounds. For a more recent review of this distribution, see Jones [76]. The general form of the density function for random process z using four different parameters κ_1, κ_2, z_{min}, z_{max} is represented as follows

$$\begin{aligned} f(x) &= \kappa_1\kappa_2 x^{\kappa_1-1}(1-x^{\kappa_1})^{\kappa_2-1}, \\ x &= \tfrac{z-z_{min}}{z_{max}-z_{min}} \qquad x \in [0,1], \end{aligned} \tag{3.47}$$

where κ_1 and κ_2 are two parameters playing the main role to create different functions such as exponential- or normal-shape distribution, and z_{min} and z_{max} are respectively the lower and upper bounds of random variable z.

Deriving the Double-Bounded Cumulative Density Function (DB-CDF) from the density function (equation 3.47) is a very straightforward calculation, unlike in the Beta distribution, which can be written as

$$F(x) = 1 - (1 - x^{\kappa_1})^{\kappa_2}. \tag{3.48}$$

The nth moment of x can be computed using the following equation:

$$\bar{x^n} = \frac{[(n/\kappa_1)!](\kappa_2!)}{(n/\kappa_1+\kappa_2)!} = \frac{[\Gamma(n/\kappa_1+1)](\Gamma(\kappa_2+1))}{\Gamma(n/\kappa_1+\kappa_2+1)} = \kappa_2 \times B(n/\kappa_1+1,\kappa_2), \tag{3.49}$$

where $B(\cdot)$ is the beta function.

With historical or simulated data for random variables, all parameters in the probability functions can be computed using either Algorithm 8 or 9 described here. Algorithm 8 was developed based on the method in [75].

The advantage of DB-PDF is the simple analytical forms for its DB-CDF, unlike the numerical integral required for beta CDF.

The formulation of non-Gaussian model

Using the historical data for inflows in one of Algorithms 8 or 9, the different parameters including κ_1, κ_2, which specify the shape of distribution, and z_{max} and z_{min}, which are

Algorithm 8 : Finding parameters of DB-PDF using moments

1: Calculate first four empirical moments from random samples z (empirical moments)

$$\bar{z^n} = mean(z^n) \quad \text{for} \quad n = 1, \cdots, 4.$$

2: Calculate theoretical moments as a function of four parameters κ_1, κ_2, z_{min}, z_{max}

$$\begin{cases} \bar{z'} &= z_{min} + \bar{x} * \xi, \\ \bar{z'^2} &= z_{min}^2 + 2\bar{x}\xi z_{min} + \bar{x^2}\xi^2, \\ \bar{z'^3} &= z_{min}^3 + 3\bar{x}\xi z_{min}^2 + 3\bar{x^2} z_{min}\xi^2 + \bar{x^3}\xi^3, \\ \bar{z'^4} &= z_{min}^4 + 4\bar{x}\xi z_{min}^3 + 6\bar{x^2} z_{min}^2\xi^2 + 4\bar{x^3} z_{min}\xi^3 + \bar{x^4}\xi^4, \end{cases}$$

where ξ=z_{max}-z_{min} and $\bar{x^n}$ can be calculated using Equation 3.49.

3: Estimate the parameters using theoretical and empirical moments (e.g., least squares method).

the estimations of the minimum and maximum of the inflow in real data, respectively, are determined for all time periods. Depending on the value of κ_1 and κ_2, the distributions of inflow will vary from one period to another (e.g., in November, the inflow may have an exponential behavior and it changes to normal in December) while they have been considered normal for all periods in the previous models. Moreover, in our new formulation, in contrast to the other three models, inflow is not split into constant and random values ($\bar{I}_i^t$, the mean of inflow, and η_i^t, a zero-mean random variable). Here, I_i^t plays the role of inflow as a random variable. Because the density and cumulative probability functions in this new distribution are the function of new random variable x, the moments should be computed using the following linear transformation

$$x_i^t = \frac{I_i^t - z_{min}^t}{z_{max}^t - z_{min}^t} \quad \Rightarrow \quad I_i^t = x_i^t \cdot (z_{max}^t - z_{min}^t) + z_{min}^t. \tag{3.50}$$

Suppose that the total water flowing from upstream to downstream reservoirs is de-

Algorithm 9 : Finding parameters of DB-CDF using an empirical CDF

1: Sort the data (the samples of z as a random variable) in an ascending order, i.e., $z_1 \leq z_2 \leq \cdots \leq z_M$.

2: An empirical CDF such as

$$\hat{F}(z_j) = \frac{j}{M+1} \quad \text{for} \quad j = 1, \cdots, M.,$$

is formulated.

3: Use these estimations to create a new data set $\hat{z}$ based on the inverse function of Equation 3.48

$$\hat{z}_j = z_{min} + (z_{max} - z_{min})[1 - (1 - \hat{F}(z_j))^{1/\kappa_2}]^{1/\kappa_1} \quad \text{for} \quad j = 1, \cdots, M$$

4: Estimate the parameters using z_j and $\hat{z}_j$ and a suitable norm (e.g., least squares method) .

termined using the release policy based on Approach 1 (no deficit and spillage included in the release policy that is $u_{ij}^t = k_{ij}^t + s_i^t$), the dynamics can be written as:

$$\begin{aligned} s_i^t &= \left\{ x_i^t \cdot (z_{i,max}^t - z_{i,min}^t) + z_{i,min}^t - k_{ii}^t + \sum_{l=1 \ l \neq i}^{N} (k_{li}^t + s_{li}^{t-1}) \right\} \cdot 1_{[s_{i,min}^t, s_{i,max}^t]}(\hat{s}_i^t) \\ &+ \{s_{i,min}^t\} \cdot 1_{(-\infty, s_{i,min}^t)}(\hat{s}_i^t) + \{s_{i,max}^t\} \cdot 1_{(s_{i,max}^t, \infty)}(\hat{s}_i^t). \end{aligned} \tag{3.51}$$

The moments can be derived analogous to what have been performed in other models; however, at the moment, numerical integrations are used. The following formulations represent the moments as constraints in this type of modeling

$$\begin{aligned} E(s_i^t) &= \left[z_{i,min}^t + K_i^t\right] \cdot \int_{L_1}^{L_2} f_{x_i^t}(x_i^t)\, dx_i^t + \left[z_{i,max}^t - z_{i,min}^t\right] \int_{L_1}^{L_2} x \cdot f_{x_i^t}(x_i^t)\, dx_i^t \\ &+ \left[s_{i,min}^t\right] \cdot \int_0^{L_1} f_{x_i^t}(x_i^t)\, dx_i^t + \left[s_{i,max}^t\right] \cdot \int_{L_2}^{1} f_{x_i^t}(x_i^t)\, dx_i^t, \end{aligned} \tag{3.52}$$

$$
\begin{aligned}
E(s_i^t)^2 &= \left[z_{i,min}^t + K_i^t\right]^2 \cdot \int_{L_1}^{L_2} f_{x_i^t}(x_i^t)\, dx_i^t + 2\left[z_{i,max}^t - z_{i,min}^t\right] \cdot \left[K_i^t + z_{i,min}^t\right] \\
&\times \int_{L_1}^{L_2} x \cdot f_{x_i^t}(x_i^t)\, dx_i^t + 2\left[z_{i,max}^t - z_{i,min}^t\right]^2 \int_{L_1}^{L_2} x^2 \cdot f_{x_i^t}(x_i^t)\, dx_i^t \qquad (3.53) \\
&+ \left\{s_{i,min}^t\right\}^2 \cdot \int_0^{L_1} f_{x_i^t}(x_i^t)\, dx_i^t + \left\{s_{i,max}^t\right\}^2 \cdot \int_{L_2}^{1} f_{x_i^t}(x_i^t)\, dx_i^t,
\end{aligned}
$$

where K_i^t can be calculated using Equation 3.32 and the lower and upper bounds, L_1 and L_2, are

$$
\begin{cases}
L_1 = \frac{S_{i,min}^t + K_i^t - z_{i,min}^t}{z_{i,max}^t - z_{i,min}}, \\
L_2 = \frac{S_{i,max}^t + K_i^t - z_{i,min}^t}{z_{i,max}^t - z_{i,min}}.
\end{cases}
$$

Because of using approximation techniques in finding $z_{i,min}^t$ and $z_{i,max}^t$, the transformed lower and upper bounds (L_1 and L_2) in the above integrals might be invalid (i.e., the lower or upper bound becomes less than zero or higher than one) because they are approximated during the optimization process. To overcome this issue, the following rules should be considered in setting the bounds:

$$
\begin{cases}
L_1 < 0 \;\Rightarrow\; L_1 = 0, \\
L_2 \le 0 \;\Rightarrow\; L_1 = L_2 = 0 \;\Rightarrow\; \int_{L_1}^{L_2} [\cdots] = 0, \\
L_2 > 1 \;\Rightarrow\; L_2 = 1, \\
L_1 \ge 1 \;\Rightarrow\; L_1 = L_2 = 1. \;\Rightarrow\; \int_{L_1}^{L_2} [\cdots] = 0
\end{cases}
$$

Moreover, if the bounds get the maximum and minimum values (i.e., $L_1 = 0$ and $L_2 = 1$), the analytical procedure using Equation 3.49 can be performed to calculate the integrals

in the moments equations. The numerical integrals in all experimental results in the later section are calculated using "*quadgk*" of Matlab.

So far, we only applied the FP models in reservoir management case studies. In the next section, we would like to demonstrate how to use these techniques to model a warehouse problem.

3.8 FP model in warehouse management

As has been mentioned in Chapter 2, the purpose of a warehouse optimization problem is to determine an ordering policy such that it minimizes the total cost.

In most traditional warehousing problems, the ordering time, in addition to ordered quantity, is considered as a decision variable, because of existing randomness in the lead time variable. However, if this variable is deterministic, the ordered quantity can play the role of sole decision variable. Given this assumption, the stochastic decision can be defined based on one of the approaches developed for stochastic release. For example, the stochastic decision based on Approach 1 can be written as follows:

$$u_{ij}^t = k_{ij}^t - s_j^{t-1}, \tag{3.54}$$

where s_j^{t-1} is equivalent to storage level j for which the order has been placed.

Figures 3.4 illustrates networks consisting of four distribution centers, with some retailers in their downstream in which each retailer has one demand point.

The solid triangles and rectangles represent the various storage spaces (the distribution centers and the retailers, respectively) which are supposed to be involved in the optimization problem. The dotted rectangles in the most-upstream layers, which are denoted as external sources (suppliers), are assumed to be unlimited resources (e.g., open markets) which can supply the amount of quantities ordered from distribution centers. Other dotted rectangles in the most-downstream layer of the networks are demand points with stochastic demands. The bidirectional arrows indicate possible transfer between two storage spaces. The unidirectional arrows, on the other hand, illustrate the flows of products or items from one place to another where the reverse flows are meaningless

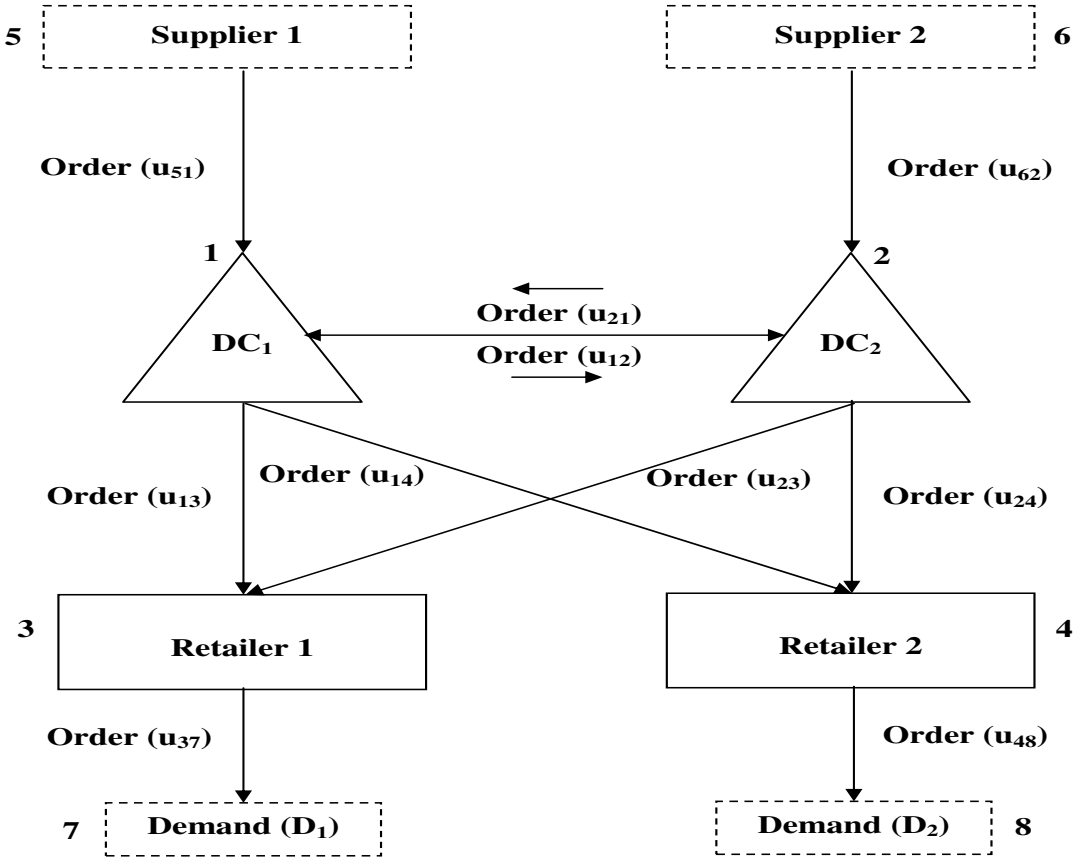

Figure 3.4: The network of warehouses and retailers

(e.g., flows from distribution centers toward external sources) or have not been defined for some feasibility reasons. Furthermore, because the suppliers and the demand points are not going to be managed in the optimization problem, they are called virtual spaces. The balance equation in the warehousing problem is similar to the reservoir application; however, it can be written for distribution centers and retailers in a different way as follows:

$$\text{distribution center:} \quad S_i^{t+1} = S_i^t + u_{ii}^t - D_i^t, \tag{3.55}$$

$$\text{retailer:} \quad s_j^{t+1} = s_j^t + u_{jj}^t - d_j^t, \tag{3.56}$$

where D_i^t and d_j^t are the stochastic demands for Distribution center i and retailers j, respectively, and u_{ii}^t or u_{jj}^t are the total quantity ordered by source i or j from other

sources. The stochastic demands for distribution centers are quite different from those used in the moment equations for retailers, since the features of the stochastic demands such as mean, standard deviation, or density function are given as inputs for retailers, rather than for distribution centers. In other words, the stochastic demands at the retailers should be transformed to the upper levels where distribution centers are. The following approximation formula might be used for finding the stochastic demands for these storage spaces in the case of no connection between retailers, and assuming that each retailer can send an order only to one distribution center:

$$D_i^t = \sum_{l=1 \ \& \ l \neq i}^{N} (d_l^t - s_l^{t-1}) \times \delta_{il}, \tag{3.57}$$

D_i^t, as the stochastic demand in distribution center i, is calculated using all stochastic demands (d_l^t for $l = 1 \cdots N$) which can place orders with source i and δ_{ij}, as an element of the routing in warehousing, can be defined in a similar fashion to that of reservoirs as follows:

$$\delta_{ij} = \begin{cases} 1 & \text{if } j^{th} \text{ source can place an order from } i^{th} \text{ source} \\ 0 & \text{otherwise.} \end{cases} \tag{3.58}$$

When the retailers can place orders with different distribution centers, their demands might be distributed between all respective distribution centers in a proper way.

As it is clear, the summation of some random variables appear in Equation 3.57. Based on the normality assumption for all demands in the FP model in Approach 1, the respective summation can be assumed to be a normal distribution with the following statistical features

$$\begin{aligned} E(D_i^t) &= \sum_{l=1 \ l \neq i}^{N} \left[E(d_l^t) - E(s_l^{t-1})\right] \times \delta_{il}, \\ \sigma^2(D_i^t) &= \sum_{l=1 \ l \neq i}^{N} \{\sigma^2(d_l^t) + \overbrace{[E(s_l^{t-1})^2 - \left(E(s_l^{t-1})\right)^2]}^{\sigma^2(s_l^t)}\} \times \delta_{il}, \end{aligned} \tag{3.59}$$

The respective expected value of all indicator functions and the related integrals in the first and the second moments can be computed analogous to what was done in the

previous sections.

In the case of Non-Gaussianity of demands, the suitable distributions should be fitted using one of the Algorithms 8 or 9 described in section 3.7. The fitting process for distribution centers should be implemented for the summation of all demands of the respective retailers.

3.9 Summary

In this chapter, we demonstrated how to apply RL techniques in storage management and developed type I and type II opposition schemes to speed up the learning process of RL techniques. A new version of type II opposition learning using knowledge extracted from a trained neural network was introduced. We also presented a new methodology to find the variance of the discounted accumulated reward and embedded it with risk-aversion factor β into the standard objective function for all learning methods under investigation.

Nonlinear stochastic FP models have been developed for multi-storage applications using different approaches for finding the expected value of releases from upstream reservoirs (the quantity of items to be ordered in warehousing problems). Moreover, the normality assumption in the original FP models has been relaxed by using a beta-equivalent distribution.

In the next chapter, these techniques will be tested on various case studies.

Chapter 4

Experimental Results

4.1 Introduction

In the following sections, both RL techniques and FP models are applied to some reservoir applications, single- and multi-reservoir, as a proper representation of storage management problems. Furthermore, to investigate the efficiency of developed opposition-based RL techniques under risk consideration, a grid-world problem as a common example in the RL literature is chosen for the experimental results.

