DISSERTATION

OPTIMAL RESERVOIR OPERATIONS FOR RIVERINE WATER QUALITY IMPROVEMENT: A REINFORCEMENT LEARNING STRATEGY

Submitted by

Jeffrey Donald Rieker

Department of Civil and Environmental Engineering

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Spring 2011

ABSTRACT

OPTIMAL RESERVOIR OPERATIONS FOR RIVERINE WATER QUALITY IMPROVEMENT: A REINFORCEMENT LEARNING STRATEGY

Complex water resources systems often involve a wide variety of competing objectives and purposes, including the improvement of water quality downstream of reservoirs. An increased focus on downstream water quality considerations in the operating strategies for reservoirs has given impetus to the need for tools to assist water resource managers in developing strategies for release of water for downstream water quality improvement, while considering other important project purposes. This study applies an artificial intelligence methodology known as reinforcement learning to the operation of reservoir systems for water quality enhancement through augmentation of instream flow. Reinforcement learning is a methodology that employs the concepts of agent control and evaluative feedback to develop improved reservoir operating strategies through direct interaction with a simulated river and reservoir environment driven by stochastic hydrology. Reinforcement learning methods have advantages over other more traditional stochastic optimization methods through implicit learning of the underlying stochastic structure through interaction with the simulated environment, rather than requiring *a priori* specification of probabilistic models. Reinforcement learning can also be coupled with various computing efficiency techniques as well as other machine learning methods such as artificial neural networks to mitigate the “curse of dimensionality” that is common to many optimization methodologies for solving sequential decision problems.

A generalized mechanism is developed, tested, and evaluated for providing near-real time operational support to suggest releases of water from upstream reservoirs to improve water quality within a river using releases specifically designated for that purpose. The algorithm is designed to address a variable number of water quality constituents, with additional flexibility for adding new water quality requirements and learning updated operating strategies in a non-stationary environment. The generalized reinforcement learning algorithm is applied to the Truckee River in California and Nevada as a case study, where the federal and local governments are purchasing water rights for the purpose of augmenting Truckee River flows to improve water quality. Water associated with those acquired rights can be stored in upstream reservoirs on the Truckee River until needed for prevention of water quality standard violations in the lower reaches of the river.

This study shows that in order for the water acquired for flow augmentation to be fully utilized as a part of a longer-term strategy for water quality management, increased flexibility is required as to how those waters are stored and how well the storage is protected from displacement through reservoir spill during times of high runoff. The results show that with those flexibilities, the reinforcement learning mechanism has the ability to produce both short-term and long-term strategies for the use of the water, with the long-term strategies capable of significantly improving water quality during times of drought over current and historic operating practices. The study also evaluates a number of variations and options for the application of reinforcement learning methods, as well as use of artificial neural networks for function generalization and approximation.

ACKNOWLEDGEMENTS

Credit for the completion of this dissertation lies with the Lord Jesus Christ, through whom all things are possible.

I would like to thank my advisor Dr. John Labadie for his help and guidance through the completion of this project. It has been an honor and privilege to work with him through the lengthy duration of the project. I would also like to thank my entire doctoral committee for their support throughout, including Dr. Darrell Fontane, Dr. Donald Frevert, and Dr. Charles Anderson. I appreciate Don Frevert's role in connecting me with the Bureau of Reclamation, creating opportunities for me like this one and many others. I would additionally like to acknowledge the Bureau of Reclamation for their support of this project.

I thank my wife, Cathy Rieker, for her immeasurable love and support through the completion of this project. It would not have been a reality without her, and the countless hours she spent caring for our family and household which provided me the chance to see the project to completion. Thanks to our children Ryan, Megan, and Erin, for allowing me to sneak away for a portion of their early years to accomplish this. I also thank all those of our family who supported both Cathy and I through this effort; Donald and Marilyn Rieker, Tom and Linda McDavid, Greg Rieker and Julie Steinbrenner, Geoff and Susan Barker and their children Alex and Elizabeth, Grier and Karen Laughlin and their children Addison, Grier, and Caroline, as well as the numerous members of our extended families.

My appreciation also goes to my younger brother Dr. Greg Rieker, who completed his doctoral work in advance of me. This was a special opportunity for me to follow him in this achievement, and his help and guidance were critical to my project's completion.

Thanks to all those who assisted with the review and completion of this dissertation, including my parents Don and Marilyn Rieker, Kim Mitchell, Tom Strekal, Tom Scott, Jaime Trammell, and several others. Many thanks to our model development team; including Shane Coors, Heather Gacek, Mike Mann, Tom Scott, Jeff Boyer, Pat Fritchel, Nola Mitchell, and many others. I also appreciate all of those that provided tremendous support to both myself and my wife Cathy during this work, including Kenneth and Callie Parr, Betsy Rieke, Joe and Jill Andrews, Jay and Paula Frey, Marc and Amber Walling, Karen Lamb, AnneMarie McCann, and countless others who are too numerous to list. Thanks to all!

TABLE OF CONTENTS

LIST OF FIGURES

TABLE OF ACRONYMS

acre-foot	Volumetric equivalent to one acre, one foot deep
ADP	Active dynamic programming
ANN	Artificial neural network
cfs	Cubic foot/feet per second
cms	Cubic meters per second
COE	US Army Corps of Engineers
CPU	Central processing unit
DOI	Department of the Interior
DOJ	Department of Justice
DP	Dynamic programming
DRI	Desert Research Institute
DSS	Decision support system
DSSAM/DSSAMt	Dynamic Stream Simulation and Assessment Model (with temperature)
EIS	Environmental Impact Statement
EPA	US Environmental Protection Agency
FWM	Federal Water Master
GPI	Generalized policy iteration
HEC-PRM	Hydrologic Engineering Center - Prescriptive Reservoir Model

HSPF	Hydrologic Simulation Program - Fortran
LCT	Lahontan cutthroat trout
Local agencies	City of Reno, City of Sparks, and Washoe County
MC	Monte Carlo
MCM	Million cubic meters
MDP	Markov Decision Process
NCDC	National Climatic Data Center
NDEP	Nevada Department of Environmental Protection
NOAA	National Oceanographic and Atmospheric Administration
OCAP	Operating Criteria and Procedures for the Newlands Project, Nevada
OFU	Optimism in the face of uncertainty
PSA	Preliminary Settlement Agreement
Reclamation	Bureau of Reclamation
TCID	Truckee-Carson Irrigation District
TD	Temporal difference
TDS	Total dissolved solids
TMDL	Total maximum daily load
TMWA	Truckee Meadows Water Authority
TMWRF	Truckee Meadows Water Reclamation Facility
TrHSPF	Truckee River HSPF Model
Tribe	Pyramid Lake Paiute Tribe
TROA	Truckee River Operating Agreement
TROM	Truckee River Operations Model
Truckee Meadows	Reno-Sparks metropolitan area

TRWQ	Truckee River Water Quality Model
USGS	United States Geological Survey
WARMF	Watershed Analysis Risk Management Framework
WASP	Water Quality Analysis Simulation Program
WBO	Weather Bureau Office
WQSA	Water Quality Settlement Agreement

1 – INTRODUCTION

1.1 – Background and Motivation

Complex water resources systems often involve a wide variety of competing objectives and purposes. Improvement of water quality downstream of reservoirs is one objective that sometimes exists on regulated river systems. Reservoir operations and other demands on river systems alter the timing and amount of flow in a river, often with impacts to the natural environment such as degraded quality of water in the river. Pollutant loadings to the river also tend to degrade water quality, introducing a need for mitigation through alternative approaches to reservoir operations. In recent decades, there has been an increased focus on downstream water quality considerations in the operating strategies for reservoirs, and in many locations reservoir operations are being reviewed to evaluate methods to improve downstream water quality while still meeting the other objectives of the reservoir system (Kerachian and Karamouz 2007; Shirangi, Kerachian et al. 2008). Specific amounts of water are even being allocated for the purpose of improving downstream water quality through instream flow augmentation (Reno, Sparks et al. 1996). For this reason, there is an increasing need for tools to assist water resource managers in making decisions on how and when to release water specifically allocated for downstream water quality improvement, while taking into consideration water released for other objectives and the interrelated benefits and impacts caused by releases for the various objectives.

The Truckee River in California and Nevada serves as a case study for the need to develop strategies for the use of flow augmentation for downstream water quality improvement. The federal government, in association with the local government agencies in and around Reno, NV, is purchasing water rights within the Truckee River basin. The water rights purchasing program is a settlement of litigation that resulted from the approval and operation of the Reno-Sparks wastewater treatment facility. The purchased water is to be used solely for the purpose of augmenting Truckee River flows to improve water quality. The Bureau of Reclamation (Reclamation), which is involved with the settlement, is analyzing options for the management of the newly purchased water, so that it may suggest an improved long-term strategy for use of the purchased water to the group of agencies that will be coordinating its storage and release from basin reservoirs. Due to the high variability in daily flow and water quality on the Truckee River, a daily operational decision support system (DSS) will be required to carry out the strategy.

A previous effort provided an initial review of water quality issues in the Truckee River, and suggested potential operating strategies for the purchased water (Neumann 2001; Neumann, Rajagopalan et al. 2003). This resulted in an initial problem solution based on an operating policy designed to focus on short-term improvements to a single water quality variable. Recommendations for future research were provided, including the development of a more generalized system capable of adapting to other water quality variables and models, as well as a system capable of focusing on longer-term improvements to water quality rather than more immediate improvements (Neumann 2001).

The challenges in developing a generalized system to assist with decisions for management of water quality in any river basin include those associated with any multi-objective environment. Many of the objectives can be competitive in nature, and complex operating policies developed through years of litigation, regulation, court order, and institutional experience can define a river and reservoir system which has large variability within each day's basic operation. In addition, many demands on the river, including the demand for improved water quality, cannot always be fully met due to the scarcity of the water resource. The result is a system which is difficult to model using traditional statistical methods, and thus an optimization problem that taxes the application of more traditional solution methods. The development of a solution that focuses on water quality issues in a complex river and reservoir environment requires a system capable of seeking optimal overall strategies, but also capable of reacting to observed conditions in a real-time situation. It must also be capable of learning new strategies as conditions change within the environment, thus having the ability to deal with non-stationary surroundings.

A wide variety of optimization techniques have been applied in a theoretical sense to multi-objective issues within the field of water resources, and many have been applied to water quality problems more specifically. However, most optimization techniques applied in past studies have encountered significant hindrances, and practical application of these techniques has been less common (Labadie 2004). Development of a system that is useful to water managers and will be practically applied by water managers is an imposing task. Important considerations include the amount of time and computing resources required to operate complex simulation and optimization models on current desktop computing systems, as well as ease of use by water resource managers.

1.2 – Objectives of Study

The objectives of this study include the development, testing, and evaluation of a generalized mechanism based on artificial intelligence methods known as reinforcement learning that can suggest releases of water from reservoirs for the improvement of downstream water quality within a river using water specifically dedicated and stored in reservoirs for that purpose. The reinforcement learning methodology is applied to the case study on the Truckee River as a demonstration of its efficacy. The methodology provides an overall strategy for the management of water quality using the dedicated water supply, but also affords near real-time operational decision support based on that strategy. The methodology allows adaptive and flexible modification of strategies based on changing conditions within the environment of the river and reservoir system. The reinforcement learning system is generalized to interface with any set of preexisting hydrologic and water quality models, making it applicable to basins where physically-based models have already been developed. The methodology is capable of accommodating a variable number of water quality constituents, and sufficiently flexible for addressing the addition of new water quality requirements or goals in the future.

1.3 – Contribution of This Research

This research demonstrates the application of an emerging technology within the water resources field known as reinforcement learning, and how it can be used in the development of reservoir operating strategies to address downstream water quality issues. The research develops a generalized set of tools capable of application to any river system with similar issues, and which can be applied to systems with different preexisting hydrologic and water quality models. New methods of implementing reinforcement learning technology are developed for improving performance and sustaining viable results. The research shows how judicious application of reinforcement learning can take full advantage of parallel processing capabilities currently available on desktop computing systems, thereby greatly enhancing efficiency and ease of use.

The research illustrates the ability of reinforcement learning to provide an effective linkage between simulation and optimization. Modeling efforts often focus on the development and application of a simulation model capable of accurately replicating a river and reservoir system, but without the ability to develop optimal solutions. Optimization efforts often focus on development of the best solutions, while sacrificing accuracy in modeling the physical environment. This research and application of reinforcement learning shows the linkage of both simulation and optimization in a way that takes advantage of the abilities of both approaches.

The reinforcement learning methodology is applied to a case study on the Truckee River, and since it is constructed within the framework of other models and DSS's being developed by the parties that will be involved in the release scheduling for the purchased water on the Truckee River, it is expected that it will be implemented in practice as one of the primary tools for deciding how to use the purchased water. Though artificial intelligence methods have been applied to water resources issues in the past, the science is still new to the water resources field and real-world application of the theories is currently rare. This application will help bridge the gap between the academic theories presented herein and the real-world operation of a reservoir system. In addition, it provides a written synthesis of the history and summary of the current state of knowledge with respect to water quality issues on the Truckee River, as well as hydrologic and water quality modeling on the river.

The research develops strategies for more efficient use of the tabular methods that form the traditional basis for reinforcement learning algorithms. Further, it evaluates generalization strategies for reinforcement learning algorithms. The primary generalization technique evaluated is the use of artificial neural networks, another artificial intelligence technology gaining popularity in the water resources field.

Additionally, the research addresses difficult challenges with the application of reinforcement learning to the problems presented, including issues related to the limited water supply available for meeting water quality objectives, and therefore the limited ability for the reinforcement learning mechanism to fully experience and explore the state and action space. These challenges are addressed, in part, through application of a variety of different action exploration methods, including some which are not as extensively utilized in association with reinforcement learning.

1.4 – Organization of This Dissertation

This dissertation is organized into six chapters. The current chapter provides an introduction to the study problem. The second chapter gives a review of past studies which provide support for the current effort, while ensuring that this study provides a unique and beneficial contribution to the profession. The third chapter develops the methodology that is used for the research by providing background into the fields of artificial intelligence and reinforcement learning, and detailing the development of a reinforcement learning agent for the improvement of water quality. The fourth chapter describes the application of the methodology to the Truckee River case study, including an introduction to the study area and background on pertinent water quality issues and legal agreements within the study area. The fifth chapter discusses and analyzes the results of that application. The sixth chapter provides a summary, conclusions, and recommendations for future work.