4.2 RL methods

To verify that the RL techniques using their opposition-based versions can speed up the process of learning, two different applications are considered: A grid-world application with a stochastic environment which is a typical example in RL [59, 77], and a single reservoir application as a practical problem as an example of storage management [16, 40, 41]. We will also demonstrate how to minimize risk, in addition to maximizing the expected value of accumulated reward, as the formal objective function of RL techniques for these two case studies.

The following assumptions are used for both applications:

- We have only focused on opposite actions based on type I in the first application and type II for the second application. However, these experiments could be performed for opposite action and opposite state simultaneously using both opposition schemes.

- The learning parameter α in the grid-world application changes at a rate of $\frac{1}{N}$ ($\alpha < 1$) during the learning process and N is the number of observations for each action-state pair ($N > 1$). However, the learning rate α is constant in the reservoir application.

- In both applications, we assumed that there is no dependency between the action taken and the values of random variables specified after each interaction with the environment.

- In order to assess the efficiency of the learning methods, we used the SDP method to generate global optimum policies for both applications and compared these methods in terms of selected performance criteria obtained through the simulation using respective stationary or optimal policies.

- The parameter λ in Q(λ) and sarsa(λ) is set to 0.5 for both applications. The parameter $\lambda = 0$ corresponds to Q-Learning and λ=1 corresponds to full Monte-Carlo simulation .

4.2.1 Grid-world application

The stochastic grid-world application (Figure 4.1) is chosen from the website prepared by Poole in 2003 [77], with a 10×10 grid with standard moves including *up*, *down*, *right*, and *left* by one square. If the agent takes one of these actions, there exists a chance of 70% of going in that direction (true direction); however, there is a 30% chance of choosing another from the three remaining actions with an equal probability (about 10% for each action). The agent is punished with -1 if it hits the outside border of the grid. There is no punishment or reward for other cells, except those which are illustrated in Table

Table 4.1: Reward/punishment for 10×10 grid-world

cells	**(9,8)**	**(8,3)**	**(4,5)**	**(4,8)**
Reward/punishment	10	3	-5	-10

4.1. The agent receives reward or punishment right after taking every action in these respective cells (states) shown in the table. Moreover, it is assumed that if the agent receives reward, it will jump randomly to one of the corners. The objective function here is to maximize the accumulated reward for all possible states (i.e., finding the maximum value functions for all given states).

We made one change to this problem to make it more complex:

- instead of having a standard move, we consider King's moves [45] including eight different directions as shown in Figure 4.1.

As shown in Figure 4.1, the actions are labeled in a specific fashion, such that the result for opposite action based on type I is the same as Equation 2.9 for finding the opposite action. For example, if the action *up* (action number 1) is chosen, its opposite using this equation will be $1 + 8 - 1 = 8$ or *down* (action number 8).

Another issue in this problem is to find the next state when the opposite action is chosen. Actually, it should be extracted using the action taken by the agent. Two situations might occur:

- If the agent went in the desired direction, it can be reasonably assumed that the agent goes in the desired direction if the opposite action is chosen.
- If the agent did not go in the desired direction for the action taken, it can be assumed that the agent goes in the wrong direction if it takes the opposite action. Assume that the agent takes an action, but it goes in an undesired direction in one iteration. The agent can memorize this direction, and use this knowledge to find out what would happen if it took the opposite action in that iteration.

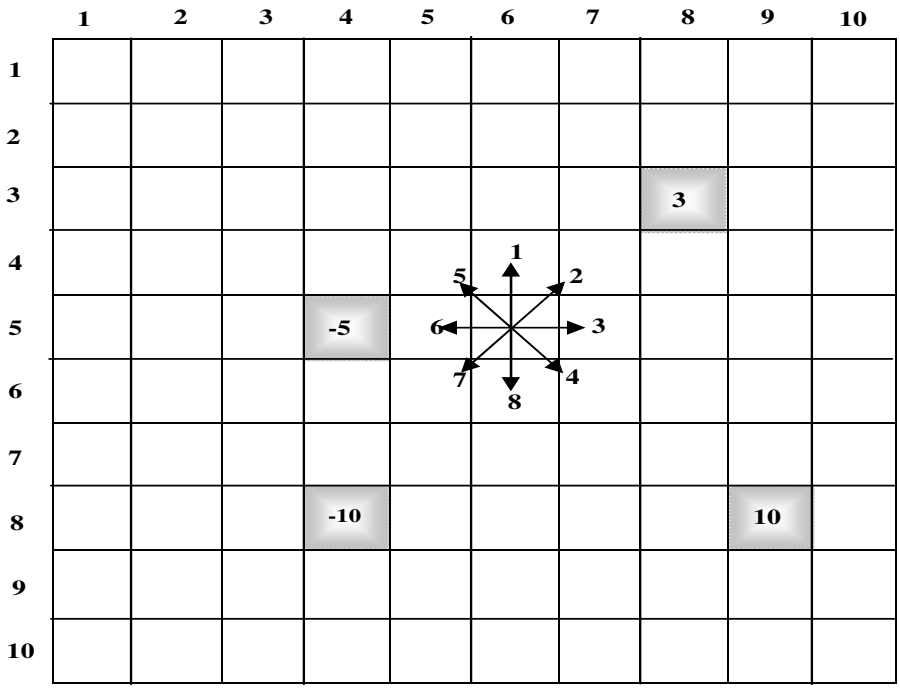

Figure 4.1: Grid-world application with king's moves

We have used a simulation program to evaluate the sub-optimal or the trained policy (i.e., the policy evaluation step) to calculate the following performance criteria:

- **The Distance of Action-Value functions (DAV)**: the action-value functions and value function are again approximated using the stationary policies extracted at the end of the learning process through a simulation routine. The distance between these approximations and the true value functions extracted by SDP can be a new criterion. This can be mathematically formulated as follows:

$$D = \sum_{i} |V^{\pi'}(i) - V^{\pi}(i)|, \tag{4.1}$$

where π' is the stationary policy at the end of the learning technique, π is the optimal policy extracted by SDP, and V represents the value function.

It is obvious that the smaller the distance, the higher the performance of the respective technique.

- **The Expected Reward (EXR)**: one way to measure the performance of a stationary or optimal policy is to find the expected value of reward for each episode. An episode in most Markov Decision Processes (MDPs) such as grid-word applications is defined based on the goal states (also called terminal or absorbing states). Usually, when the agent reaches one of these states, one episode is completed and the next episode should be started. In the respective experimental results, 100 iterations constitute one episode.

- **The Summation of Standard deviation (SOS)**: In the case of minimizing risk in addition to maximizing average accumulated reward, the standard deviation of accumulated discounted reward is obtained through Equation 3.12. The summation of all variances pertinent to the actions of trained policy could be used to measure the extent of variability of the accumulated discounted reward:

$$D = \sum_{i} \sigma(i, a^{\pi'}) \tag{4.2}$$

  where $\sigma(i, a^{\pi'})$ is the standard deviation of accumulated reward for state i and action a taken by policy π' which has been derived from the learning process.

Since we are using random data generation in the learning process in our grid-world application, the policy and the value functions could be different at the end of each complete learning episode/period. Therefore, to have a better assessment, we performed the learning process and the evaluation step multiple times (here is 20 times) for each set of learning parameters, and calculated the mean and variance.
Other experimental settings are given as follows:

- We chose a ϵ-greedy policy with arbitrary value, say $\epsilon = 0.2$, and a Boltzmann policy in which the temperature parameter τ was initialized with the maximum number of episodes, for example, as in Table łTAB:3, decayed by two after each episode, and set to one if it became zero or negative.

Table 4.2: Performance criteria for SDP in a 10×10 grid world application

Criterion	DAV	EXR
domain ranges	292.8±42.33	83.8±2.1
average	302	84.2
Standard deviation	25.86	1.4

- Because the immediate reward for most action-state pairs is zero (immediate reward is zero for all moves except those which lead to squares shown in Table 4.1), most action-value functions are not updated at the beginning of learning. Therefore, to increase the accuracy of the learning, the learning step α only changes if the pertinent action-value function is updated to a non-zero value; otherwise the number of observations for that action-state pair does not affect α.

- The performance of each learning method and its opposition version are compared based on the median and the range. Alternatively, the mean and the standard deviation are used for comparisons.

- Table 4.2 shows the respective performance criteria pertinent to SDP in which the optimal policy is used in a simulation routine for multiple times using different sequence of data. DAV denotes the total distance between the true and the approximate value functions obtained through the optimization and the simulation, respectively. It can be used as an upper bound for computing the mentioned criteria.

As demonstrated in Tables 4.3 and 4.4, for two small and large number of episodes (20 and 200), the type I opposition learning has considerably contributed to the improvement in speed and the accuracy of the learning process in both learning methods (Q-Learning and Q(λ)), in which the respective performance criteria are better than the regular learning. The opposition superiority is also demonstrated in Figures 4.2 and 4.3 for the distance of action-value functions (DAV), for the two learning methods with different numbers of episodes.

Based on observations in the experiments, and partially illustrated in the tables and figures, the advantages of the opposition learning schemes become apparent especially during the early stage of learning (episodes less than 20). This means that a desired accuracy can be achieved in a shorter time, which is equivalent to algorithm acceleration. Moreover, it seems that a Boltzmann policy has better results compared to a ϵ-greedy one. The reason might be that since the reward in this application only propagates through some limited cells (states (9, 8) and (8, 3)), it takes time to influence other cells with longer distances from these cells. Therefore, it turns out that the learning process in this case (with delayed reward) needs more exploration than exploitation, especially at the beginning of the learning process. The Boltzmann policy, with the possibility of tuning the temperature parameter τ to set the extent of exploration and exploitation, could increase the accuracy of learning for both small and large numbers of episodes. The results in Tables 4.3 and 4.4, in which the best option for each criterion is specified in bold, verifies this fact for most defined performance criteria in terms of the variation domains in both Q-Learning and Q(λ).

Furthermore, the results of performance criteria for learning methods using a Boltzmann policy are close to upper boundary values for a grid world application, obtained from SDP and shown in Table 4.2.

Table 4.3: Performance criteria of Q-Learning in a 10×10 grid world application

Episode		**20**		**200**	
Criterion		**DAV**	**EXR**	**DAV**	**EXR**
ϵ-greedy	**QLR**	593±109	24.9±13.9	352±45	72±7.4
	OQLR-I	**459±55**	**46.4±14.2**	**349±51**	**76.2±4.8**
Boltzmann	**QLR**	615.1±52	24.6±8.4v	286±52	75±5.3
	OQLR-I	**479±58**	**40.59±11.4**	**307±43**	**81.1±3.6**

QLR: Q-Learning, **OQLR-I:** Type I Opposition Q-Learning

In the case of penalizing objective function with variance in an off-line learning situation, the average distances of action-value function (DAV) and average summations of

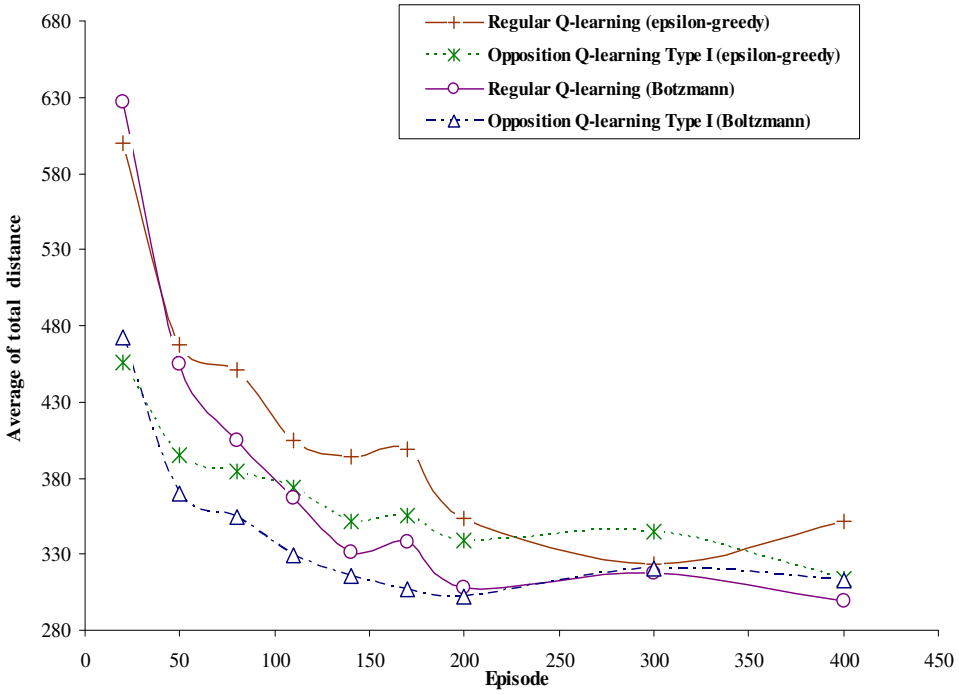

Figure 4.2: DAV in Q-Learning for grid world application

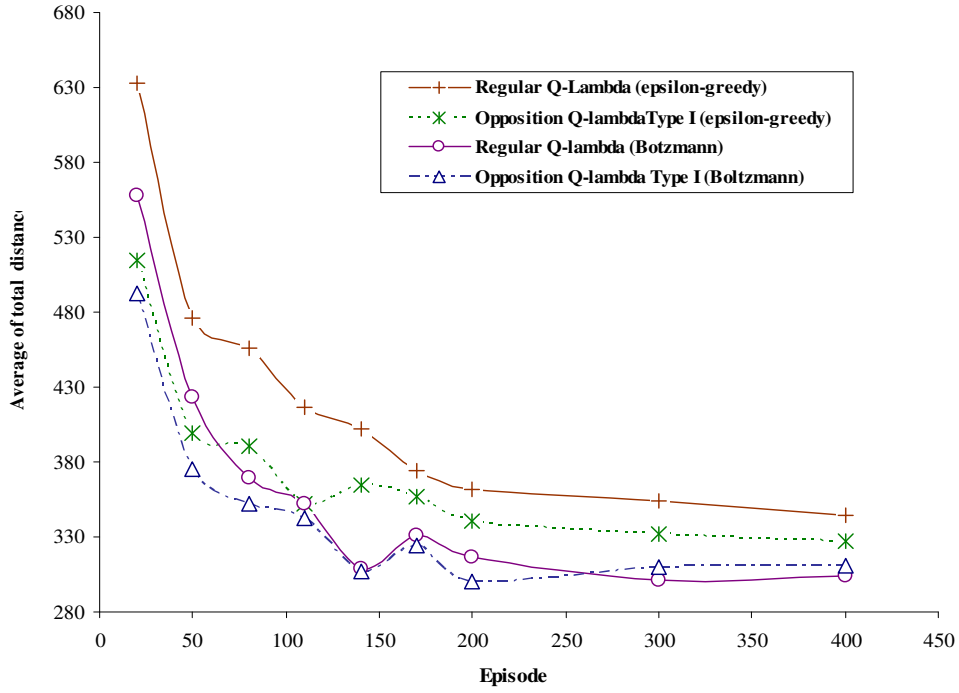

Figure 4.3: DAV in Q(λ) for grid world application

Table 4.4: Performance criteria of Q(λ) in a 10×10 grid world application

Episode		**20**		**200**	
Criterion		**DAV**	**EXR**	**DAV**	**EXR**
ϵ-greedy	**QLA**	636±80.5	15±20.1	348±48	71.1±5.9
	OQLA-I	**519±41**	**39.4±8.2**	**333±35**	**73.1±3.5**
Boltzmann	**QLA**	558±41	33.1±16	309±40	77.1±2.1
	OQLA-I	**506±128**	**42.7±18.3**	**288±47**	**81.1±2.4**

QLA: Q(λ), **OQLA-I:** Type I Opposition Q(λ)

standard deviation (SOS) have a decreasing trend as the risk-aversion factor β increases. Figures 4.4-4.5 illustrate these relatively smooth decreasing curves for both mentioned criteria.

It is also obvious that the results related to risk-neutral control (β=0) are almost the same in terms of different performance criteria for all policies derived from both learning methods and their opposition versions. However, by embedding the variance term into the objective function and increasing β, some changes in these performance measures (their domain ranges) in favor of opposition learning schemes have appeared. For example, the opposition versions for both learning methods have superior results in terms of DAV for all βs compared to the regular learning methods (Figure 4.4). A more interesting point in this figure is that the respective opposition version of Q(λ) has the best value of DAV (i.e., the lowest one) for all βs considered in our experiments, and even better values for SOS criterion compared to regular Q(λ) for some βs (e.g., where $\beta = 0.8$, the values of DAV and SOS for regular Q(λ) are 465.8 and 108.53, while they are 449.9 and 101.38 for type-I opposition version of Q(λ), respectively). Table 4.5 demonstrates the changing domain of the average expected reward (EXR) in a risk penalizing objective function. These results also verify the relative preference for opposition Q(λ), OQLA, compared to Q-Learning.