2 – LITERATURE REVIEW

A literature review was completed providing background into previous integration of water quality concerns with operational decision support. This review included both “traditional” and artificial intelligence/machine learning methods for decision support and optimization.

2.1 – Decision Support and Optimization for Water Quality Management

Computer models have been used extensively over recent decades to simulate most aspects of hydraulics, hydrology, and river systems management. Additionally, optimization algorithms have been added to these models for the purpose of improving the management of river basins through improved operation of reservoirs and other water control structures. These simulation and optimization models are finding increasing operational use through the implementation of decision support systems, which are used to aid water managers in short, middle, and long term planning of water control operations (Labadie 2004). This section reviews methods currently in use for the management of water quality within river basins around the world. The methods are separated into two distinct categories; traditional methods and methods based on artificial intelligence technologies. For the purposes of this review, traditional methods are considered to be methods that have been utilized in the water resources field for many years. These are the methods that generally use a variety of equations to produce optimal or near-optimal results based on some combination of inputs, constraints, and goals. In contrast, this review considers artificial intelligence methods to be those based on theories taken from the computer science study of intelligence and some forms of computer learning. These are the methods that generally have not been used or have only very recently been tested in the water resources field, and which focus on the ability of a computer to learn optimal or preferred actions from experience with a simulated environment. Often these methods have been inspired or derived from methods used by living organisms to evolve or learn. It should be noted that methods considered "statistical learning" will be classified into the category of traditional methods for the purposes of this review. These are the methods that allow a computer to learn the characteristics of a problem primarily by focusing on statistics of the inputs provided.

2.1.1 – Traditional Methods

Traditional decision support and optimization methods have been studied and applied to water resources management for several decades. Often these investigations and applications focus on multi-objective optimization and decision support, and water quality improvement is often one of the many objectives. One of the more common applications has been the use of these techniques for the determination of optimal pollutant loadings to a river, often from waste water treatment plants.

A comprehensive review of previous studies involving multi-objective river basin optimization and decision support is outside the scope of this research, however previous efforts provide extensive reviews of this subject (Yeh 1985; Wurbs 1993; Labadie 1998; Labadie 2004; Labadie 2005; Wurbs 2005; Rani and Moreira 2010). Labadie (Labadie 2004) summarizes traditional optimization schemes that have been applied to river basin management as follows:

- Implicit Stochastic Optimization
    - Linear programming
    - Network flow optimization
    - Nonlinear programming
    - Discrete dynamic programming
    - Differential dynamic programming
    - Discrete-Time Optimal Control Theory
- Explicit Stochastic Optimization
    - Chance-constrained programming
    - Stochastic linear programming
    - Stochastic dynamic programming
    - Stochastic optimal control
    - Multiobjective optimization methods

These methods have been applied in various decision support systems, as discussed in Labadie and Wurbs (Labadie 2004; Wurbs 2005). Some of the current generalized decision support systems using these methods include WRIMS (CalSim), MODSIM, HEC-ResSim, and RiverWare, among others.

2.1.2 – Artificial Intelligence and Reinforcement Learning Methods

During the past two decades, computing methods from the field of artificial intelligence have gained popularity in water resources modeling and optimization. A survey of recent literature reveals a cross-section of efforts that apply artificial intelligence and machine learning techniques to various water resources issues. Two of the more prevalent methods in use are genetic algorithms and artificial neural networks. Genetic algorithms are essentially search methods designed to find optimal solutions to problems based very generally on evolutionary theory, as discussed in greater detail in Section 3.1.2.1 of this document. Artificial neural networks, also referred to as ANN's or neural networks, are data classification and regression methods based generally on the functionality of the brain, as discussed in greater detail in that same section of this document.

Labadie provides an overview of a variety of applications of genetic algorithms for the optimization of river and reservoir operations. Labadie also discusses several applications of neural networks to river and reservoir operations optimization (Labadie 2004). Rani et al. also provides an overview of applications of neural networks and genetic algorithms (Rani and Moreira 2010). Solomatine discusses the use of neural networks, among other computational intelligence techniques, in water resources control applications (Mohammadian, Sarker et al. 2003). Specific efforts discussed in the literature provide details on various uses of neural networks, most commonly as applied to the issue of reservoir operation and hydraulic network or sewer system operation (Raman 1996; Solomatine 1996; Savic and Walters 1999; Lobbrecht and Solomatine 2002; Chaves and Kojiri 2007; Darsono 2007). As a result of the rise in popularity of neural networks in the field of water resources, the American Society of Civil Engineers formed a task committee which reported on the technology and its applications in the field (ASCE 2000; ASCE 2000).

Genetic algorithms and neural networks have also gained popularity in the specific field of water quality management. Otero et al. and Wan et al. applied genetic algorithms to the optimal management of salinity in estuaries (Otero 1995; Wan, Labadie et al. 2006). Wen et al. used a neural network in association with multi-objective optimization to attempt to optimize the management of water quality in a river basin (Wen 1998). Chen et al. describe a genetic algorithm coupled with fuzzy programming that develops water quality management strategies for a river basin (Chen 1998). Roehl et al. utilized neural networks in the control of salinity issues in a tidally affected river basin (Roehl, Conrads et al. 2000). Chaves coupled neural networking with fuzzy stochastic dynamic programming and genetic algorithms to review optimal operation of a reservoir for water quality purposes (Chaves 2004; Chaves and Kojiri 2007). In several of these applications, neural networking technology was applied to the modeling of water quality, which is an area where much focus has been applied in recent decades. The usefulness of neural networks in dealing with the complexities of water quality modeling is revealed through a review of a small segment of the wide variety of past applications (Maier 1996; Maier 2000; Rounds 2002; Risley, Roehl et al. 2003; Suen 2003).

More recently, research and application of reinforcement learning technology has begun to take place in the field of water resources planning and management. In 1996, Wilson introduced reinforcement learning as a potential technique for real-time optimal control of hydraulic networks (Wilson 1996). Wilson noted that one of the significant issues facing reinforcement learning at the time was the ability to generalize and approximate the functions that form a basis for the technology. Wilson also indicated that neural networks, coarse-coded look-up tables, memory-based approximators, decision trees, and classifier systems were available to address the issue, recommending the coarse-coded look-up table option as the most appropriate. He also noted that though the technology could be extended to deal with multi-objective issues, no results had yet been reported on that type of application. Bouchart et al. presented a reinforcement learning model for the control of multiple reservoir systems (Bouchart 1998). However, Bouchart et al. noted that the performance of the reinforcement learning model remained sub-optimal due to the lack of techniques available to provide reservoir inflow sequences during the training of the reinforcement learning model that would train the model to perform optimally (Savic and Walters 1999). That paper described a genetic algorithm approach to develop training inflow sequences to improve performance of the reinforcement learning mechanism.

Castelleti et al. proposed a new approach which they referred to as “Q-learning planning” or “Qlp”, and applied the approach to the operational management of a reservoir (Castelletti 2002). The approach essentially integrated a Q-learning mechanism from reinforcement learning technology with traditional stochastic dynamic programming mechanisms to produce a hybrid system for reservoir management. Bhattacharya et al. developed a real-time control system using reinforcement learning coupled with a neural network for the control of water drainage in low-lying areas in The Netherlands (Bhattacharya 2003). In this application, several neural networks were utilized, including one to address the function approximation issue of the reinforcement learning mechanism.