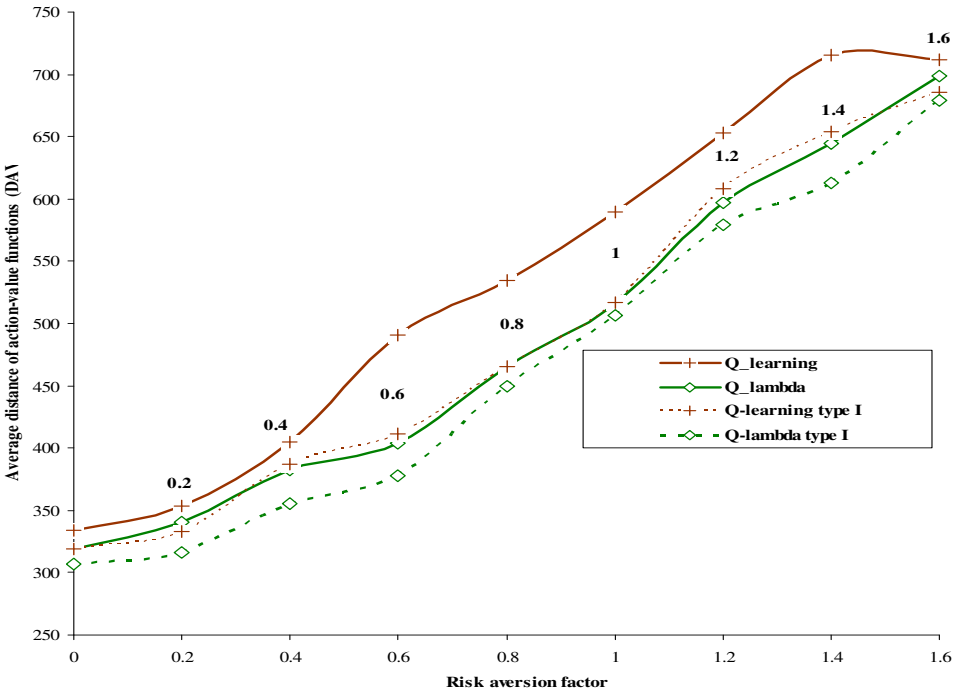

Figure 4.4: DAV for risk consideration in grid world application

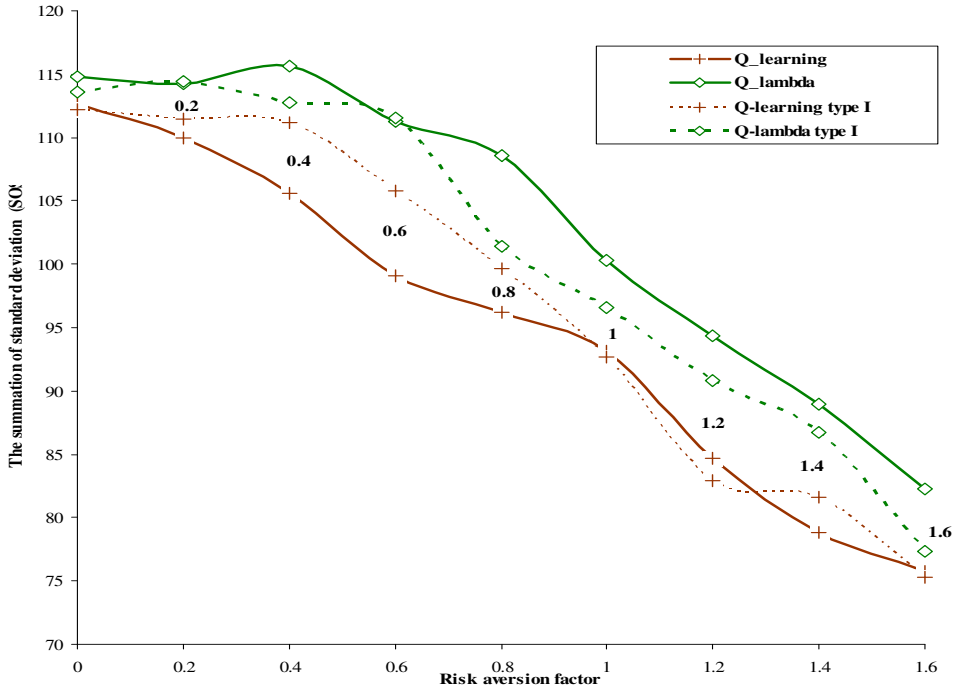

Figure 4.5: SOS for risk consideration in grid world application

Table 4.5: The average expected reward (EXR) in a risk-penalized objective function for grid world application

learning	β								
method	0	0.2	0.4	0.6	0.8	1	1.2	1.4	1.6
QLR	69.7±13.9	69.5±10.5	57.5±19.7	42.3±21.8	42.3±17	30.2±18.4	15.4±28.6	17.3±28	8.5±18.4
OQLR-I	**78.1±6.4**	**77.3±5.8**	**67.7±13.1**	**59.9±15.8**	**57.7±13.5**	**38.9±26.2**	**32.5±27.1**	**22.2±21.5**	**11.5±24.4**
QLA	**78.8±3.9**	76.7±6	61.2±19.8	57.7±18.8	44.2±19.4	34±23.4	29.3±16.5	17.9±24.6	7.4±16.9
OQLA-I	78.8±4.9	**74.9±8.7**	**71.8±8**	**68.8±9.2**	55.8±13	**46.9±16.9**	**32.6±20.5**	**25.7±18.9**	**12.6±22.2**

QLR: Q-Learning, **OQLR-I:** Type I Opposition Q-Learning **QLA:** Q(λ), **OQLA-I:** Type I Opposition Q(λ)

4.2.2 Reservoir management

In this section, a single-reservoir application from Fletcher and Ponnambalam [16] is investigated in two different situations:

1. Inflow is the only stochastic variable in the system model. In this situation, all mentioned RL techniques with their opposition versions are compared to each other. The purpose of these experiments is to demonstrate the effect of opposition-based learning in speeding up the learning process, and to find out which RL techniques have more efficient solutions. All respective comparisons are also performed where the risk is embedded into the mathematical models.

2. In addition to inflow, the energy price is considered as a stochastic variable due to price deregulation and market privatization. In this more complex situation, the efficiency of Q-Learning in RL versus two other types of traditional techniques, including Two-Stage Stochastic Programming (TSP) and Fletcher-Ponnambalam (FP) modeling technique (with and without risk consideration), are compared.

One cycle in this problem is one complete year with 12 months. Minimum and maximum Storage and Release in different months of the year are given in Table 4.6.

Table 4.6: Maximum and minimum storage and release, average inflow, and demand

Value m^3	**Month**											
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Max. storage	8	8	8	8	8	8	8	8	8	8	8	8
Min. storage	1	1	1	1	1	1	1	1	1	1	1	1
Max. release	4	4	6	6	7.5	12	8.5	8.5	6	5	4	4
Min. release	0	0	0	0	0	0	0	0	0	0	0	0
Average inflow	3.4	3.7	5	5	7	6.5	6	5.5	4.3	4.2	4	3.7

Inflows of different months in a year are normally distributed and given in Table 4.6. Evaporation from the reservoir is ignored.
The objective function in Fletcher's model [16] is to maximize the total net benefit from releases. The benefit of release per/unit is given in Table 4.7.

Table 4.7: The benefit of release per unit for each month of a year

Month	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Benefit	1.4	1.1	1.0	1.0	1.2	1.8	2.5	2.2	2.0	1.8	2.2	1.8

When prices are stochastic, they are normally distributed and values in Table 4.7 represent the expected values of prices.
To compare the performance of the policies for four different learning methods, Q-Learning, Q(λ), sarsa, and sarsa(λ), the following criteria can be extracted from the simulation:

- The average of annual benefit (μ_B),
- The standard deviation of annual benefit (σ_B),
- The coefficient of variation ($C.O.V = \frac{\sigma_B}{\mu_B}$).

Other assumptions for this problem are summarized as follows:

- Storage volume as a state variable of the system and water release as the main decision variable (action) were discretized into 7 and 55 equal intervals, respectively. Therefore, the number of possible states was 8, and the number of total possible actions taken by the agent became 56.
- Immediate reward in this application is the product of the actual release (u^t) and the price per unit (c^t) in every time period. Where the inflow is the only stochastic

variable , the immediate reward has to be normalized in order to make the training of Neural Networks (NN), as explained below, more accurate when applying the type II opposition-based learning. The normalization is performed using the maximum value of u^t and c^t in every iteration as

$$r = \frac{u^t \times c^t}{\max\limits_t\{R^t_{max}\} \times \max\limits_t\{c^t\}}.$$

This is done to minimize problems arising from the training data not being a complete representation of the entire data.

- The number of years in each episode was set to 30.

- A Multi-Layer Perceptron (MLP) with one hidden layer and 20 nodes was used to approximate the respective action-value functions in order to find the type II opposite action using Equation 3.6. The tangent-sigmoid transfer function is used in every node including hidden and output layers and all desired outputs (action-value functions) are normalized in order to have values between -1 and 1. The gradient descent backpropagation technique is also used to train the network. The training process is performed using the Matlab(R) toolbox.

- In order to demonstrate the accuracy and efficiency of all methods, the extracted stationary policies were simulated for 2000 years under the same conditions in which the learning and optimization parts have been performed. Furthermore, because of using simulation, and of the possibility of ending up with different values for the respective criteria, each experiment was performed 10 times and the averages of all values were then used for comparison.

- C.O.V=0.3, for inflows.

- To find the admissible actions for every state, the optimistic scheme (in which the maximum inflow in every period is considered in the mass balance equation (Equation 1.2)) was employed [48].

- The agent takes actions according to an ϵ-greedy policy with $\epsilon = 0.1$ as in [48].

- In the case of considering risk where inflow is the only random variable, we added the standard deviation of action-value function, instead of variance to the objective function, because mean and standard deviation have similar scale of values. Risk-aversion factor β in this situation varied from 0.1, as a low risk-penalizing objective function, to 3, as a high risk-penalizing one, for all learning methods. However, when the price in addition to inflow is stochastic (the second situation), the variance is directly embedded into the objective functions and β ranges between zero and 30 in order to see all possible average annual benefits.

- In order to make sure that all action-state pairs will be sufficiently visited in the risk-penalizing objective function, we considered a large value for the number of episodes, equivalent to 900. The first training of the neural network was performed at the 100^{th} episode and repeated every 100^{th} episode thereafter. In every training a set of new data for training and testing based on the action-value functions was fed to the network.

- The target performance (the total average benefit), where inflow is the only stochastic variable, was set to 102.27. This performance was obtained through the simulation using the optimal policy extracted from the SDP method.

- In the version of opposition-based learning, the type I opposite action was extracted (Equation 3.4) before the first training of the function approximation, and then the opposite was chosen using a type II scheme (Equation 3.6) with approximate functions until the end of learning. Moreover, the training and testing data were produced using the average number of observations for all action-state pairs.

Inflows as uncertain

Figures 4.6-4.7 compare different learning methods and their type II opposition versions. As could be expected, the regular or opposition versions of sarsa (SLR) and sarsa(λ) (SLA) have lower average annual benefit compared to Q-Learning (QLR) and Q(λ) (QLA) for episodes 50 to 1500.

Furthermore, despite having a significant jump in Q(λ) compared to Q-Learning in the number of episodes equivalent to 50 in the regular learning (95.02 versus 91.69), Q(λ) produces slightly better results in terms of the average criterion for all episodes in both learning types, regular and opposition-based. However, their average annual benefits get closer as the number of episodes increases, such that for the maximum number of episodes, say 1500, they are almost the same ($\mu \approx$101). This might imply that the opposition learning scheme should be partially performed at the beginning of the learning process.

Moreover, the opposition versions of all learning methods have better results for a small number of episodes (e.g., 50 episodes). As shown in Figure 4.7, the variability of average reward for the opposition version of all four learning methods, especially for Q-Learning and Q(λ), are remarkably small compared to regular learning, illustrated in Figure 4.6 (i.e., the opposition version is more robust).

By embedding the risk into the respective objective function and increasing the value of the risk-aversion factor β, we expect to observe a decreasing trend in average and standard deviation of annual benefit during the simulation using the policies extracted from the learning part in the same situation. Tables 4.8-4.9, which illustrate the changing domain of these two criteria for both Q-Learning and Q(λ) for different β values, verify this fact.

Inflows and prices as uncertain

Now, we would like to compare Q-Learning as a well-known technique in RL with Two-Stage Stochastic Programming (TSP) and Fletcher-Ponnambalam (FP) models as nonlinear optimization models in which the stochastic nature of two different random variables (inflow and price) are explicitly embedded into their mathematical formulation. Moreover, the linear objective function used in our experiments related to this situation is the zeroth order approximation of Taylor's series for energy function, which is presented

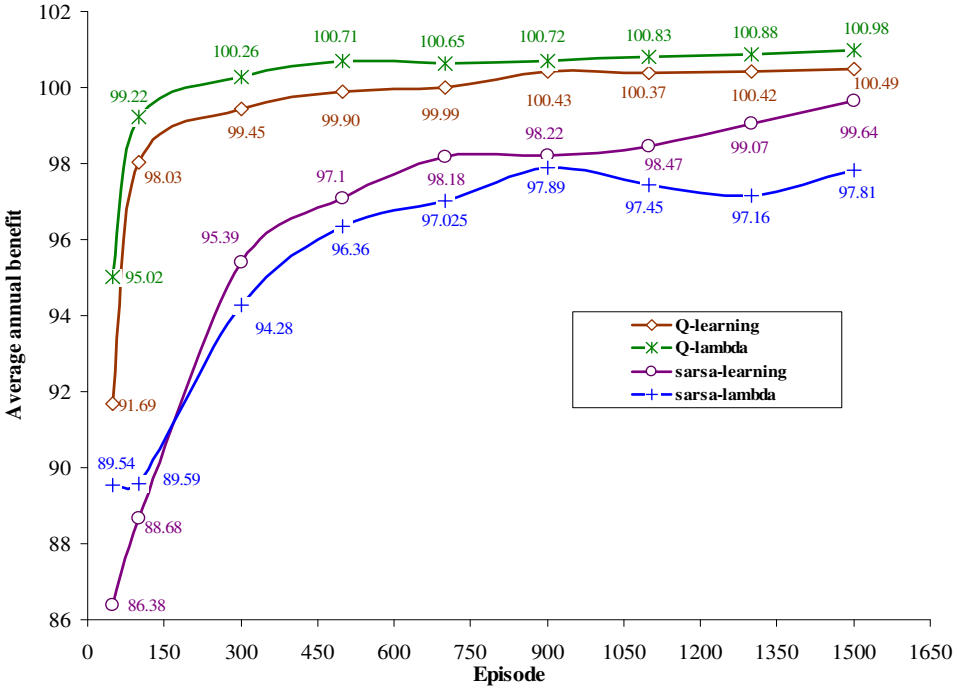

Figure 4.6: The comparison of average annual benefit in regular learning methods for reservoir application

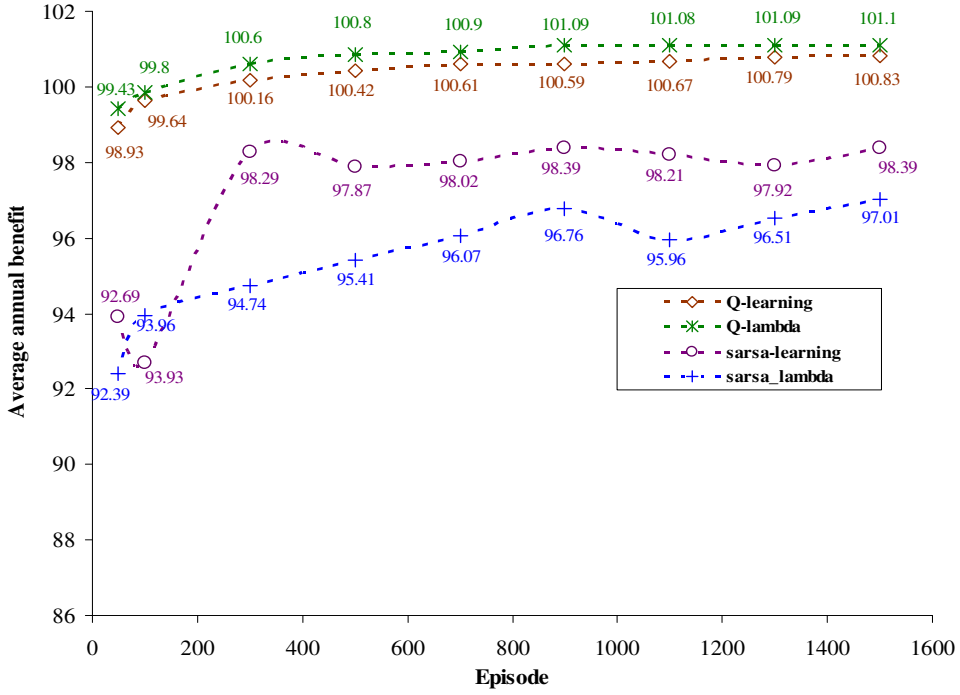

Figure 4.7: The comparison of average annual benefit in type II opposition learning methods for reservoir application

Table 4.8: The average annual benefit with risk-penalized objective function for different β

learning method	β								
	0.1	0.5	0.9	1.3	1.7	2.1	2.5	2.9	3.3
QLR	100±0.4	101±0.4	100±0.8	98±1	98±2	97±1	95±2	93±3.7	93±4
OQLR-II	**101±0.7**	**101±0.5**	**101±0.3**	**100±0.7**	**99±1**	**98±1**	**97±0.9**	**97±1**	**95.2±2.19**
QLA	101±0.5	100±0.5	99±0.6	99±0.9	**98±0.9**	97±1	**97±2**	**96±2**	**96±1**
OQLA-II	**101±0.8**	**101±0.6**	**100±0.7**	**99±0.5**	98±1.5	**96±1.6**	95±2	93±4	92 ±3.8

QLR: Q-Learning, **OQLR-II:** Type II Opposition Q-Learning **QLA:** Q(λ), **OQLA-II:** Type II Opposition Q(λ)

Table 4.9: The average standard deviation of annual benefit with risk-penalized objective function for different β

learning	β								
method	0.1	0.5	0.9	1.3	1.7	2.1	2.5	2.9	3.3
QLR	**8.8±0.4**	**8.3±0.7**	**7.5±0.6**	**7.1±0.3**	6.7±0.8	6.2±0.8	6.5±1	6.3±1.1	62±1.2
OQLR-II	8.7±0.4	8.4±0.4	8.2±0.2	7.5±0.6	**6.7±1.1**	**7.2±0.5**	**6.3±1.2**	**6.4±0.9**	**6.3±1.3**
QLA	**8.9±0.3**	**8.2±0.3**	**7.4±0.7**	**7.4±0.3**	**7.2±0.6**	**7.6±0.6**	7.4±0.6	7.3±0.4	7.23±0.56
OQLA-II	8.8±0.4	8.1±0.6	7.8±0.7	7.3±0.7	7.3±0.6	6.7±0.9	**6.6±0.8**	**6.7±1.1**	**6.4±0.8**

QLR: Q-Learning, **OQLR-II:** Type II Opposition Q-Learning **QLA:** Q(λ), **OQLA-II:** Type II Opposition Q(λ)

by Loucks et al. [9]. It can be written as:

$$\max_{s^t,u^t}\left\{f^t(s^t,u^t)\right\} \;=\; \max_{s^t,u^t}\left\{\sum_{t=1}^{T}(u^t\times C^t + C_s\times s^t)\right\}, \tag{4.3}$$

where u^t is the amount of release during period t, C^t is the unit price of water released during period t, and C_s is the constant value for storage volume which is equivalent to 0.1 in the experiments.

Objective function with risk consideration

By adding the risk terms in the objective function of Equation 2.12 in two-stage programming, the formulation will not be linear anymore (it is called Two-Stage Stochastic Programming (TSP)). This new objective function can be written as:

$$\begin{aligned}&\max_{u^t,s^t}\{f^t(s^t,u^t)\} = \\ &\max_{u^t,s^t}\left\{\sum_{t=1}^{T}\{(E(C^t)\times u^t + C_s\times s^t - \mu^t_{N1}) - \beta\times[(\sigma^t_{N1})^2 + \sigma^2(C^t)\times(u^t)^2]\}\right\},\end{aligned} \tag{4.4}$$

where $\sigma^2(C^t)$ is the variance of water price and μ^t_{N1} and $(\sigma^t_{N1})^2$ are defined as follows:

$$\mu^t_{N1} = \sum_{l=1}^{N_1} P^{t,l}\times(CSF^t\times SF^{t,l} + CSU^t\times SU^{t,l}), \tag{4.5}$$

$$(\sigma^t_{N1})^2 = \sum_{l=1}^{N1} P^{t,l}[(CSF^t\times SF^{t,l} + CSU^t\times SU^{t,l}) - \mu^t_{N1}]^2. \tag{4.6}$$

To minimize the variance or the standard deviation simultaneously with the expected value of benefits in the FP model, a new term should be added to Equation 2.21:

$$\begin{aligned}&\max_{E(s^t),u^t}\{f^t(E(s^t),u^t)\} = \\ &\max_{E(s^t),u^t}\left\{\sum_{t=1}^{T}\{E(C^t)\times u^t + C_s\times E(s^t) - \beta\times[\sigma^2(C^t)\times(u^t)^2 + C_s^2\times\sigma^2(s^t)]\}\right\}.\end{aligned} \tag{4.7}$$

The objective function in the learning methods is also calculated through Equation 3.11.

Assumptions and comparison results

Choosing a range between 0.01 and 0.5 for values of coefficient of variation of both random variables, we have investigated the results of TSP, FP, and Q-Learning (QLR) in terms of policy obtained from the optimization or learning phase, and results acquired from the simulation, in three main categories:

- low uncertainty for inflow with increasing uncertainty in price;
- low uncertainty for price with increasing uncertainty in inflow;
- increasing uncertainties of both inflow and price.

Table 4.10 demonstrates 9 different sets of uncertainties (coefficients of variation) for price and inflow covering the above three categories. Moreover, to achieve appropriate

Table 4.10: The set of coefficients of variation for inflow and price

Stochastic parameter	**Set No.**								
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Inflow	0.01	0.01	0.01	0.2	0.4	0.5	0.2	0.4	0.5
Price	0.01	0.2	0.5	0.01	0.01	0.01	0.2	0.4	0.5

performances in the simulation using the policy resulting from Q-Learning for different uncertainties specified in Table 4.10, the total number of episodes and the learning rate α have been set in such a way that the maximum annual average and minimum variance are achieved. It seems that for high uncertainties of price or inflow, lower learning rates and higher repetitions (the total number of episodes performed in learning process) are more suitable. However, for low uncertainties, the learning rate and the number of repetitions can be reasonably high and low, respectively. Therefore, to obtain a suitable policy, the values of these parameters have been set and shown in Table 4.11.
The action policy used in the learning process is ϵ-greedy with ϵ=0.1. Table 4.12 demonstrates the values of average annual benefit, its standard deviations, and the coefficients

Table 4.11: Learning parameter and the number of repetitions in Q-Learning

	Uncertainty for price or inflow		
parameter	<0.3	0.3≤ and <0.5	=0.5
Learning parameter	0.5	0.35	0.3
Total number of episodes	1000	2000	2000

of variation obtained through the simulation for three methods TSP, FP, and Q-Learning (QLR) in the mentioned categories (Table 4.10). There is no risk involved in the objective function; that is, the risk aversion factor β in the respective term in the objective function equals to zero.

Table 4.12: Performance criteria of TSP method for three categories

	Set	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
TSP	**Average**	108.9	110.32	95.8	102	102.3	99.8	104.1	81.3	62.3
	σ	0.54	7.34	13	7.3	11.78	10	10.1	9.15	8.1
	Cov	0.005	0.066	0.136	0.072	0.115	0.1	0.097	0.113	0.13
FP	**Average**	108.43	108.27	108.48	-	101.9	98.05	-	101.64	97.97
	σ	0.43	7.44	18.26	Not feasible	9.43	11.29	Not feasible	17.03	20.6
	Cov	0.004	0.069	0.169	-	0.093	0.115	-	0.167	0.21
QLR	**Average**	107.44	104.89	102.82	105.36	102.21	100.15	104.37	99.4	97.46
	σ	1.9	6.6	15.12	5.49	10.27	12.35	8.53	15.15	18.38
	Cov	0.018	0.063	0.147	0.052	0.101	0.123	0.082	0.152	0.189

As seen in Table 4.12, Q-Learning for most coefficients of variation of price and inflow has comparable results with those criteria obtained through the two other optimization techniques. However, the performance criteria in both the FP model and the Q-Learning method are closer to each other (e.g., for the higher uncertainty coefficients for price and inflow = 0.5 (set No. 9)), μ=97.46 and σ=18.38 in Q-Learning, μ=97.97 and σ=20.6 for FP, and μ=62.3 and σ=8.1 for TSP). Moreover, Q-Learning has a feasible solution for all different categories, whereas it seems that the constrained optimization method in *fmincon* of MATLAB(R) cannot solve the FP model in some cases (e.g., where the coefficients of variation for inflow and price are 0.2 and 0.01, respectively).

The trade-off of average accumulated reward versus the standard deviation is illustrated in Figure 4.8 for the three methods. This figure demonstrates that to obtain a specific

annual average benefit, we should properly set the risk factor, β, in the objective function of each stochastic method. Furthermore, it shows that all three methods give almost the same standard deviation during the simulation when their average annual benefits are identical, even though the release decisions are not the same. It also indicates that QLR, as a Monte Carlo-based optimization technique, can consider stochasticity and risk minimization as being as accurate as two other stochastic methods, which are promising results. The TSP method also uses scenarios like the QLR method, but it is not chosen further in this thesis. The FP method results are similar to QLR and will be considered further in this thesis, because this model does not need scenarios and is an explicit method.

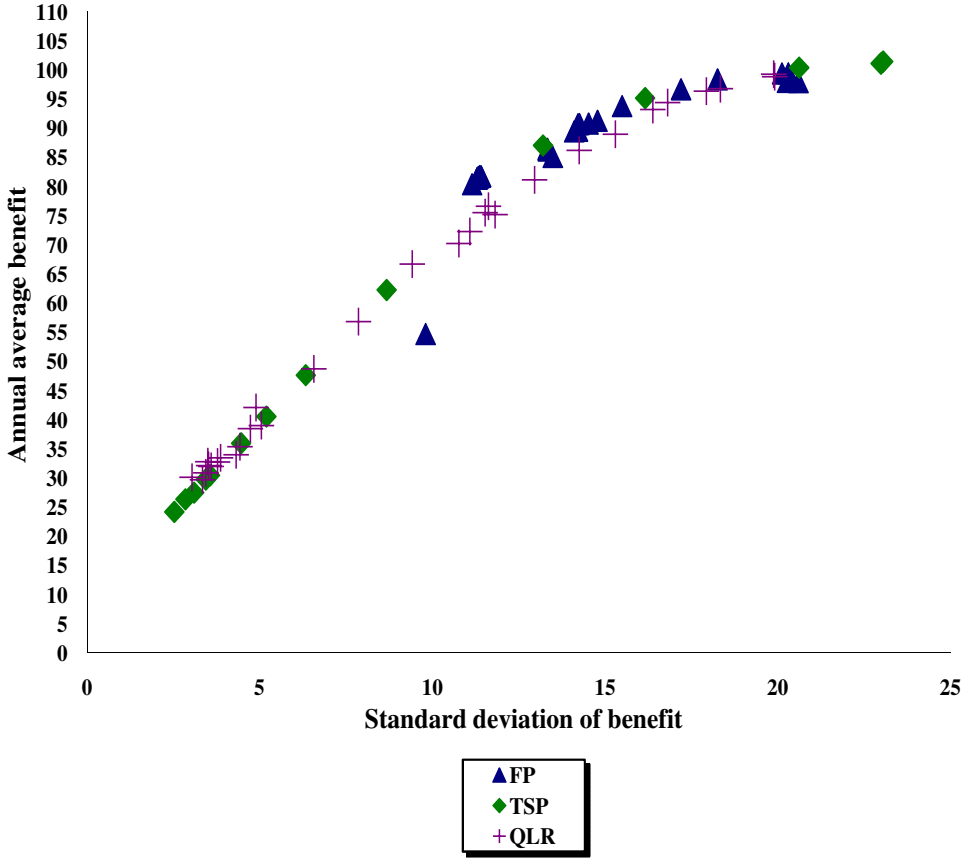

Figure 4.8: Trade-off between average annual benefit and its standard deviation for $0 \leq \beta \leq 30$ ($Cov(Inflow)$=0.5 & $cov(Price)$=0.5)

4.3 FP method

To investigate the performance of the FP method, we have chosen a large-scale problem, namely, the multi-reservoir system in India called the Parambikulam-Aliyar Project (PAP) [78]. Firstly, to study the various methods in detail, just the first two reservoirs in a serial configuration of the Parambikulam-Aligar Project (PAP) was chosen as the first case-study. Finally, to demonstrate the applicability of these types of modeling for large-scale applications in a stochastic situation, the modeling and the optimization part are performed for the entire system composed of a five-reservoir system from the PAP case study, and results are presented and discussed in the last section.

4.3.1 PAP case study

The PAP project is composed of nine reservoirs which are interconnected through different tunnels and canals (30km tunnels and 240km canals). The main purpose of this project was to satisfy the existing irrigation demands in the most-downstream reservoirs for two states (Tamilnadu and Kerala) by diverting the surplus flows of the rivers whose waters emanate from the western slopes of the Anamalai mountains. This system is also utilized to generate power at many dams and tunnels. The system is simplified to five reservoirs as illustrated in Figure 4.9 [78]. The numbers inside triangles in Figures 4.9 represent the active capacities, which can be considered as the maximum storage volume. Furthermore, the minimum storage volumes are assumed to be zero for all reservoirs and periods. Other necessary information which is mostly obtained from [78] are summarized as follows:

Inflows

The hydrological data are available for the period of 1971-1982 on a monthly basis for the five reservoirs. Tables 4.13-4.14 illustrate the statistical values for all five reservoirs. These observations can also be used to find the suitable distribution functions of inflows using one of the algorithms in the previous chapter (Algorithm 8 or 9) and are described

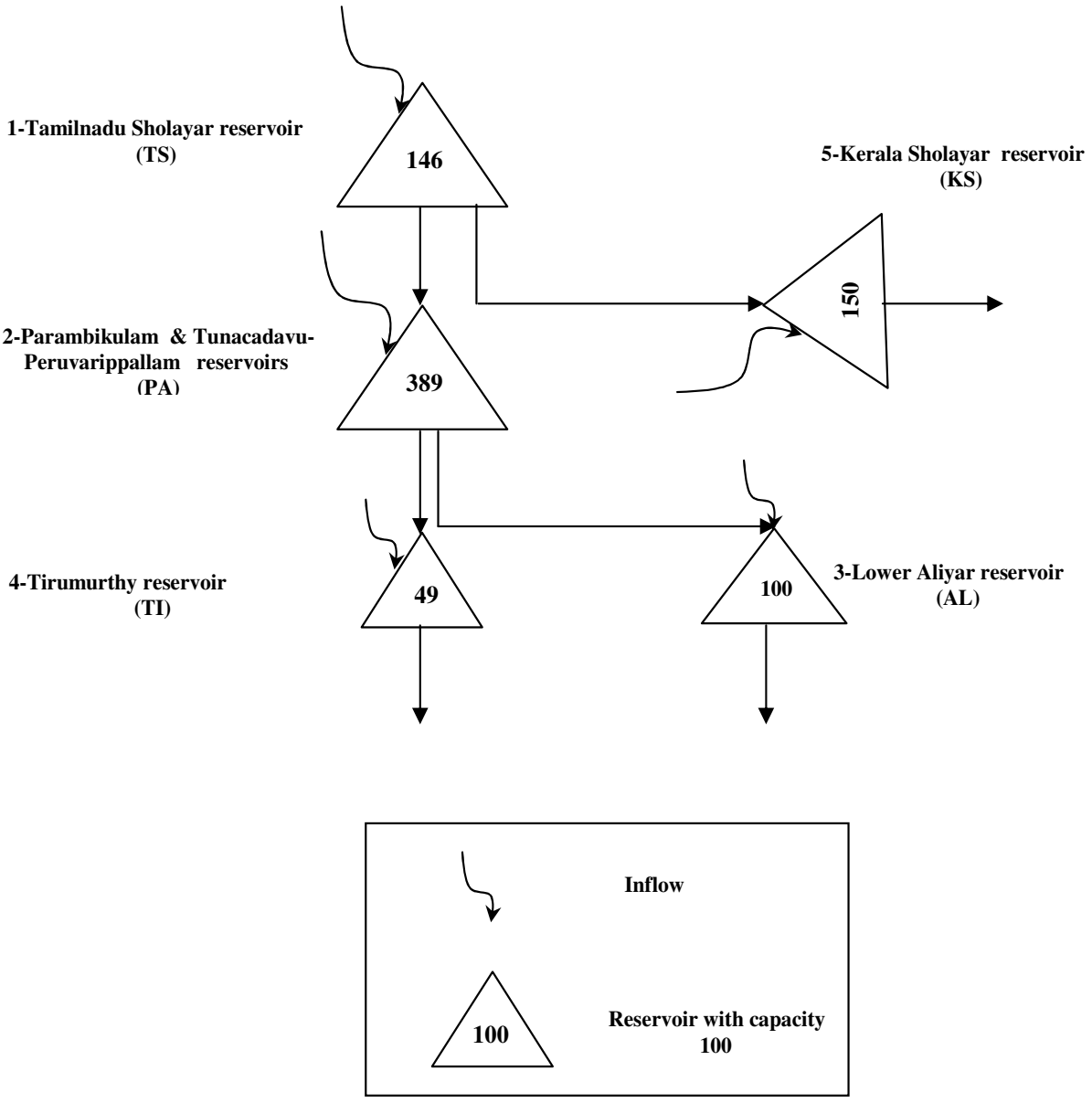

Figure 4.9: Parambikulam-Aligar Project (PAP)-reduced system

later.

Furthermore, based on the nature of PAP case study, there are high correlations be-

Table 4.13: The means of inflows

reservoir	**periods**											
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
TS (1)	117.3	143	62.3	31.2	20.8	11.7	4.5	4.8	5.2	4.8	7.5	61.6
PA (2)	85.5	99.8	56	44.8	34.2	17.5	10.3	7.6	6.3	5.5	6.8	38.6
AL (3)	39.2	49.8	28.2	27.3	25.8	20.6	9.4	7.3	6.2	3.3	3.9	16.4
TI (4)	1.3	2.1	1.8	6.3	9.3	8.9	3	2.1	2.1	1.8	1.8	1.3
KS (5)	42.1	47.4	22.1	16.2	12.7	17.4	14.8	5.3	5.3	4.3	5.6	19.7

Table 4.14: The standard deviations of inflows

reservoir	**periods**											
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
TS (1)	48.1	87.1	29.9	8.7	5.7	1.8	1.6	2.5	2.3	2	5	46.6
PA (2)	25.1	73.6	45.1	21.8	19	11.9	15.2	13.2	7.6	5.8	7.7	27.7
AL (3)	16.7	25.	8.8	9.4	15.1	17.5	5.2	3.6	4.1	3	3.9	9.5
TI (4)	1.7	1.9	2.1	7	11.9	8.8	2.3	2.4	2.5	2	1.5	1.3
KS (5)	10.4	20	13.7	5.5	3.6	10.1	10.2	3.4	3.7	1.7	3.3	15.1

tween the first reservoir (Tamilnadu Sholayar reservoir) and the second (Parambikalami and Tunacadavu-Peruvarippallam reservoirs), third (Lower Aliyar reservoir), and fifth reservoir (Kerala Sholayar reservoir) in terms of natural inflows.

Generating inflows

The Double-Bounded Probability Density Functions (DBPDF) are fitted based on the historical data. Inflows are generated using these fitted distributions. It should be noted

that these fitted distributions are not usually normal-like as illustrated in Figure 4.10 for some typical months. Moreover, the fitted Double-Bounded CDFs (DBCDFs) are suitably matched to those CDFs approximated from the historical data (DCDFs) (see Figure 4.11 for example).