Lee and Labadie created a multi-reservoir operations control mechanism based on the Q-learning approach of reinforcement learning (Yi 2005; Lee and Labadie 2007).

That application focused on multi-objective optimization of a two reservoir system, and compared the results of the operating rules developed through reinforcement learning to those developed with implicit stochastic dynamic programming and sampling stochastic dynamic programming. Generalization and function approximation were accomplished by a technique known as K-means clustering, and applied to a tabular implementation of the reinforcement learning mechanism with improved success over simple percentile partitioning of the definition of the current state of the hydrologic and reservoir system (Lee and Labadie 2007). Castelletti et al. applied reinforcement learning to the issue of controlling a selective withdrawal structure on a reservoir to optimize water quality targets in the reservoir and immediately downstream (Castelletti, Garbarini et al. 2009). Though this was an application of reinforcement learning to a water quality issue, that study differs significantly from this dissertation research since it was a focused application of reinforcement learning on the selective withdrawal process from a reservoir rather than flow augmentation for river water quality.

More recent applications of reinforcement learning technology to water resources issues have also coupled reinforcement learning mechanisms with other emerging technologies to attempt to produce alternative solutions. Mariano-Romero et al. developed a hydraulic network optimization scheme based on multi-objective distributed Q-learning, which essentially utilized a multi-agent approach to Q-learning (Mariano-Romero 2007). Abolpour et al. combined reinforcement learning with fuzzy logic in an application to improve river basin water allocation (Abolpour 2007). Mahootchi et al. initially developed optimal operational policies for a single reservoir system using Q-learning (Mahootchi, Tizhoosh et al. 2007), and then went on to combine Q-learning with opposition-based learning in the development of operational policies for a reservoir system (Mahootchi 2007; Tizhoosh and Ventresca 2008).

Thus far, most applications of reinforcement learning technologies to reservoir operations and multi-reservoir operations have been focused on optimization of complete reservoir operations in scenarios where most normal operating policies are able to be adequately tested and evaluated for their usefulness. These scenarios ensure that the reinforcement learning agent has the ability to sufficiently learn the value of its actions from reasonably long sequences of simulated experience, assisting with the development of an optimal or near-optimal operating policy. No studies were found in the literature that utilize reinforcement learning to develop an agent that focuses on specific reservoir operations for downstream water quality management by using flow augmentation.

3 – ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

This study applies artificial intelligence and machine learning methods to the scheduling of reservoir releases for the improvement of water quality. The end result is a DSS which is constructed using selected methods from the artificial intelligence field of study. This chapter summarizes the development of those methods and the DSS. The system is tested on a case study focused on the release of WQSA-stored water on the Truckee River, which is discussed in the proceeding chapters.

The first section of this chapter presents a review of the general fields of artificial intelligence and machine learning, detailing the decision support methods available from within these fields. These methods generally come in the form of “rational agents” that have the ability to learn over time. From this review, a particular “rational agent” method is selected for application to the study problem. The section goes on to discuss a learning mechanism that is particularly well-suited to the selected method known as “reinforcement learning”. The second section of this chapter details the development of the generalized DSS using the selected rational agent and learning method.

3.1 – Artificial Intelligence and Machine Learning

3.1.1 – Artificial Intelligence

3.1.1.1 – The "Rational Agent"

The field of "artificial intelligence" encompasses the study and development of computer systems that either think or act in a human-like manner, or alternatively in a rational way. Rationality is generally defined as operating in an optimal fashion based on knowledge and experience. One of the main focuses within the field of artificial intelligence is the creation of systems that have the ability to perceive and act on the surrounding environment in a rational way. Such a system is known as a "rational agent", and often these systems have other attributes such as the abilities to adapt to change or work under autonomous control. The field of artificial intelligence is closely linked to related fields such as statistics, control theory, operations research, and decision theory. However, artificial intelligence has been set apart as a unique field due to its focus on certain human characteristics and the application to computers (Russell and Norvig 2003).

The development of a rational agent is the primary focus of this research. The rational agent should be able to perceive the surrounding environment using sensors, and act upon it using actuators. The agent behaves rationally by maximizing its performance while acting on its environment, using an internal mapping between the perceptual inputs (state and performance information from the environment) it receives and the actions it takes. This mapping is known as the “agent function,” which represents the agent’s understanding of the environment upon which it is acting and the effects of its actions. This agent function can be information that was supplied to the agent at the time of its creation, but in order to operate at a higher level, a rational agent needs to have the ability to learn from its actions and alter the function based on new knowledge and experience. This ability to learn also grants a level of autonomy to the rational agent. The agent function (which is actually a theoretical abstraction) is implemented in an “agent program”. The program operates on the agent’s “architecture”, which consists of the physical computing device and potentially (depending on the nature of the agent) the physical actuators and sensors (Russell and Norvig 2003).

The rational agent is primarily designed to maximize its performance. For this reason, the agent must be provided with a performance measure from which it may evaluate the degree to which it succeeds in its attempts to act upon its environment. The development of the performance measure is a very important aspect of the design of the rational agent, as this component of the agent will generally determine the agent’s overall behavior and mode of operation. A “greedy” agent will tend to always act in a way that maximizes its immediate performance by selecting actions that return the highest immediate value from the performance measure, essentially exploiting its knowledge of the performance measure. In order to learn from its actions and gain new knowledge, it is important for an agent to occasionally explore other actions that do not have the highest immediate reward. By acting in a non-greedy fashion a certain percentage of the time, the agent may learn actions that improve the ability to achieve its goals in the longer term, while not necessarily maximizing immediate reward. The ability to balance actions between exploitation of immediate rewards and exploration of non-greedy actions is key to the development of a successful agent (Russell and Norvig 2003).

There are four general categories of rational agents currently in use. These are simple reflex agents, model-based reflex agents, goal-based agents, and utility-based agents. The simple reflex agent basically has a simple internal mapping of the actions that should be taken by the agent depending on the state of the environment encountered by the agent. This could be compared to the concept of basic stimulus-response rules. This type of agent is usually not well suited to complex environments, as it must have knowledge of the exact action that is taken under any perceived condition. Additionally, this agent must be able to fully observe the state of its environment at all times in order to act according to the rules that govern its operation. The model-based reflex agent is an extension of the simple reflex agent that attempts to overcome this limitation by maintaining an internal representation of the most likely state of the agent's environment. The internal representation of the environment serves as a model of what the environment may be like when the agent is unable to fully observe the actual environment, and essentially serves as a way for the agent to attempt to continue to act on the environment around it even in the event that it cannot sense all parts of the environment at all times (Russell and Norvig 2003).

The goal-based agent extends beyond the simple mapping between perceived states and actions by allowing the agent to pursue a goal. By orienting the agent around a goal, the agent no longer simply reacts to the state of the environment, but is forced to evaluate possible actions and choose the action that will achieve the goal. In more complex environments, this may involve taking more than one action. An agent designed for this type of environment generally uses techniques that search for a series of actions that achieve the goal. In complex environments where a simple search through all possible actions is not possible, the agent can be designed to use planning techniques in order to achieve the goal. Another type of goal-based agent is the knowledge-based agent, which develops knowledge about the environment and uses logic and reasoning to select actions that will achieve the desired goal (Russell and Norvig 2003).

The utility-based agent is a more generalized form of the goal-based agent that is better suited to environments with multiple or conflicting goals, or environments where goals cannot always be fully achieved. This type of agent attempts to maximize its performance given that perfect performance is not likely or even possible. The performance of the agent is continuously measured by the degree of goal achievement, which is a value placed on the state of the environment the agent is in. These values are also known as the “utility” of each state. The “utility function” is a mapping of states to utility values, and with complete knowledge of the utility function of the states of the environment the agent may achieve, the agent can attempt to reach states that are more desirable or which maximize the degree to which it has achieved its goals. Alternatively, the utility function can map actions to utility values, allowing the agent to maximize the use of actions from particular states that will likely result in achievement of the agent’s goals. In an uncertain environment, if the agent also has knowledge of the probabilities with which its actions can achieve certain states, the agent can attempt to maximize its expected utility. This is known as the “principle of Maximum Expected Utility”, and the attempt to make rational decisions using a combination of utilities with probability is called “decision theory”. An agent operating under these theories is known as a “decision-theoretic agent” (Russell and Norvig 2003).

Each of the different types of agents has the ability to become a “learning agent”. The ability to learn from experience is a key component to most agents. It is a method for the agent to improve its performance. Because of this, a learning agent generally does not need any initial knowledge of the environment or the actions it should take, since it has the ability to completely learn its agent function from scratch. Alternatively, prior knowledge of the environment can help reduce the amount of time required for an agent to perform optimally, or can prevent the agent from entering extremely poor situations that have significant consequences. Agents generally learn from experience by improving their knowledge of the environment or the effects of their actions. This may include the knowledge of the results of actions for simple reflex agents, the knowledge of how the environment changes over time for model-based reflex agents, the knowledge of which actions achieve goals for a goal-based agent, and the knowledge of the utility of states and/or the ability for actions to find the highest utility (best action policy) for utility-based agents (Russell and Norvig 2003).

The concept of a rational agent is central to the study of artificial intelligence. A rational agent should have the ability to act on its environment to maximize its performance measure, and advanced agents have the ability to learn from the environment in order to improve performance. The different categories of agents and some specific types of agents are summarized in Table 3.1 as follows:

Table 3.1 – Summary of agent categories, agent functions, and learning mechanisms

Agent Type	**Agent Function**	**Learning Mechanism**
Reflex-based Agents - Simple reflex - Model-based reflex	If-Then or Stimulus-Response rules	Methods that improve the agent's response rules (or model of the environment in the case of a model-based agent)
Goal-based Agents - Search and planning - Knowledge-based	Knowledge base of solutions or reasoning mechanisms to find solutions	Methods that improve the agent's knowledge base or reasoning capabilities
Utility-based Agents - Decision-theoretic	Policy function based on a utility function	Methods that improve the agent's utility function

3.1.1.2 – The "Task Environment"

The environment occupied by the rational agent is known as the "task environment." The full specification of the task environment includes not only the physical (or virtual) surroundings of the agent, but also the performance measure by which the agent is judged, the agent's actuators, and its sensors. Specification of the task environment is an important aspect of the creation of a system based on artificial intelligence methods, because this specification will have a large impact on the degree to which the rational agent can achieve the desired goals or behavior (Russell and Norvig 2003).

The specification of the agent's physical or virtual surroundings within the task environment can also be categorized by a number of attributes, as outlined here (Russell and Norvig 2003):

- Fully or partially observable as seen by the agent
- Deterministic or stochastic with respect to transitions between states
- Episodic or sequential with respect to the agent reaching "terminal" states
- Static or dynamic as to whether the agent has time to respond before change occurs
- Discrete or continuous with respect to the nature of the states experienced by the agent
- Single or multiple agents operating in the environment
    - Competitive or cooperative agents if a multiagent environment

The most complex environment is one that combines the more difficult attributes listed; essentially one that is partially observable, stochastic, sequential, dynamic, continuous, and where multiple agents will be acting. It is important to provide the agent with the simplest specification of the given environment, while allowing the agent's sensors to take in all pertinent information that has an impact on the desired agent behavior.

The specification of the performance measure is one of the most important parts in the development of a system based on artificial intelligence methods. The performance measure needs to identify the degree of success for an agent's actions, but care needs to be taken to prevent the agent from seeking actions that provide immediate reward from the performance measure without seeking the overall desired outcome. For this reason, it is generally best to design a performance measure that focuses on the overall goal of the agent rather than immediate sub-goals. Additionally, the performance measure should be one component of the agent's task environment that it does not have control over. This prevents the agent from altering the performance measure to meet the outcome of its own actions (Sutton and Barto 1998; Russell and Norvig 2003).

3.1.2 – Machine Learning

The field of machine learning is focused on the ability to understand and learn about input information using computational methods. Common to most types of machine learning is the concept of input data; the information that is to be better understood and learned about. The differences between types of machine learning arise from the different types of feedback available to the learner. The next three sections outline the three primary types of learning; supervised learning, unsupervised learning, and reinforcement learning (Russell and Norvig 2003).

3.1.2.1 – Supervised Learning

Supervised learning encompasses the set of problems that have output values which are directly dependent on the input data. The goal of a supervised learning problem is to learn a function that predicts the dependent output values based on the independent input values. The function is learned by observing input and output values from a training dataset. The output values serve as a "teacher" or "supervisor," providing feedback to the learner about the usefulness of the learned function. This type of learning is also known as inductive learning, where the function represents a hypothesis that is tested using the example data. The supervised learning problem is generally broken down into two categories. The two categories include regression, which deals with quantitative and usually continuous data, and classification, which deals with qualitative information (Hastie, Tibshirani et al. 2001; Russell and Norvig 2003).

Supervised learning is commonly, but not always, accomplished using statistically-based methods. Some supervised learning methods include but are not limited to:

- Linear regression
- Linear classification
- Kernel methods
- Decision tree methods
- Nearest neighbor methods
- Neural Networks

3.1.2.1.1 – Neural Networks and Genetic Algorithms

One form of supervised learning of particular interest to this study is that of artificial neural networks. The creation of neural networks was one of the earliest topics of artificial intelligence research, focusing on the creation of a computing structure that would mimic the basic functioning of brain cells. The neuron is a brain cell that generally collects and processes electric signals. In computing, neurons are "nodes" or "units" that have one or more input signals and an output signal. The input signals generally carry numeric information known as an "activation", which is generally information from a particular sector of the environment or the output from another unit. The neuron or unit also contains an "activation function". Each input to the unit has a numeric weight applied to it to determine the strength and sign of that particular input, and the sum of the weights multiplied by the input activations is provided to the activation function to generate an output value. The functionality of a unit is shown in Equation 3.1:

$$a_i = g\left(\sum_{j=0}^{n} W_{j,i} a_j\right) \qquad \text{(Eqn 3.1)}$$

where:

$a_i = Unit\ output$

$a_j = Unit\ input$

$W_{j,i} = Input\ weight$

$n = Number\ inputs$

$g = Activation\ function$

The activation function is designed to be “active” if the sum of the inputs surpasses some threshold, and “inactive” otherwise. It also should be non-linear, to prevent the entire algorithm from turning into a simple linear function. This generally imitates the mechanism produced by an actual neuron in the brain, which “fires” an electronic signal once the inputs to it surpass some threshold. Generally, either a threshold (step) function or sigmoid (logistic) function is used in computing. With these types of functions, in addition to the inputs from the environment, each unit is provided an additional fixed input equal to negative one. This represents the “bias” applied to the function, and that bias is given its own weight like each of the other inputs. The bias sets the activation threshold value within the activation function, because if the sum of the inputs multiplied by their respective weights exceeds the bias multiplied by its weight, the overall activation provided to the activation function will be positive and therefore the function will generate an “active” signal. If the sum of the inputs multiplied by their weights is less than the bias multiplied by its weight, the overall activation will be negative and the function will generate an “inactive” signal. For the threshold (step) function, the active signal is equal to one, and the inactive signal is equal to zero. For the sigmoid function, the active signal is above 0.5 but less than 1, and the inactive signal is less than 0.5 but greater than 0. The threshold and sigmoid functions are shown in Figure 3.1, and the sigmoid function is shown in Equation 3.2.