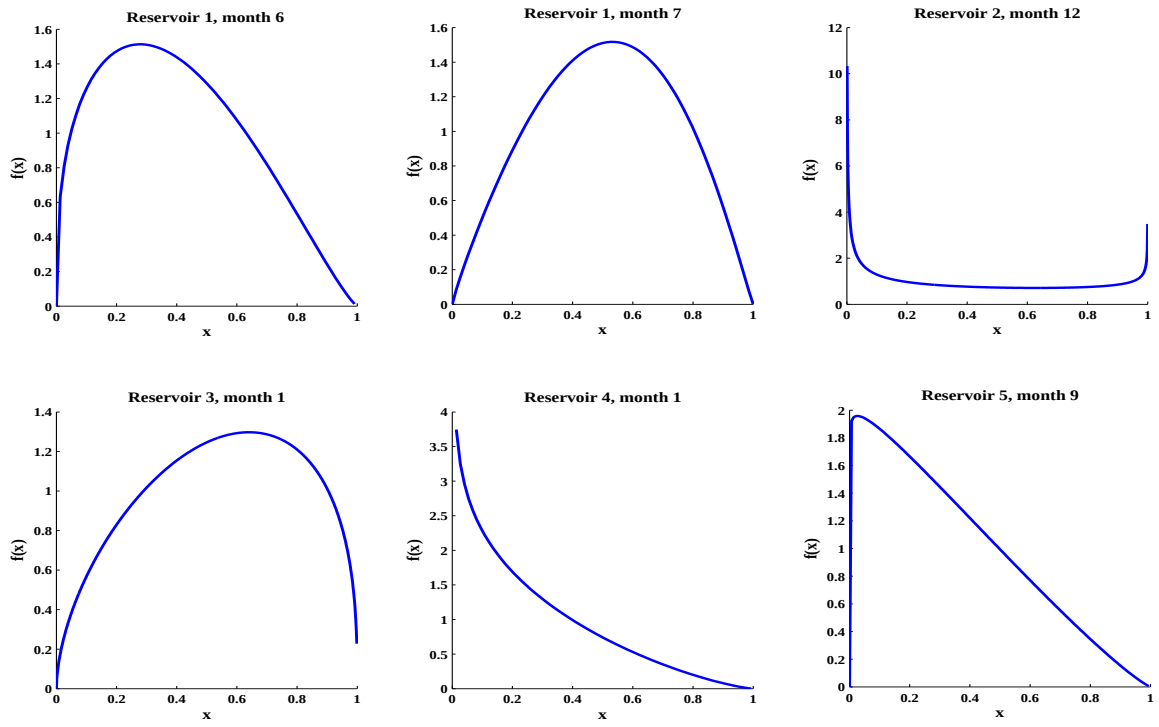

Figure 4.10: The probability density functions for reservoirs in some months

Because the CDFs are integrals, the differences are smoothed out, unlike in the PDFs. However, in FP models and simulations, CDFs are used.

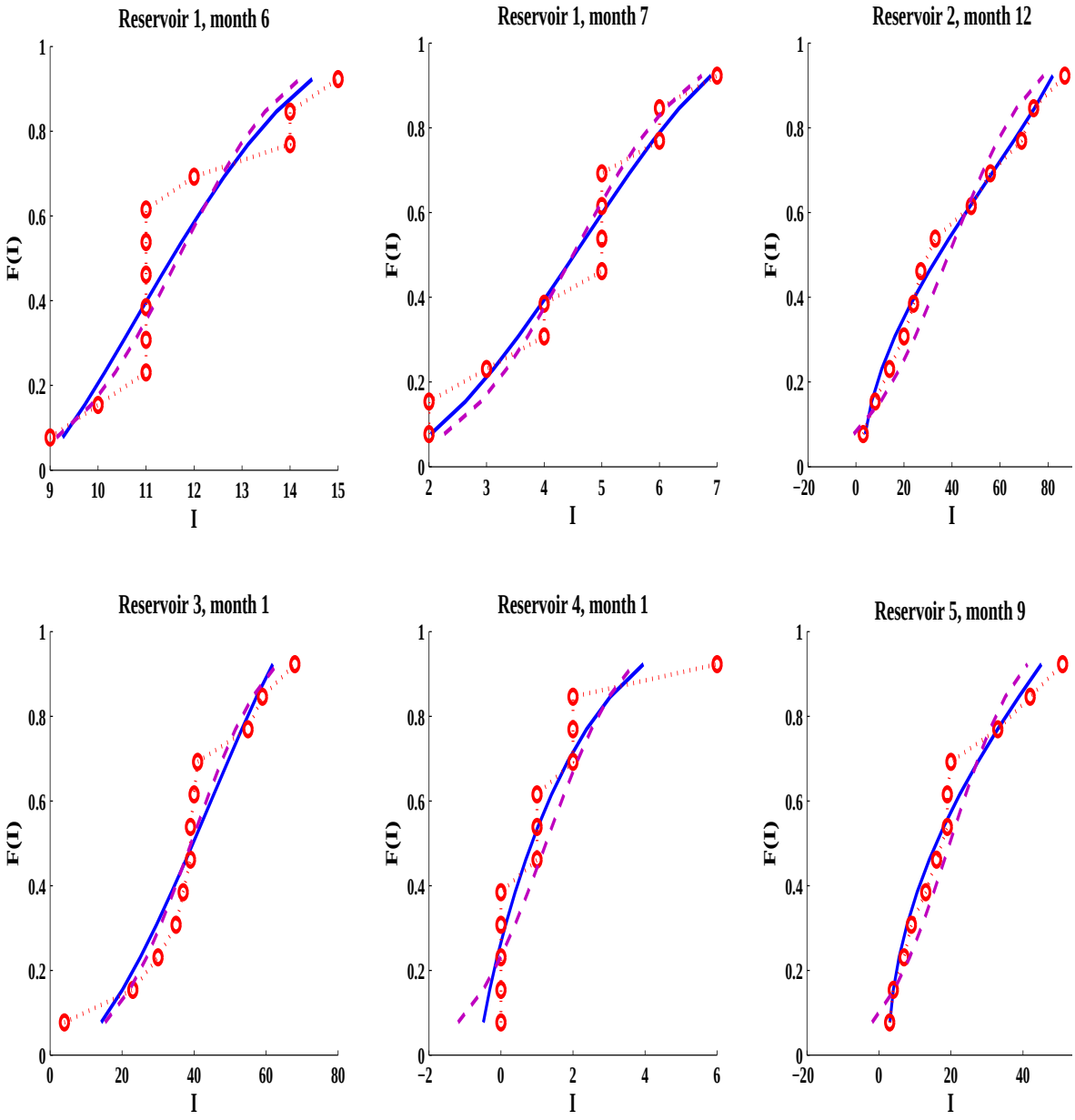

Figure 4.11: Cumulative Distribution Functions (CDF) obtained based on: 1-Normal distribution (NCDF), 2-Double-Bounded distribution (DBCDF), and 3-Empirical (ECDF)

Objective function and respective coefficients

The objective function is defined as

$$Z(u^t) = \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T} c_i^t \times u_{ij}^t, \tag{4.8}$$

where c_i^t is the benefit per unit which is given in Table 4.15.

Table 4.15: The benefit per unit release

reservoir	**periods**											
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
TS (1)	0.6	1	1	0.05	0.2	0.2	0	0	0	0	0	0
PA (2)	0.8	0.9	1	0.4	0.4	0.7	0.8	0.8	0.8	0	0	0
AL (3)	0.25	0.35	0.35	0.25	0.35	0.3	0.3	0.3	0	0	0.2	0.25
TI (4)	0.55	0.9	1	0.22	0.28	0.42	0.58	0.62	0.44	0	0	0
KS (5)	0.1	0.25	0.3	0.22	0.25	0.4	0.5	0.4	0.4	0.3	0.2	0.2

Moreover, it is assumed that any spillage from reservoirs do not affect the value of the objective function. Furthermore, based on the nature of PAP case study, 20% loss should be considered for all releases flowing to the fourth reservoir (Tirumurthy) from the third reservoir (Paramabikulam reservoir) because of the long tunnel used for water flow between these reservoirs.

Boundary conditions for release

The lower release bounds are set to zero for all periods. Table 4.16 also illustrates the upper bounds for all different connections in PAP case study (i.e., the release from reservoir i to reservoir j). It is assumed that these bounds are the same for twelve months. The diagonal elements in this table indicate the total maximum releases for each of the five reservoirs.

Table 4.16: The upper bounds of release

reservoir	TS	PA	AL	TI	KS
TS (1)	173.62	123.9	-	-	49.72
PA (2)	-	115.4	57.70	57.7	-
AL (3)	-	-	49.23	-	-
TI (4)	-	-	-	105.12	-
KS (5)	-	-	-	-	66.67

Initialization

As described in Chapter 3, there exist three types of variables in the optimization problem of the FP method:

1. expected value of storage volume, $E(s_i^t)$;
2. the variance of storage volume, $\sigma^2(s_i^t)$;
3. the constant part of the release from i to j, k_{ij}^t.

A simulation of policy from [78] is done to set the initialization for variables in items 1 and 2. The initial values for the third variable, k_{ij}^t can be set using the following formula:

$$k_{ij}^t = u_{ij}^t - E(s_i^t),$$

where u_{ij}^t is the release policy used in the simulation program for initialization.

4.3.2 Evaluation criteria

Having considered that the objective function in our problem is to maximize the expected value of total annual benefit, the following performance criteria can be extracted from simulation to compare the various models:

- the average of annual benefit (μ_B),
- the standard deviation of annual benefit (σ_B),
- the ratio of annual standard deviation to annual average of benefit ($cov = \frac{\sigma_B}{\mu_B}$),
- the distance between statistical features of storages estimated from the simulation to those corresponding features extracted from the optimization process for every individual reservoir i as:

$$\begin{aligned} &\text{the absolute value distance for expected values:} \quad DOE_i = \sum_{t=1}^{T} |\hat{E}(s_i^t) - E(s_i^t)|, \\ &\text{the absolute value distance for standard deviations:} \quad DOS_i = \sum_{t=1}^{T} |\hat{\sigma}(s_i^t) - \sigma(s_i^t)|, \end{aligned} \tag{4.9}$$

where $\hat{E}(s_i^t)$ and $\hat{\sigma}(s_i^t)$ are the expected value and the standard deviation of storage obtained from the simulation for reservoir i, respectively, and $E(s_i^t)$ and $\sigma(s_i^t)$ are the corresponding statistical features obtained from the optimization. This measures how well the estimation calculated in the optimization compares with the calculated values in the simulation.

Because the Cumulative Distribution Functions (CDFs) are used to calculate first and second moments of storages, the two main constraints in FP optimization models, the models are highly nonlinear and non-convex. Therefore, these models might be sensitive to initial solutions. Given this fact, the optimization and simulation process can be repeated multiple times (e.g., 30 times) with different initial solutions (e.g., randomly generated) and the maximum and the range of benefits ($\max(\mu_B)$ and $\max(\mu_B) - \min(\mu_B)$) are used for comparison.

Furthermore, another optimization method is formulated where all integrals in moment equations in Chapter 3 can be calculated through Monte Carlo simulation in every iteration of the optimization, which is called the Monte Carlo-based optimization approach. The average and the standard deviation of benefit obtained through this method can be used as criteria for comparison.

Complexity

Finding the complexity of nonlinear non-convex optimization problems is very difficult, and is rarely investigated in the literature [79]. However, finding the number of constraints and variables as a function of the number of reservoirs might give us a better idea about the computational time or the complexity of the problem.
In terms of variables, adding one reservoir to an existing set of system reservoirs creates $3 \times T$ new variables for the optimization problem. The number of equality constraints are a function of N (N is the number of reservoirs). Therefore, the order of complexity in terms of the size of the problem is $O(NT)$.

4.3.3 Two-reservoir case study

In order to demonstrate the results of all the mentioned performance criteria in a shorter CPU time, the first two reservoirs of the PAP case study (Reservoirs 1 and 2) which are serially connected to each other have been chosen. The following parameters are tuned for the experiments:

1. The optimum release policy obtained from the optimization process is simulated for 4000 years to extract the performance criteria. The results were found to be insensitive after this length of time for simulation. The number of years used in simulation in every iteration of the Monte Carlo-based optimization is 150 years.

2. All optimization processes are performed using "*fmincon*" in the Matlab(R) toolbox.

Under the Gaussian assumption, two different approaches for computing the expected value of release have been introduced. To simplify the way of demonstrating results, these FP-based model approaches are abbreviated as follows:

- **GS1:** Gaussian distribution function, Approach 1 (using upstream mean releases with lower accuracy)

- **GS2:** Gaussian distribution function, Approach 2 (using upstream mean releases with higher accuracy).

Relaxing the Gaussian assumption and using the Double-Bounded (DB) probability distribution function, two new FP-based models are created which are titled as follows:

- **DB1:** DB distribution function, Approach 1 (using upstream mean releases with lower accuracy),
- **DB2:** DB distribution function, Approach 2 (using upstream mean releases with higher accuracy).

Each of the above two FP-based models has the Monte Carlo-based optimization version which can be analogously named as (stochastic inflows are generated using the DBPDF fit)

- **MC1:** Monte Carlo-based optimization, Approach 1 (using upstream mean releases with lower accuracy),
- **MC2:** Monte Carlo-based optimization, Approach 2 (using upstream mean releases with higher accuracy).

The means, standard deviations, and coefficients of variation, obtained from the simulation of the system for the objective function in Equation 4.8 corresponding to different models, are summarized in Table 4.17. Regarding the results in this table, the following facts can be highlighted

- All models constructed using DBPDF seem to have comparable performance with those models which use Gaussian distributions or simulation
- The results for models based on Approach 2 for calculating the expected value of releases are superior to those based on Approach 1;

Table 4.17: The performance criteria for FP-based models

criteria	**FP model based on**					
	Gaussian		**Non-Gaussian**		**Monte Carlo**	
	GS1	**GS2**	**DB1**	**DB2**	**MC1**	**MC2**
μ_B	855.4	922.39	890.16	**951.1**	892.6	935.3
σ_B	163.4	156.1	175.4	**138.1**	131.7	130.7
$(C.O.V)_B$	0.19	0.17	0.2	**0.15**	0.15	0.14

μ_B: Aver. benefit, σ_B: Std. of benefit, and
$(C.O.V)_B$: Coefficient of variation of benefit $(\frac{\sigma_B}{\mu_B})$

- As it has been previously mentioned, the performance criteria related to Monte Carlo-based optimization models can be a reliable basis to check the validity of other models. As can be seen, the results for all performance criteria in both Gaussian- and non-Gaussian-based problems are quite comparable to those which are extracted from the corresponding Monte Carlo-based optimization models. However, these Monte Carlo-based optimization take about five times more CPU time than others.

In terms of the closeness of the optimization model results to the simulation results (the expected values and the standard deviations of storages, SOS and DOE, for all periods), Table 4.18 demonstrates the respective performance criteria (Equations 4.9) for all mentioned models in both reservoirs of the case study. Note here that simulation is performed using optimal release decisions derived from the corresponding optimization model. Overall, the DB2 seems to be a better method in terms of the proximity of the solutions for storage moments calculated in FP models and the corresponding simulation. So far, all investigations are made based on a single run, in which the respective optimization processes for all different FP models are started with the initial solution obtained from [78]. To assess the sensitivity of different FP-based models to initial solutions, all experiments have been performed thirty times, in each of which a random initial release

Table 4.18: The total distance between optimization and simulation results

Res.	Criteria	FP model based on					
		Gaussian		Non-Gaussian		Monte Carlo	
		GS1	GS2	DB1	DB2	MC1	MC2
TS (1)	DOE_1	44.2	41.9	21.7	53.2	54	66.3
	DOS_1	380.5	373.6	20.1	21.8	26.3	38.1
PA (2)	DOE_2	201.7	51.6	150.1	61.9	45.4	50.2
	DOS_2	129.1	111.7	208.5	101.2	67.4	70.8

policy is generated. The maximum expected values, in addition to the respective standard deviation obtained through the simulation, can be used for comparison of different modeling techniques. Moreover, the range of variability of expected values of benefit can also be utilized to measure the sensitivity of FP models to initial solutions (Table 4.19).

Table 4.19: The expected value and the standard deviation of annual benefit with different initializations

criteria	FP model based on					
	Gaussian		Non-Gaussian		Monte Carlo	
	GS1	GS2	DB1	DB2	MC1	MC2
$[\max(\mu_B),\ (C.O.V)^*_B]$	[901.55, 0.19]	[965.08, 0.16]	[899.4, 0.2]	[964.26, 0.17]	[957.97, 0.17]	[952.82, 0.17]
range*	267.87	185.88	256.92	342.83	326.89	145.83

$*$range=$\max(\mu_B) - \min(\mu_B)$, $*(C.O.V)_B$=Coefficient of variation $(\frac{\sigma_B}{\mu_B})$

As expected, using Approach 2, where the expected values of upstream reservoir releases are calculated more accurately, leads to better performance. Moreover, the averages of the Gaussian- and non-Gaussian-based FP models using the same release approach are close to each other.

4.3.4 Five-reservoir case study-PAP

Because the Monte Carlo-based optimization version of FP models are computationally expensive, we only considered the FP models based on two different approaches for Gaussian and non-Gaussian distribution functions in the five-reservoir case study. The performances related to the benefit obtained in simulation are summarized in Table 4.20 in which every experiment is run thirty times.

Table 4.20: The average expected value and the standard deviation of annual benefit-PAP

criteria	**FP model based on**			
	Gaussian		**Non-Gaussian**	
	GS1	**GS2**	**DB1**	**DB2**
$[\max(\mu_B),\ (C.O.V)^*_B]$	[1215.7, 0.1499]	[1436.4, 0.1449]	[1227.3, 0.1925]	[1324.5, 0.1448]
range*	166.6	855.01	363.73	354.49

*range=$\max(\mu_B) - \min(\mu_B)$, *$(C.O.V)_B$=Coefficient of variation ($\frac{\sigma_B}{\mu_B}$)

Analogous to the performances in the two-reservoir case study, the performances presented in Table 4.20 also verify that the optimization processes related to both Gaussian- and non-Gaussian-based FP models lead to close performance at the end of simulation when the average annual benefit is a little bit better for Gaussian-based FP model using Approach 2.