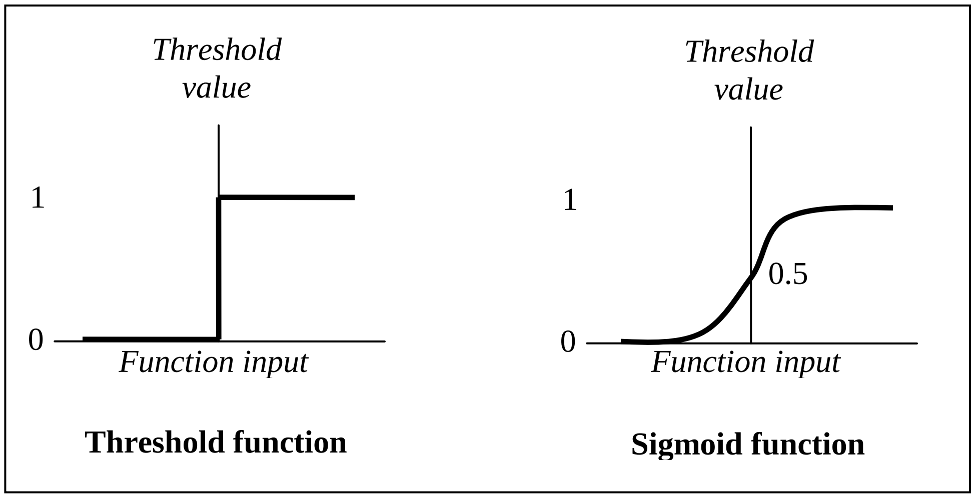


Figure 3.1 – Typical neural network activation functions

$$\frac{1}{(1+e^{-x})} \qquad \text{(Eqn 3.2)}$$

where:

$x = \frac{a}{p}$

$a = Activation$

$p = Sigmoid\ scale\ factor$

Neural networks are generally composed of many units. These units are organized in “layers”, with a layer of input units, a layer of one or more output units, and any number of “hidden layers” containing “hidden units”. The networks are generally categorized as “feed-forward” or “recurrent” networks, depending on whether the network is designed to feed inputs directly into outputs, or feed some of its own outputs back into its own inputs or hidden layers. A basic feed-forward neural network is shown in Figure 3.2 (Russell and Norvig 2003; Buckland 2010).

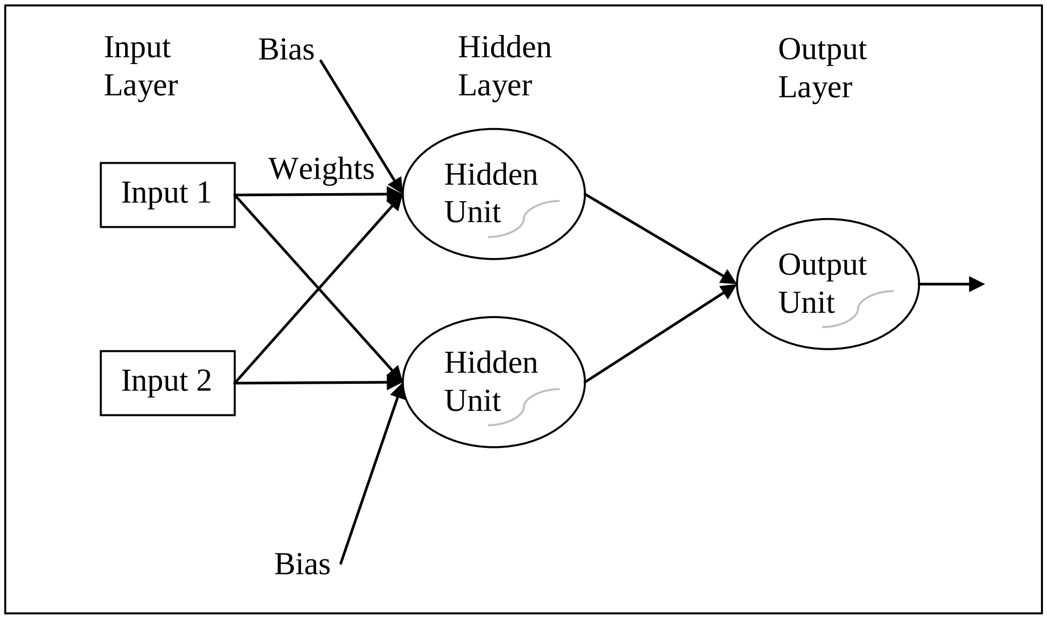


Figure 3.2 – Basic neural network structure

Training of a neural network is a supervised learning procedure, accomplished by adjusting the weights of the network until the outputs most closely match those provided to the network as example data. The training mechanism can be accomplished through a variety of algorithms. One popular mechanism is known as back-propagation, where the network's output error is back-propagated through the network to adjust weights (Bishop 2006). Another popular mechanism is the use of genetic algorithms. Genetic algorithms are generally defined as search mechanisms inspired by the splicing effect of chromosomes in the reproduction of higher level life forms, following the general theory of evolution. For that reason they are sometimes referred to as "evolutionary methods" (Russell and Norvig 2003; Buckland 2010).

Genetic algorithms begin with a “population” of randomly generated solutions to a problem, which are generally symbolized or encoded as strings of numeric symbols. These strings can be binary (0’s and 1’s) or potentially vectors of numbers representing a solution, such as a vector of weights that define a complete neural network. The strings are first evaluated to identify how well they solve the problem at hand. This defines that particular individual solution’s “fitness”. Then pairs of solutions are randomly selected, based on their overall fitness. Individual solutions with a higher fitness are selected with a higher probability than solutions with lower fitness. Each individual solution’s selection probability is directly related to their fitness value. The pairs are then “mated” by randomly selecting a “crossover” point in the string of the solution. The portion of the string that occurs before the crossover point of the first individual in the pair is matched with the portion of the string after the crossover point in the second individual, and vice versa, producing two new solutions or “offspring”. The offspring then replace the original pair in the population. Once the entire population has been subjected to the crossover mechanism, the process is iterated until the best solutions converge to an optimal solution. Additionally, individual solutions are selected at random with a specified frequency for a process known as “mutation”. An individual solution selected for mutation has one or more segments of its solution string replaced by a randomly generated new segment. The effect of this action is to occasionally discover new solutions which may be superior to the existing solutions in the population. Figure 3.3 illustrates the crossover and mutation operations of a genetic algorithm (Russell and Norvig 2003; Buckland 2010).

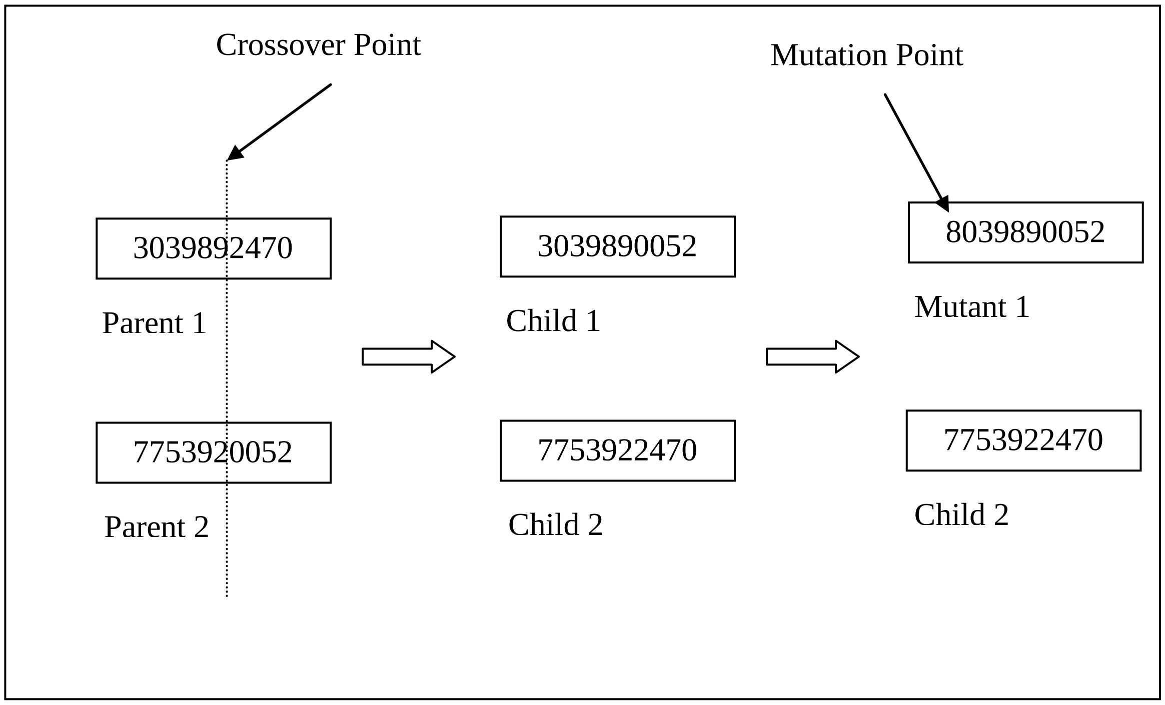


Figure 3.3 – Basic genetic algorithm processes, illustrating the evolution of two potential solution strings

3.1.2.2 – Unsupervised Learning

Unsupervised learning focuses on the set of learning problems for which there is no output data related to the input data. In this category of problems, the primary objective is to characterize properties of the input data. Because there is no output data, the problem no longer involves a "teacher" or "supervisor," and the properties of the data must be determined without being able to quantify the error in estimates made by the learner (Hastie, Tibshirani et al. 2001). Primary applications of unsupervised learning include discovery of groupings of particular data points within a dataset, estimation of the distribution of a dataset, and reduction of the dimension level of a dataset. These applications are known as clustering, density estimation, and visualization respectively (Bishop 2006).

Similar to the supervised case, unsupervised learning methods are also generally derived from methods of statistical analysis. Some unsupervised learning methods include but are not limited to:

- Association rules
- Cluster analysis
- Self-organizing maps

3.1.2.3 – Reinforcement Learning

Reinforcement learning shares some properties of supervised learning in the sense that a dependent output is generated using input data, but also has similarities with unsupervised learning in the sense that the correct answer is not provided to the reinforcement learner. The basis for reinforcement learning is the process of learning by interaction, or trial and error, in order to discover the optimal output given the input data. The primary application of reinforcement learning is to find the best actions to take in any given situation. In this sense, reinforcement learning helps an agent associate an appropriate action to take with the situation it is facing. Reinforcement learning focuses on evaluating actions, rather than instructing an agent on the correct action. In this way, the evaluation illustrates whether one action is better and should be preferred over another. The discovery of optimal actions is based on rewards provided for taking good actions, and likewise, disincentives for taking poor actions. This primary application of reinforcement learning causes it to be more directly linked to the concepts of an artificial intelligence and rational agents interacting with their environment as presented in section

3.1.1 (Sutton and Barto 1998; Bishop 2006). Reinforcement learning has even been suggested to "encompass all of [artificial intelligence]: an agent is placed in an environment and must learn to behave successfully therein" (Russell and Norvig 2003).

Reinforcement learning has a basis in the psychology of animal learning, as well as optimal control (using value functions and dynamic programming) and temporal-difference methods. It focuses on the problem of improving the actions of a goal-based agent, and more specifically a utility-based (decision-theoretic) agent because of inherent connections to decision theory. Reinforcement learning has the ability to deal with uncertain environments, and can be used for both real-time control and planning. The ability to carry out both functions revolves around its focus on optimizing overall value or utility, rather than immediate reward. The discovery of actions that achieve this goal are based on developing a balance between attempting actions that exploit knowledge of immediate rewards, and exploration of non-greedy actions that may lead to higher overall utility (Sutton and Barto 1998; Russell and Norvig 2003).

Unlike supervised and unsupervised learning, reinforcement learning methods are not based strictly around statistical analysis, but rather are methods specific to the field of artificial intelligence that focus primarily on the estimation of utility. These methods include, and are sometimes a blend of:

- Dynamic Programming – a technique to determine the value of being in a state based on complete knowledge of the environment
- Monte Carlo Analysis – a technique to determine the value of being in a state by averaging the overall return learned from experience
- Temporal-Difference – a technique of determining the value of being in a state by leveraging the knowledge of the value of other states and learning from experience

(Sutton and Barto 1998; Russell and Norvig 2003)

3.1.3 – Development of an Intelligent Agent based on Machine Learning

Artificial intelligence is primarily concerned with the development of a rational agent as outlined in Section 3.1.1.1. One of the keys to successful agent development is the creation of the agent function, mapping every situation to a desirable action. Some of the machine learning methods outlined in section 3.1.2 are methods well suited to the development of the agent function, allowing a rational agent to learn the most desirable actions to take in any state by using information and data provided to the agent. In the case of supervised learning, the agent may be provided with optimal actions that should be taken in given states. For reinforcement learning, the agent discovers optimal actions by interaction with the environment. The result of either of these learning methods is a complete specification of actions that should be taken in any state, which is known as a policy. A policy resulting in the best overall outcome for the agent is known as the optimal policy. A policy resulting in the best immediate reward at each action interval is known as the greedy policy. The greedy policy is not necessarily the optimal policy for an agent, since optimal immediate rewards may not always optimize the overall outcome for the agent. Agents sometimes follow a policy that is greedy with respect to the rewards they expect to receive from the environment, but occasionally introducing a non-greedy exploratory action to test alternative policies that might result in a better immediate reward or overall outcome. Selection of those exploratory actions occurs at random, with a frequency related to some small probability usually symbolized as ε. A policy that follows exploratory actions with a frequency of ε is generally called an "ε-

greedy policy”. Two slightly more structured exploratory action policies are known as “softmax action selection” (also known as “Boltzmann exploration”) and the “optimism in the face of uncertainty” method, which both attempt to prevent the agent from exploring an action that might have significantly low reward or poor consequences (Sutton and Barto 1998; Russell and Norvig 2003; Szepesvári 2010).