As previously noted, there is no assumption of the distribution of random storages for calculating the moments in FP models. Figure 4.12 demonstrates histograms of storages for some months in the five reservoirs. These values in these histograms are from simulation using a policy obtained from the optimization process. The figure demonstrates the complexity of distribution of storages which are unknown at the time of optimization.

In spite of the non-Gaussian inflows (Figure 4.10), the Gaussian-based model does as well as the DB-PDF-based model. This is due to the need for the use of CDFs which smooth the differences between various distributions.

It might be appropriate to compare the expected value and the standard deviation of storages obtained through the optimization and the simulation in different FP mod-

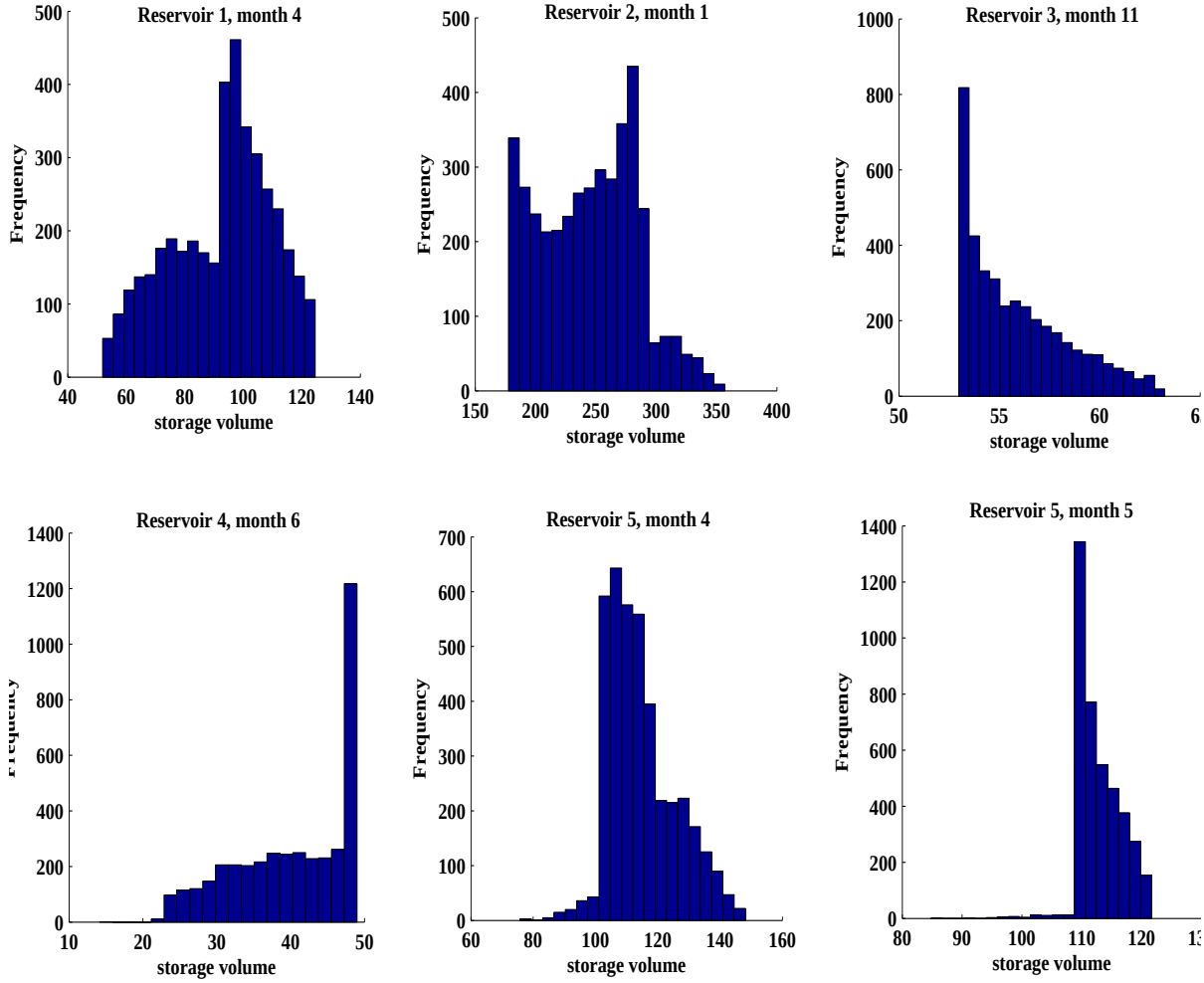

Figure 4.12: The histograms of storage volume for some months in the PAP system

els using Equation 4.9. Figures 4.13 and 4.15 illustrate the average expected values of storage from the optimal solutions in the optimization and the corresponding estimated values from the simulation for the non-Gaussian-based FP models. As can be observed, these averages match well for the model using Approach 2. However, there are some differences in the standard deviations of storages as presented in Figures 4.14-4.16. This might be due to the existence of correlations between storages which are not taken into consideration in the current models, which should be pursued in future.

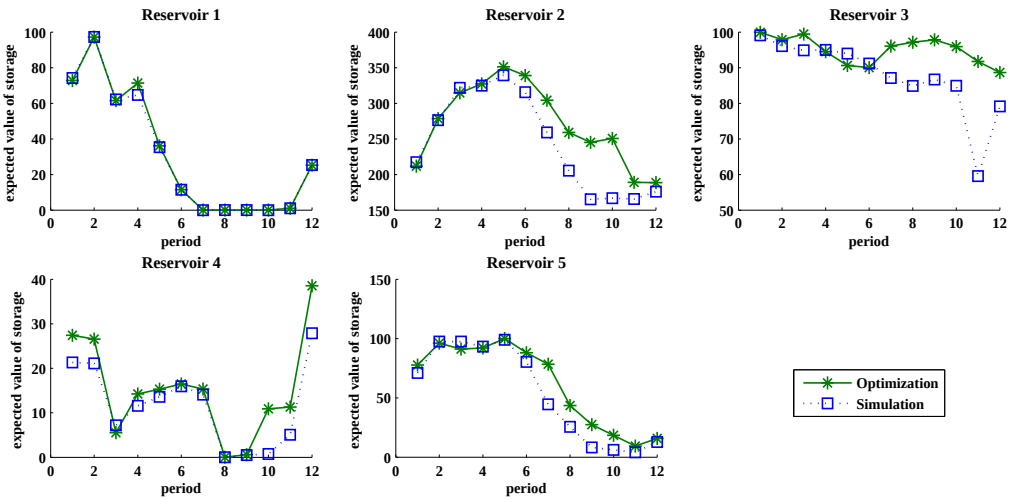

Figure 4.13: Comparing the expected value of storage obtained through the optimization and the simulation for PAP case study (non-Gaussian-based FP model, Approach 1)

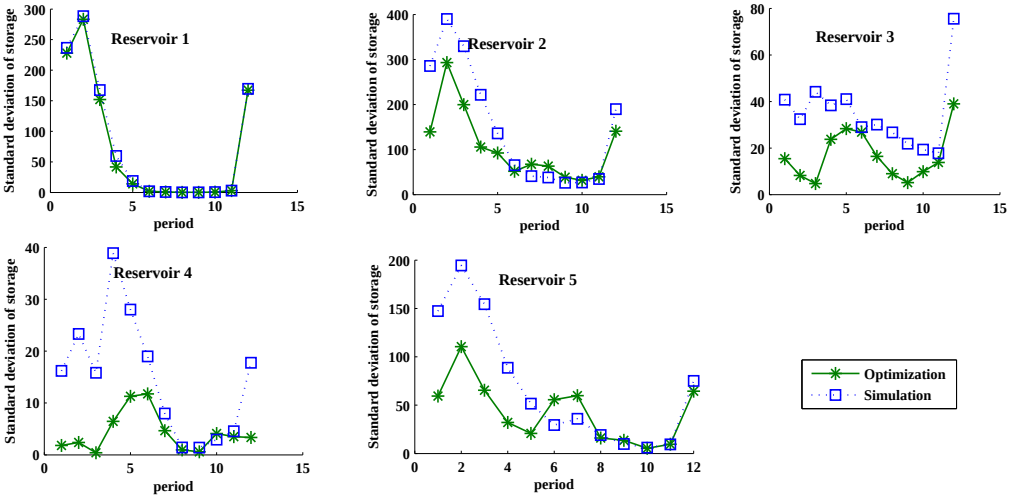

Figure 4.14: Comparing the standard deviation of storage obtained through the optimization and the simulation for PAP case study (non-Gaussian-based FP model, Approach 1)

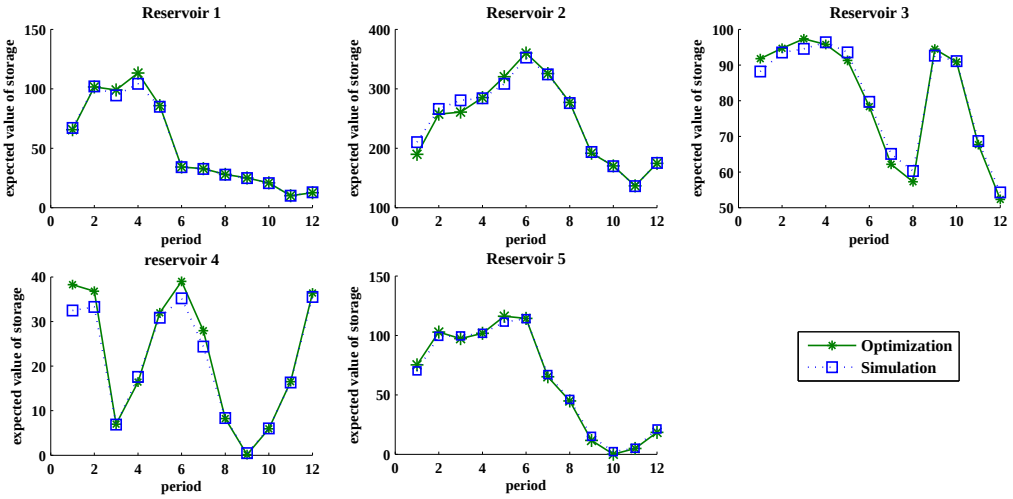

Figure 4.15: Comparing the expected value of storage obtained through the optimization and the simulation for PAP case study (non-Gaussian-based FP model, Approach 2)

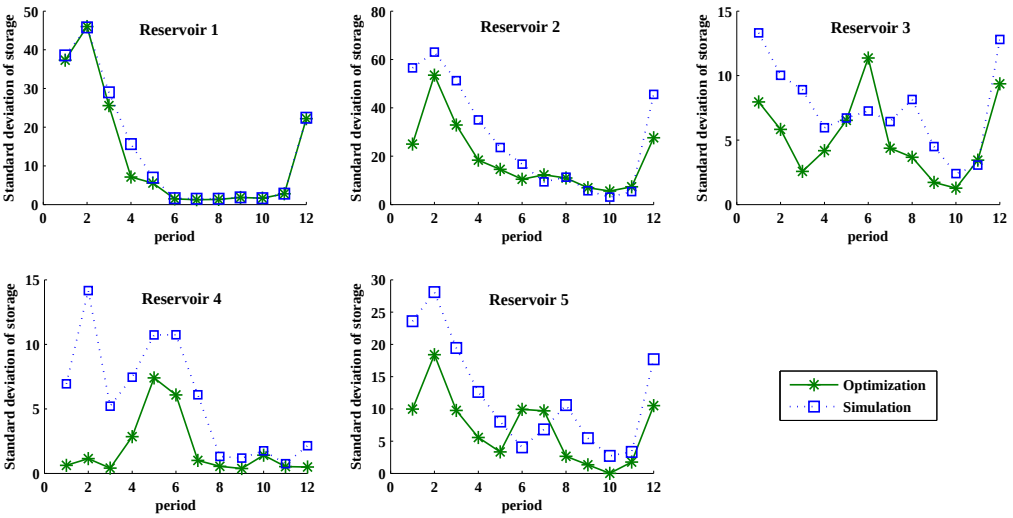

Figure 4.16: Comparing the standard deviation of storage obtained through the optimization and the simulation for PAP case study (non-Gaussian-based FP model, Approach 2)

4.4 Summary

In this chapter, we investigated the performance of developed RL methods in two different case studies: a 10×10 grid-world application with delayed reward, and a single reservoir application with immediate reward, where inflows and prices are uncertain. A new version of type II opposition-based learning was developed in which a multi layer perceptron (MLP) is employed for function approximation. We have shown that both types of OBL schemes in both case studies (grid-world and reservoir applications) have good results. In applying FP models to a multi-reservoir system, two different approaches for finding the expected value of release based on Gaussian and non-Gaussian assumptions were tested on two case studies. The following points can be summarized:

- Using Approach 2 leads to a better average annual benefit and less standard deviation of annual benefit in some cases; however, it is computationally more expensive compared to Approach 1 (e.g., in the five-reservoir case study with Gaussian-based model, Approach 2 takes about three times more CPU time than Approach 1).

- There was no assumption for the distribution of storages, however, the mean of the storages as part of optimal solutions in the optimization process matches the corresponding values estimated through the simulation.

- By looking at the performance criteria in the two-reservoir case study, the non-Gaussian-based models may provide better solutions; however, it has analogous performance to Gaussian-based models in the five-reservoir case study.

- In this case study, inflows are not normally-distributed, however, the optimal policy and the performance criteria corresponding to Gaussian-based models were not far from those obtained through the Monte Carlo-based and non-Gaussian-based FP models.

Chapter 5

Summary and Conclusions

The main objective in this thesis was to develop optimization methods to determine optimal policies in multi-storage applications operating in a stochastic environment. Discrete dynamic programming based techniques can solve multi-storage problems using aggregation/decomposition techniques for reducing dimensionality. In the first part, we focused on Reinforcement Learning (RL) techniques, which are based on discrete dynamic programming, as adaptive and model-free methodologies to tackle these problems. Four popular RL techniques including Q-Learning and Q(λ) as off-policy, and sarsa and sarsa(λ) as on-policy techniques have been used. There are several contributions in this part which can be summarized as follows:

- Demonstration of how to apply these four RL techniques to storage management applications is presented. Moreover, to find the set of admissible actions, two different schemes were suggested: optimistic and pessimistic. The updating process of action-value functions were also accomplished using a weighting scheme. Type I and type II opposition schemes for these four learning techniques to speed up the learning process are employed. A new type II opposition approach using function approximation has also been proposed to assist the decision-maker in finding the opposites.

- These learning techniques usually consider the expected value of accumulated re-

ward (value function) to make a decision or to find a policy. This criterion alone cannot be reliable for decision-making in many cases, and risk should be considered. A new methodology to consider risk in learning methods was developed.

In addition, in this thesis, a new nonlinear optimization modeling technique developed by Fletcher and Ponnambalam for single reservoir was extended to solve multi-storage optimization problems. Also, the Gaussian-distribution limitation on inflows was removed to a more general distribution. Finally, as an illustration of extension of this work to other storage management cases, a warehouse management problem was modeled.

In order to demonstrate the efficiency of the respective optimization models, the learning techniques, and their proposed extensions, the following applications have been considered for the experimental results:

- a single reservoir case study with uncertainties in prices and inflows,
- a two-reservoir case study,
- a five-reservoir case study located in India.

It should be noted that all performance criteria (e.g., the average, the standard deviation, the coefficient of variation of annual benefit, ...) have been obtained through a simulation routine using the corresponding policies derived from learning methods (e.g., Q-Learning, sarsa, ...) and TSP and FP optimization methods.
To assess the learning methods in the single reservoir, Stochastic Dynamic Programming (SDP) techniques for both applications have been used in which the global optimal decision policies can be provided. The following achievements can be pointed out:

1. Using opposition schemes in the investigated learning methods, specially in the early stage of learning process, causes a fast convergence to the steady state situation. Moreover, the respective performance criteria using the policies derived from the opposition-based learning process with a small number of episodes are

reasonably close to SDP results; in contrast in regular learning methods, the results close to SDP are obtained after almost four times the number of episodes in opposition-based learning methods.

2. Because the learning methods are based on the simulations of the random variables, the optimal policy at the end of the learning process (where the action-value functions converge to the steady state situation), and the corresponding performance criteria could vary. Using opposition schemes in the learning process decreases the variability of policies and the corresponding performance criteria when the respective learning process is repeated multiple times (i.e., using opposition schemes in the learning methods makes them more robust).

3. The trade-off of standard deviation versus expected benefit in all the three methods, namely, Q-learning, TSP, and FP traverses a similar curve. This curve assists a decision-maker to choose an optimal policy considering risk, which is represented by standard deviation.

In the two-reservoir case study, a Monte-Carlo-based optimization technique has been employed. In this technique, all respective integrals in the first and the second moments of storages in FP models are approximated using a Monte-Carlo simulation. In the five-reservoir case study, running the Monte-Carlo-based optimization technique is too time-consuming; therefore, only the Gaussian- and the non-Gaussian-based FP models are considered for performing the optimization process. Moreover, because of using Cumulative Distribution Functions (CDF) in approximating the moment equations, the FP models are non-convex which implies that the optimization process could be sensitive to initial solutions. Therefore, the optimizations have been performed with different random initial solutions multiple times. The best solution among all different local optima at the end of optimization process is opted for comparison (e.g., the best performance criteria of Gaussian-based are compared to the best ones pertinent to non-Gaussian-based FP models). The following achievements can be summarized:

1. The performance criteria for two- and five-reservoir case studies confirm that the second approach of calculating the expected value of releases (the upstream mean releases with higher accuracy) always leads to superior results for Gaussian and non-Gaussian-based FP models in terms of average and standard deviation of annual benefits.

2. The optimal expected value and the variance of storages as decision variables in FP models are reasonably matched to the corresponding values obtained through simulation. However, the expected values of storages are better matched compared to the variances.

All the mentioned methods have advantages and disadvantages which can be summarized as follows:

1. TSP is easier and more general to be applied in the stochastic storage management problems; however, it could exponentially increase the size of the optimization problems that need to be solved.

2. All learning methods can be performed in an off-line (simulation) and an on-line (real-time decision-making) situation. They are model-free which means that the transition probabilities are not needed. The learning processes are incremental which indicates that they can be continued forever. However, the implementation of these techniques for multi-storage applications requires approximation techniques to reduce the model dimensionality.

3. There is no discretization in FP models which make this technique more attractive to be applied in multi-storage applications. Moreover, the probabilities of spill and empty are available. However, the optimization model is non-convex.

5.1 Future work

1. Applying RL techniques for multi-reservoir or multi-storage applications could be time-consuming, or computationally impossible in some cases. The Aggregation-

Decomposition (A/D) methods presented by Turgeon [14] and Ponnambalam and Adams [13] can be used in RL techniques.

2. In FP models, the first and the second moment of storage have been used as constraints. However, correlations between storages were not considered and which could increase the accuracy of results.

Appendix A

An initial policy for PAP

Table A.1: The mean of releases resulting from a steady-state simulation performed using the rule curve sets for PAP

from/to	**periods**											
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1 to 2	44.6	74.8	71.5	5	18.6	14.4	0	0	0	0	0	6.9
1 to 5	54.4	30	10	10	10	9.9	14.4	4.3	7.8	9.5	11.3	18.8
2 to 3	9.4	10	0	9.2	9.2	0	0	0	0	0	0	0
2 to 4	70	80	96.2	29.9	29.8	61.3	68.6	63.5	52.8	0	0	0
***3 to 3**	23.4	33.3	33.2	25	34	28.5	26.3	22.8	0	0	15.7	21.3
***4 to 4**	52	83.5	87.6	25	29.6	37.7	43.9	45.3	35.4	0	0	0
***5 to 5**	11	26	29	22	24	39	50	41	42	28.6	18.5	20

* i to i means the total release from reservoir i

Appendix B

The cost function with spills loss

The expected benefit of the release from reservoir i to reservoir j in period t, $E(c_{ij}^t)$, can be computed as:

$$E(c_{ij}^t) \quad = \quad f_1^t(A_1) + f_1^t(A_2) + f_2^t(A_3), \tag{B.1}$$

where A_1, A_2 are the actual releases and A_3 is the spillage (the controlled release is subtracted from the total release) and $f(.)$ is the corresponding loss or revenue function. These parameters can be therefore computed as:
Where s_i^t is within bounds,

$$A_1 \quad = \quad [k_{ij}^t + E(s_i^{t-1})] \cdot \left\{ \int_{s_{i,min}^t - \bar{I}_i^t + k_{ii}^t - \sum_{l=1 \ l \neq i}^{N} E(u_{li}^t)}^{\infty} f_{\eta_i^t}(\eta_i^t)\, d\eta_i^t \right\} , \tag{B.2}$$

Where s_i^t is less than minimum storage,

$$\begin{aligned} A_2 &= \psi_{ij}^t \times \left\{ \left[-s_{i,min}^t + E(s_i^{t-1}) + \bar{I}_i^t + \sum_{l=1, l\neq i}^{N} E(u_{li}^t) \right] \right. \\ &\times \left[\int_{-\infty}^{s_{i,min}^t - \bar{I}_i^t + k_{ii}^t - \sum_{l=1\ l\neq i}^{N} E(u_{li}^t)} f_{\eta_i^t}(\eta_i^t)\, d\eta_i^t \right] \\ &+ \left. \int_{-\infty}^{s_{i,min}^t - \bar{I}_i^t + k_{ii}^t - \sum_{l=1\ l\neq i}^{N} E(u_{li}^t)} \eta_i^t f_{\eta_i^t}(\eta_i^t)\, d\eta_i^t \right\}, \end{aligned} \tag{B.3}$$

Where s_i^t is higher than maximum storage,

$$\begin{aligned} A_3 &= \psi_{ij}^t \times \left\{ \left[\overbrace{-s_{i,max}^t + E(s_i^{t-1}) + \bar{I}_i^t + \sum_{l=1, l\neq i}^{N} E(u_{li}^t)}^{\text{total release with spillage}} - \overbrace{\left(k_{ii}^t + E(s_i^{t-1})\right)}^{\text{controlled release}} \right] \right. \\ &\times \left[\int_{s_{i,max}^t - \bar{I}_i^t + k_{ii}^t - \sum_{l=1\ l\neq i}^{N} E(u_{li}^t)}^{\infty} f_{\eta_i^t}(\eta_i^t)\, d\eta_i^t \right] \\ &+ \left. \int_{s_{i,max}^t - \bar{I}_i^t + k_{ii}^t - \sum_{l=1\ l\neq i}^{N} E(u_{li}^t)}^{\infty} \eta_i^t f_{\eta_i^t}(\eta_i^t)\, d\eta_i^t \right\}. \end{aligned} \tag{B.4}$$

As can be seen, because the controlled release is the same for both situations of which the end-of-period storage is within the containment or higher than maximum level, the upper bound of the integral in term A_1 is set to infinity. On the other hand, the portion of water which is overflowed is calculated in term A_3 by subtracting the release based on the policy from the total release with spillage included. This is also quite common in real world case studies of storage management, that terms A_1 and A_2 are treated with the same cost function while term A_3, which has usually a negative impact on the objective function, is treated with a different cost function.

Appendix C

Moment equations for two serial reservoirs

The first moment of storage for the second reservoir in the case of two serial reservoirs (Part b of Figure 3.2) using the new estimations obtained for the expected value of releases in Equations 3.44 and 3.46 can be written as:

$$
\begin{aligned}
&E(s_2^t) = \\
&[\bar{I}_2^t - k_{11}^t + E(u_{12}^t)] \cdot \left\{ \tfrac{1}{2} \left[erf\left(\tfrac{[s_{2,max}^t - (\bar{I}_2^t - k_{11}^t + E(u_{12}^t))]}{[2Var(\eta_2^t)]^{1/2}} \right) - erf\left(\tfrac{[s_{2,min}^t - (\bar{I}_2^t - k_{11}^t + E(u_{12}^t))]}{[2Var(\eta_2^t)]^{1/2}} \right) \right] \right\} \\
&-\left(\tfrac{[Var(\eta_2^t)]^{(1/2)}}{\sqrt{2\pi}}\right) \cdot \left\{ \exp\left(-\tfrac{1}{2} \tfrac{[s_{2,max}^t - (\bar{I}_2^t - k_{11}^t + E(u_{12}^t))]^2}{Var(\eta_2^t)} \right) - \exp\left(-\tfrac{1}{2} \tfrac{[s_{2,min}^t - (\bar{I}_2^t - k_{11}^t + E(u_{12}^t))]^2}{Var(\eta_2^t)} \right) \right\} \\
&+(s_{2,min}^t) \cdot \left\{ \tfrac{1}{2} \left[1 + erf\left(\tfrac{[s_{2,min}^t - (\bar{I}_2^t - k_{11}^t + E(u_{12}^t))]}{[2Var(\eta_2^t)]^{1/2}} \right) \right] \right\} \\
&+(s_{2,max}^t) \cdot \left\{ \tfrac{1}{2} \left[1 - erf\left(\tfrac{[s_{2,max}^t - (\bar{I}_2^t - k_{11}^t + E(u_{12}^t))]}{[2Var(\eta_2^t)]^{1/2}} \right) \right] \right\}.
\end{aligned}
\tag{C.1}
$$

Suppose that the spillage from the upstream reservoir would also be available for the downstream one without any losses; therefore, the second choice in equation 3.46 can be

used for finding $E(u_{12}^t)$ as:

$$
\begin{aligned}
&E(u_{12}^t) = \\
&\left[k_{12}^t + E(s_1^{t-1})\right] \cdot \left\{\tfrac{1}{2}\left[erf\left(\tfrac{s_{1,max}^t - \bar{I}_1^t + k_{11}^t}{[2Var(\eta_1^t)]^{1/2}}\right) - erf\left(\tfrac{s_{1,min}^t - \bar{I}_1^t + k_{11}^t}{[2Var(\eta_1^t)]^{1/2}}\right)\right]\right\} \\
&+ \left[-s_{1,min}^t + E(s_1^{t-1}) + \bar{I}_1^t)\right] \times \left\{\tfrac{1}{2}\left[1 + erf\left(\tfrac{s_{1,min}^t - \bar{I}_1^t + k_{11}^t}{[2Var(\eta_1^t)]^{1/2}}\right)\right]\right\} \\
&-(\tfrac{[Var(\eta_1^t)]^{(1/2)}}{\sqrt{2\pi}}) \cdot \left\{\exp\left(-\tfrac{1}{2}(\tfrac{[s_{1,min}^t - \bar{I}_1^t + k_{11}^t]^2}{Var(\eta_1^t)}\right)\right\} + \left\{-s_{1,max}^t + E(s_1^{t-1}) + \bar{I}_1^t\right\} \\
&\times \left\{\tfrac{1}{2}\left[1 - erf\left(\tfrac{s_{1,max}^t - \bar{I}_1^t + k_{11}^t}{[2Var(\eta_1^t)]^{1/2}}\right)\right]\right\} + (\tfrac{[Var(\eta_1^t)]^{(1/2)}}{\sqrt{2\pi}}) \cdot \left\{\exp\left(-\tfrac{1}{2}\tfrac{[s_{1,max}^t - \bar{I}_1^t + k_{11}^t]^2}{Var(\eta_1^t)}\right)\right\}.
\end{aligned}
\tag{C.2}
$$

As previously mentioned, it can be quite reasonable to consider spillage as available water in a downstream reservoir. However, it is inappropriate to take the spillage into account as a potential variable creating benefit. The spillage usually causes a negative impact because it might lead to flooding and other damage (e.g., irrigation-related downstream), or in an optimistic view, it does not affect the objective function. For the latest case, the expected value of release which is going to be used in the objective function should be calculated based on the first choice in Equation 3.44. This expected value of release for the first reservoir in the serial case study can be calculated as:

$$
\begin{aligned}
&E(u_{12}^t) = \\
&\left[k_{12}^t + E(s_1^{t-1})\right] \cdot \left\{\tfrac{1}{2}\left[1 - erf\left(\tfrac{s_{1,min}^t - \bar{I}_1^t + k_{11}^t}{[2Var(\eta_1^t)]^{1/2}}\right)\right]\right\} + \left[-s_{1,min}^t + E(s_1^{t-1}) + \bar{I}_1^t)\right] \\
&\times \left\{\tfrac{1}{2}\left[1 + erf\left(\tfrac{s_{1,min}^t - \bar{I}_1^t + k_{11}^t}{[2Var(\eta_1^t)]^{1/2}}\right)\right]\right\} - (\tfrac{[Var(\eta_1^t)]^{(1/2)}}{\sqrt{2\pi}}) \cdot \left\{\exp\left(-\tfrac{1}{2}(\tfrac{[s_{1,min}^t - \bar{I}_1^t + k_{11}^t]^2}{Var(\eta_1^t)}\right)\right\}.
\end{aligned}
\tag{C.3}
$$

Appendix D

FP method: A new stochastic release policy

In Equation 3.15, the release policy is assumed to depend only on the storage level of the reservoir considered. This is an inefficient policy, because the storage levels in other reservoirs might be sufficient in that period to play a supportive role for demand points downstream. To tackle this issue, a new policy can be introduced in which the stochastic release depends, in addition to its own storage, on the storage levels of all physically-connected upstream reservoirs. This policy can be written as

$$u_{ij}^t = k_{ij}^t + \sum_{l=1}^{N} s_i^t \times \delta_{li}, \tag{D.1}$$

where $\delta_{li} = 1$, if the l^{th} reservoir is connected to the i^{th} reservoir, otherwise, $\delta_{li} = 0$. The dynamics of reservoirs with respect to the new stochastic policy can be written as:

$$\begin{aligned} s_i^t &= \left\{ \bar{I}_i^t + \eta_i^t + s_i^{t-1} - \left(k_{ii}^t + \sum_{l=1}^{N} [s_i^{t-1} \times \delta_{li}] \right) + \sum_{l=1}^{N} u_{li}^t \right\} \cdot 1_{[s_{i,min}^t, s_{i,max}^t]}(\hat{s}_i^t) \\ &+ \{s_{i,min}^t\} \cdot 1_{(-\infty, s_{i,min}^t)}(\hat{s}_i^t) + \{s_{i,max}^t\} \cdot 1_{(s_{i,max}^t, \infty)}(\hat{s}_i^t). \end{aligned} \tag{D.2}$$

For instance, the dynamic for the downstream reservoir in a two-serial-reservoirs application (Part *b* of Figure 3.2) can be expressed as:

$$
\begin{aligned}
s_2^t &= \{\bar{I}_2^t + \eta_2^t + s_2^{t-1} + \overbrace{(k_{12}^t + s_1^{t-1})}^{u_{12}^t} - \overbrace{(k_{22}^t + s_1^{t-1} + s_2^{t-1})}^{u_{22}^t}\} 1_{[s_{2,min}^t, s_{2,max}^t]}(\hat{s}_2^t) \\
&+ \{s_{i,min}^t\} 1_{(-\infty, s_{2,min}^t)}(\hat{s}_2^t) + \{s_{2,max}^t\} 1_{(s_{2,max}^t, \infty)}(\hat{s}_2^t) \\
&= \{\bar{I}_2^t + \eta_2^t + k_{12}^t - k_{22}^t\} 1_{[s_{2,min}^t, s_{2,max}^t]}(\hat{s}_2^t) + \{s_{2,min}^t\} 1_{(-\infty, s_{2,min}^t)}(\hat{s}_2^t) \\
&+ \{s_{i,max}^t\} 1_{(s_{2,max}^t, \infty)}(\hat{s}_2^t).
\end{aligned} \tag{D.3}
$$

In terms of mathematical notation, the moments using the new policy are the same as Equations 3.30 and 3.31 in which $K_2^t = k_{12}^t - k_{22}^t$.

As previously mentioned, the expected value of releases in Equations 3.44 or 3.46 can be considered as an input to reservoirs downstream, subject to an existing physical connection between them. In this case, the first term in Equation D.3 depends on the previous end-of-period storage level. The dynamic in this situation can be written for the second reservoir of our example as:

$$
\begin{aligned}
s_{22}^t &= \{\bar{I}_2^t + \eta_2^t + E(u_{12}^t) - (k_{22}^t + s_1^{t-1})\} 1_{[s_{2,min}^t, s_{2,max}^t]}(\hat{s}_2^t) + \{s_{2,min}^t\} 1_{(-\infty, s_{2,min}^t)}(\hat{s}_2^t) \\
&+ \{s_{2,max}^t\} 1_{(s_{2,max}^t, \infty)}(\hat{s}_2^t).
\end{aligned} \tag{D.4}
$$

The following equation is used to find the second moment:

$$
\begin{aligned}
(s_2^t)^2 &= \{\bar{I}_2^t + \eta_2^t + E(u_{12}^t) - (k_{22}^t + s_1^{t-1})\}^2 1_{[s_{2,min}^t, s_{i,max}^t]}(\hat{s}_2^t) + \{s_{2,min}^t\}^2 1_{(-\infty, s_{2,min}^t)}(\hat{s}_2^t) \\
&+ \{s_{2,max}^t\}^2 1_{(s_{2,max}^t, \infty)}(\hat{s}_2^t).
\end{aligned} \tag{D.5}
$$

As can be seen in the above formulation, the dynamic only depends on s_{12}^t as a random variable. By taking the expectation from these equations and using the first order approximation of Taylor's series analogous to what has been explained in section 3.5, the first and the second moments can be calculated. However, it should be noted that in taking the expectation of the first term in Equation D.5, $E(s_1^t)^2$ should be substituted

with $[E(s_1^t)]^2$ as follows:

$$E\{\bar{I}_2^t+E(u_{12}^t)-(k_{22}^t+s_1^{t-1})\}^2 \implies \{\bar{I}_2^t+E(u_{12}^t)^2-(k_{22}^t+E(s_1^{t-1}))\}^2 \overbrace{-\left(E(s_1^t)\right)^2+E(s_1^t)^2}^{+Var(s_1^t)}.$$

Using this revision, the second moment can be written as:

$$\begin{aligned}
E(s_2^t)^2 &= \left\{[\bar{I}_2^t + E(u_{12}^t) - k_{22}^t - E(s_1^{t-1})]^2 + Var(s_{11}^t)\right\} \cdot E\left(1_{[s_{2,min}^t, s_{2,max}^t]}(\hat{s}_2^t)\right) \\
&+ 2[\bar{I}_2^t + E(u_{12}^t) - k_{22}^t - E(s_1^{t-1})] \cdot E\left([\eta_2^t] \cdot 1_{[s_{2,min}^t, s_{2,max}^t]}(\hat{s}_2^t)\right) \\
&+ E\left([\eta_2^t]^2 \cdot 1_{[s_{2,min}^t, s_{2,max}^t]}(\hat{s}_2^t)\right) + \{s_{2,min}^t\}^2 E\left(1_{(-\infty, s_{2,min}^t)}(\hat{s}_2^t)\right) \\
&+ \{s_{2,max}^t\}^2 E\left(1_{(s_{2,max}^t, \infty)}(\hat{s}_2^t)\right).
\end{aligned}$$

The second moment can be generalized for multi reservoir cases as:

$$\begin{aligned}
E(s_i^t)^2 &= \left\{\left[\bar{I}_i^t + \sum_{l=1}^{N} E(u_{li}^t) - k_{ii}^t - \sum_{l=1}^{N} \sum_{l\neq i} [E(s_l^{t-1}) \times \delta_{li}]\right]^2\right. \\
&\left. + \sum_{l=1}^{N} \sum_{l\neq i} [Var(s_l^{t-1}) \times \delta_{li}]\right\} \times E\left(1_{[s_{i,min}^t, s_{i,max}^t]}(\hat{s}_i^t)\right) \\
&+ 2\left[\bar{I}_i^t + \sum_{l=1}^{N} E(u_{li}^t) - k_{ii}^t - \sum_{l=1}^{N} \sum_{l\neq i} E(s_l^{t-1})\right] \cdot E\left([\eta_2^t] \cdot 1_{[s_{i,min}^t, s_{i,max}^t]}(\hat{s}_i^t)\right) \\
&+ E\left([\eta_2^t]^2 \cdot 1_{[s_{i,min}^t, s_{i,max}^t]}(\hat{s}_i^t)\right) + \{s_{i,min}^t\}^2 E\left(1_{(-\infty, s_{i,min}^t)}(\hat{s}_i^t)\right) \\
&+ \{s_{i,max}^t\}^2 E\left(1_{(s_{i,max}^t, \infty)}(\hat{s}_i^t)\right).
\end{aligned} \tag{D.6}$$

References

[1] K. Ponnambalam, "Optimization in water reservoir systems," in *Handbook of Applied Optimization*, P. M. Pardalos and M. G. C. Resende, Eds. New York: Oxford University Press, 2002, ch. 18.16. xii, 1, 12, 13

[2] J. W. Labadie, "Optimal operation of multireservoir systems: State-of-the-art review," *J. Water Resou. Plan. and Manage.*, vol. 130, no. 2, pp. 93–111, 2004. xii, 1, 3, 4, 13, 14

[3] M. Kljajic, D. Kofjac, A. Skraba, and V. Rejec, "Warehouse optimization in uncertain environment," in *Proc. of the 22nd International Conference of the System Dynamics Society*, M. Kennedy, G. W. Winch, R. S. Langer, J. I. Rowe, and J. M. Yanni, Eds. 2, 3, 36

[4] W.G. Yeh, "Reservoir management and operations models: A state-of-the-art review," *Water Resou. Res.*, vol. 21, no. 12, pp. 1797–1818, 1985. 3, 9, 13, 16

[5] S. Yakowitz, "Dynamic programming applications in water resources," *Water Resou. Res.*, vol. 18, no. 4, pp. 673–696, 1982. 3, 16

[6] C. Monthatipkul and P. Yenradee, "Inventory/distribution control system in a one-warehouse/multi-retailer supply chain," *International Journal of Production Economics*, vol. 114, no. 1, pp. 119–133, July 2008. [Online]. Available: http://ideas.repec.org/a/eee/proeco/v114y2008i1p119-133.html 3, 36

[7] M. Athans and P. Falb, *Optimal control*. U.S.A: McGraw-Hill, 1966. 4

[8] N. Sandell, P. Varaiya, M. Athans, and M. Safonov, "Survey of decentralized control methods for large scale systems," *IEEE Transaction on Automatic Control*, vol. 23, no. 2, pp. 108–128, 1978. 4

[9] D. Loucks, J. Stedinger, and A. D. Haith, *Water resource systems planning and analysis*. New Jersey, U.S.A: Prentice Hall, 1981. 4, 7, 9, 31, 95

[10] T. Kjeldsen and D. Rosbjerg, "Choice of reliability, resilience and vulnerability estimators for risk assessments of water resources systems," *Hydrological Sciences Journal*, vol. 49, no. 5, pp. 755–767, 2004. 5

[11] N. Umamahesh and P. Sreenivasulu, "Two-phase stochastic dynamic programming model for optimal operation of irrigation reservoir," *Water Resour. Manage.*, vol. 11, no. 5, pp. 395–406, 1997. 5

[12] L. Mays and Y. Tung, *Hydrosystems engineering and management*. U.S.A: McGraw-Hill, 1992. 6, 7

[13] K. Ponnambalam and B.J. Adams, "Stochastic optimization of multireservoir systems using a heuristic algorithm: A case study from India," *Water Resou. Res.*, vol. 32, no. 3, pp. 733–742, 1996. 6, 10, 17, 120

[14] A. Turgeon, "A decomposition method for the long-term scheduling of reservoirs in series," *Water Resou. Res.*, vol. 17, no. 6, pp. 1565–1570, 1981. 10, 17, 120

[15] T. W. Archibald, T. W. Archibald, K. I. M. Mckinnon, K. I. M. Mckinnon, L. C. Thomas, and L. C. Thomas, "An aggregation stochastic dynamic programming model of multi-reservoir systems," 1996. 10

[16] S. Fletcher and K. Ponnambalam, "A constrained state formulation for the stochastic control of multireservoir systems," *Water Resou. Res.*, vol. 34, no. 2, pp. 257–270, 1998. 10, 19, 37, 53, 60, 76, 87, 88

[17] T. Roefs and T. Bodin, "Multireservoir operation studies," *Water Resou. Res.*, vol. 6, no. 2, pp. 410–420, 1970. 13

[18] A. Gosavi, *Simulation-based optimization: Parametric optimization techniques and reinforcement learning.* Norwel, Massachusetts, U.S.A.: Springer, 2003. 13, 20, 22, 23

[19] D. Draper and K. Adamowski, "Application of linear programming optimization to a Northern Ontario hydro power system," *Canadian Journal of Civil Engineering*, vol. 3, no. 1, pp. 20–31, 1976. 14

[20] D. Bechard, I. Corbu, R. Gagnon, G. Nix, L. Parker, K. Stewart, and M. Trinh, "The Ottawa River regulation modeling system (ORRMS)," *Proceeding of International Symposium on Real-time Operation of Hydrosystems*, vol. 1, pp. 179–198, 1981, Waterloo, Canada. 14

[21] K. L. Hiew, J. W. Labadie, and J. F. Scott, "Optimal operational analysis of the Colorado-Big Thompson project," in *Computerized decision support systems for water managers*, J. Labadie and et al., Eds. Reston, Va.: ASCE, 1989, pp. 632–646. 15

[22] P. Crawley and G. Dandy, "Optimal operation of multi-reservoir system," *J. Water Resou. Plan. Manage.*, vol. 119, no. 1, pp. 1–17, 1993. 15

[23] X. Cai, D. C. Mckinney, L. S. Lasdon, and D. Watkins, "Solving large nonconvex water resources management models using generalized Bender decomposition," *Operation Research*, vol. 49, no. 2, pp. 235–245, 2001. 15

[24] K. Hiew, "Optimization algorithms for large-scale multireservoir hydropower systems," Ph.D. dissertation, Dept. of Civil Engineering, Colorado State Univ., Forts Collins, Colorado, 1987. 15

[25] M. Barros, F. Tsai, S. Yang, J. Lopes, and W. Yeh, "Optimization of large-scale hydropower system operations," *J. Water Resou. Plan. and Manage.*, vol. 129, no. 3, pp. 178–188, 2003. 15

[26] R. Bellman, *Dynamic Programming*, 1st ed. Princeton, New Jersey: Princeton University Press, 1957. 15, 17, 18

[27] R. Larson, *State increment dynamic programming*, 1st ed. New York: Elsevier, 1968. 16

[28] W. Hall, R. Harboe, W. Yeh, and A. Askew, "Optimum firm power output from a two reservoir system by incremental dynamic programming," Water Research Center, University of California, Los Angeles, Tech. Rep., 1969, contribution 130. 16

[29] P. Nopmongcol and A. Askew, "Multilevel incremental dynamic programming," *Water Resou. Res.*, vol. 12, no. 6, pp. 1291–1297, 1976. 16

[30] A. Korsak and R. Larson, "A dynamic programming successive approximation technique with convergence proofs," *Automatica*, vol. 6, no. 2, pp. 253–260, 1970. 16

[31] M. Heidari, V. Chow, P. Kokotovic, and D. Meredith, "Discrete differential dynamic programming approach to water resources system optimization," *Water Resou. Res.*, vol. 7, no. 2, pp. 273–282, 1971. 16

[32] S. Haykin, *Neural networks: A comprehensive foundation*, 2nd ed. U.S.A: Prentice Hall, 1999. 17, 23, 24

[33] A. Charnes and W. Cooper, "Chance-constrained programming," *Management Science*, vol. 6, no. 1, pp. 73–79, 1959. 18

[34] C. Revelle, E. Joeres, and W. Kirby, "The linear decision rule in reservoir management and design, 1, Development of the stochastic model," *Water Resou. Res.*, vol. 5, no. 4, pp. 767–777, 1969. 18

[35] J. R. Stedinger and J. B. Strycharczyk, "Reply," *Water Resou. Res.*, vol. 23, no. 9, pp. 1801–1802, 1987. 18

[36] J. Alvarez-Lopez, K. Ponnambalam, and V. Quintana, "Generation and transmission expansion under risk using stochastic programming," *IEEE Transactions on Power Systems*, vol. 22, no. 3, pp. 1369–1378, 2007. 19, 52

[37] P. Kall and S. W. Wallace, *Stochastic programming.* Chichester, England: John Wiley and Sons, 1994. 19

[38] S. Sen, "Stochastic programming: computational issues and challenges," University of Arizona, Tucson, Tech. Rep., 2001, from Encyclopedia of OR/MS, S. Gass and C. Harris (eds.). 19

[39] J. Jacobs, G. Freeman, J. Grygier, D. Morton, G. Schultz, K. Staschus, and J. R. Stedinger, "A system for scheduling hydroelectric generation under uncertainty," *Annals of Operations Research*, vol. 59, no. 1, pp. 99–133, 1995. 19

[40] S. G. Fletcher and K. Ponnambalam, "Reservoir design optimization using new storage moments equations," *Water Resou. Res.*, vol. 44, 2008. 19, 32, 33, 56, 58, 76

[41] J. L. Zhang and K. Ponnambalam, "Stochastic control for risk under deregulated electricity market - a case study using a new formulation," *Canadian Journal of Civil Engineering*, vol. 32, no. 4, pp. 719–725, 2005. 19, 34, 76

[42] E. L. Thorndike, *Animal Intelligence.* Darien, CT: Hafner, 1911. 20

[43] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*, 1st ed. Belmont, Massachusetts: Athena scientific, 1996. 20, 23

[44] C. Watkins, "Learning from delayed rewards," Ph.D. dissertation, Dept. of Psychology, University of Cambridge, England, 1989. 20, 23, 25, 26, 43

[45] R. Sutton and A. Barto, *Reinforcement learning: An introduction.* Cambridge, Massachusetts: MIT press, 1998. 20, 22, 24, 25, 26, 78

[46] C. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, 1992. 20

[47] B. Lamond and A. Boukhtouta, "Neural approximation for the optimal control of a hydroplant with random inflows and concave revenues," *Journal of Energy Engineering*, vol. 131, no. 1, pp. 72–95, 2005. 20

[48] M. Mahootchi, H. Tizhoosh, and K. Ponnambalam, "Reservoir operation optimization by reinforcement learning," *Proceedings of International Conference on Stormwater and Urban Water Systems*, pp. 165–184, 2006, monograph 15, Chapter 8 in Contemporary Modeling of Urabn Water Systems, James, Irvine, McBean, Pitt and Wright, Eds. 20, 89

[49] J. H. Lee, "Basin-wide multi-reservoir operation using reinforcement learning," Ph.D. dissertation, Department of Civil Engineering, Colorado State University, Fort Collins, Colorado, 2005. 21

[50] J.-H. Lee and J. W. Labadie, "Stochastic optimization of multireservoir systems via reinforcement learning," *Water Resou. Res.*, vol. 43, no. 11, 2007. 21

[51] F. Rosenblatt, *Principles of neurodynamic: Perceptrons and the theory of brain mechanisms*. Washington, DC: Spartman Books, 1962. 21

[52] B. Widrow and M. Hoff, "Adaptive switching circuits," *In IRE WESCON Convention Record Part IV*, pp. 96–104, 1960. 21

[53] A. H. Klopf, "Brain function and adaptive system - A heterostatic theory," Air Force Cambridge Research Laboratory, Bedford, MA., Tech. Rep. AFCRL-72-0164, 1972. 21

[54] A. Barto and R. Sutton, "Landmark learning: An illustration of associative search," *Biological Cybernetic*, vol. 42, pp. 1–8, 1981. 21

[55] H. Robbins and S. Monro, "A stochastic approximation method," *Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400–407, 1951. 23, 24, 52

[56] H. Tizhoosh, "Opposition-based learning: A new scheme for machine intelligence," *Proceedings of the International Conference on Computational Intelligence for Modeling Control and Automation, CIMCA*, vol. I, pp. 695–701, 2005, vienna, Austria. 27

[57] ——, "Opposition-based reinforcement learning," *Journal of Advanced Computational Intelligence and Intelligent Informatics (JACIII)*, vol. 10, no. 4, pp. 578–585, 2006. 27, 44

[58] S. Rahnamayan, H. Tizhoosh, and M. Salama, "Opposition-based differential evolution algorithms," Vancouver BC, 2006, pp. 2010–2017. 27

[59] M. Shokri, H. Tizhoosh, and M. Kamel, "Opposition-based Q(λ) algorithm," in *Neural Networks, 2006. IJCNN apos;06. International Joint Conference*, 2006, pp. 254–261. 27, 76

[60] M. Mahootchi, H. Tizhoosh, and K. Ponnambalam, "Opposition-based reinforcement learning in the management of water resources, Hawaii, U.S.A." *Proceedings of IEEE Symposium Series on Computational Intelligence*, pp. 217–224, 2007. 27

[61] H. Tizhoosh and M. Ventresca, *Oppositional concepts in computational intelligence*. Pysika-Verlag: Springer Berlin, 2008. 28

[62] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition versus randomness in soft computing techniques," *Appl. Soft Comput.*, vol. 8, no. 2, pp. 906–918, 2008. 28

[63] M. Heger, "Consideration of risk in reinforcement learning," in *Proc. of the 11th International Conference on Machine Learning*, W. Cohen and H. Hirsh, Eds. San Francisco: Morgan Kaufmann, 1994, pp. 105–111. [Online]. Available: citeseer.ist.psu.edu/heger94consideration.html 29

[64] M. R. Bessa, "Optimization of the operation of multireservoir systems: A Great Lake case study," Ph.D. dissertation, Dept. of Systems Design Engineering, University of Waterloo, Canada, 1998. 29

[65] R. Neuneier and O. Mihatsch, "Risk sensitive reinforcement learning," in *Advances in neural information processing systems*. Cambridge, MA, USA: MIT Press, 1999, pp. 1031–1037. 29, 52

[66] M. D. Pendrith and M. R. Ryan, "Estimator variance in reinforcement learning: Theoretical problems and practical solutions," In On-line search: collected papers from the 1997 Workshop, AAAI Technical Report WS-97-10, AAAI Press, pp. 81–88, 1997. 29

[67] P. Geibel, "Reinforcement learning with bounded risk," in *Proc. 18th International Conf. on Machine Learning.* Morgan Kaufmann, San Francisco, CA, 2001, pp. 162–169. [Online]. Available: citeseer.ist.psu.edu/geibel01reinforcement.html 30

[68] M. Sato and S. Kobayashi, "Variance-penalized reinforcement learning for risk-averse asset allocation," in *Intelligent Data Engineering and Automated Learning, Data Mining, Financial Engineering, and Intelligent Agents.* Springer-Berlin, 2008, pp. 333–364. 30

[69] L.-R. Chen and S. Ghosh, "Modeling and simulation of a hierarchical, distributed, dynamic inventory management scheme," *Simulation*, vol. 68, no. 6, pp. 340–362, 1997. 35

[70] A. Gupta and C. Maranas, "Managing demand uncertainty in supply chain planning," *Computers and Chemical Engineering*, vol. 27, no. 8-9, pp. 1219–1227, 2003. 35

[71] C. L. Chen and W. C. Lee, "Multi-objective optimization of multi-echelon supply chain networks with uncertain product demands and prices," *Computers and Chemical Engineering*, vol. 28, pp. 1131–1144, 2004. 36

[72] M. Mahootchi, K. Ponnambalam, and H. Tizhoosh, "Comparison of risk-based optimization models for reservoir management," *Canadian Journal of Civil Engineering*, under revision. 50

[73] F. Sahba, H. Tizhoosh, and M. Salama, "A Reinforcement learning framework for medical image sSegmentation," in *Proc. of International Joint Conference on Neural Networks (IJCNN 06)*, 2006, pp. 511–517. 53

[74] ——, "Application of opposition-based reinforcement learning in image segmentation," in *Proc. of IEEE Symposium on Computational Intelligence in Image and Signal Processing (CIISP)*, 2007, pp. 246–251. 53

[75] P. Kumaraswamy, "A generalized probability density function for double-bounded random processes," *Journal of Hydrology*, vol. 46, no. 1-2, pp. 79–88, 1980. 67, 68

[76] M. Jones, "Kumaraswamy's distribution: A beta-type distribution with some tractability advantages," *Statistical Methodology*, 2008, doi:10.1016/j.stamet.2008.04.001. 68

[77] D. Poole, "Q-learning," Website, 2004, http://www.cs.ubc.ca/~poole/demos/rl/q.html. 76, 77

[78] K. Ponnambalam, "Optimization of the integrated operation of multi-reservoir irrigation systems," Ph.D. dissertation, Dept. of Civil Engineering, University of Toronto, 1987. 99, 105, 109

[79] D. Hochbaum, "Complexity and algorithms for nonlinear optimization problems," *Annals of Operations Research*, vol. 153, no. 1, pp. 257–296, 2007. 107