The environment normally encountered by a rational agent is generally continuous in the sense that the agent’s current state is dependent upon its previous state and the action it selected. If the environment is stochastic, the agent’s current state is related to the previous state through the probability that it would reach this current state given the action it selected. This type of transition from one state to the next is known as a “Markovian transition”. Related to this concept is that of the “Markov property”. The Markov property essentially states that everything about the history of an environment is captured in the current state of the environment, or alternatively, that the response of the environment in the next timestep depends only on the state of the environment currently. The problem of sequentially deciding on actions in this environment, combined with the concepts of utility theory, is known as a “Markov Decision Process” (MDP). The basic components of an MDP are the agent’s initial state, a model of the probabilities of reaching future states given the range of possible actions, and a specification of the reward that will be achieved by reaching a particular state (also known as the reward function). The overall utility of each state is a function of all rewards that are expected to be achieved from the current point forward based on the agent’s policy, which will determine which states are most likely to be reached in the future (Sutton and Barto 1998; Russell and Norvig 2003).

The basic concepts of artificial intelligence, utility theory, and Markov decision processes combine to define all basic elements of a rational decision-theoretic agent based on reinforcement learning, which is the focus of this research. These elements and their symbolic representation are outlined as follows (Sutton and Barto 1998; Russell and Norvig 2003):

- Decision-theoretic agent	agent
- Task environment	environment
- Agent's state	s_t
- Agent's action	a_t
- Policy	$\pi(s)$
- Reward function	$R(s)$
- Value or Utility function	$V(s)$ or $Q(s,a)$
- Model of the environment**	$T(s_t,a, s_{t+1})$

**Also known as the Markovian transition model. In certain types of "model-free" reinforcement learning, the Markovian transition model is not necessary.

One of the elements shown above, the reward function, is of particular interest in studies involving reinforcement learning. The central concept of reinforcement learning is embodied in the reward function, and the development of a successful agent is generally attributed to formulation of an appropriate reward function. The reward function is the primary feedback provided to the agent that indicates the success or failure of a particular action taken in a particular state. The observation and learning from these rewards is the central task of reinforcement learning. The reward function is used to define the goal of the agent, and therefore is not something the agent is usually allowed to modify. Development of a good reward function should focus on what the agent is meant to achieve, rather than the method or path to achievement that is desired. In this way, the agent will more likely seek the final goal rather than finding rewards in repeated accomplishment of subtasks (Sutton and Barto 1998; Russell and Norvig 2003).

The value function (also known as utility function) represents the amount of rewards received by the agent over longer periods of time (also known as the overall return), and allows the agent to attempt to achieve overall goals by maximizing its total reward, rather than focusing on short term immediate rewards. Specifically, the value of a state and/or action is defined by the expected rewards that will be earned in the future as a result of the agent being in that state or taking a particular action from that state. The value for each state is generally determined using the Bellman Equation, calculated as:

$V^{\pi}(s) = \sum_a \pi(s,a) \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^{\pi}(s')]$ (Eqn 3.3)

where:

$V^{\pi}(s) = Value\ of\ state\ s\ under\ policy\ \pi$

$\pi(s,a) = Probability\ of\ taking\ action\ a\ in\ state\ s$

$T(s,a,s') =$

$Probability\ model\ of\ reaching\ state\ s' when\ taking\ action\ a\ from\ state\ s$

$R(s') =$

$Expected\ reward\ earned\ by\ getting\ to\ state\ s'\ from\ state\ s\ using\ action\ a$

$\gamma = Discount\ parameter$

Equation 3.3 above is generally known as a state-value equation, and the function V is known as the state-value function. Similarly, the equation can be defined for the combination of a state and action pair, as:

$Q^{\pi}(s,a) = \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma \sum_{a'} \pi(s',a') Q^{\pi}(s',a')]$ (Eqn 3.4)

where:

$Q^{\pi}(s,a) = Value\ of\ taking\ action\ a\ from\ state\ s\ under\ policy\ \pi$

This form of the equation is generally known as an action-value equation, and the function Q is therefore called an action-value function (Sutton and Barto 1998; Russell and Norvig 2003).

Value functions are actively improved through a process known as an update. This process involves an update to the value for each state or state-action pair using the Bellman update equation, calculated as:

$$V_{i+1}^{\pi}(s) \leftarrow \sum_{a} \pi(s,a) \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V_i^{\pi}(s')] \qquad \text{(Eqn 3.5)}$$

or:

$$Q_{i+1}^{\pi}(s,a) \leftarrow \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma \sum_{a'} \pi(s',a') Q_i^{\pi}(s',a')] \qquad \text{(Eqn 3.6)}$$

The updating process is also known as a backup, and can be represented through a backup diagram as shown in figure 3.4. The backup diagrams illustrate the relationship of a state to its successor states (or state-action pairs).

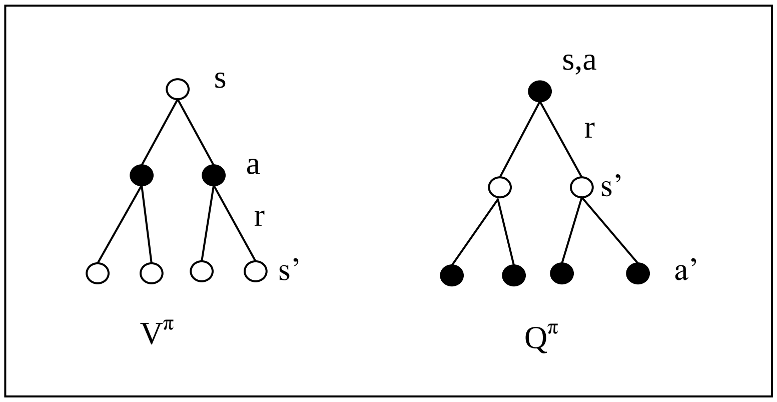


Figure 3.4 – Basic Bellman backup diagrams

Equations 3.5, 3.6, and Figure 3.4 illustrate the Bellman equation and backup diagram for an agent following a single policy. It is possible to discover the optimal policy using those equations through a process known as policy iteration. Policy iteration involves two sub-processes, policy evaluation and policy improvement. The policy evaluation process solves Equation 3.5 or 3.6 through a series of iterations in order to find a value function for all states under the current policy. Once the iterations have converged to a reasonable limit, the policy improvement process then calculates a new policy based on the principle of Maximum Expected Utility by selecting an optimal action for each state that will arrive at the successor state with the maximum value. This is also called a one-step look-ahead. These two processes work iteratively to converge to an overall optimal policy (Sutton and Barto 1998; Russell and Norvig 2003).

An alternative solution for finding the optimal policy is possible by rewriting the Bellman equation to optimize over all policies. This is known as the Bellman optimality equation, written as:

$V^*(s) = \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^\pi(s')]$ (Eqn 3.7)

where:

$V^*(s) = Value\ of\ state\ s\ under\ the\ optimal\ policy$

or:

$Q^*(s,a) = \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma \max_{a'} Q^\pi(s',a')]$ (Eqn 3.8)

where:

$Q^*(s,a) = Value\ of\ action\ a\ taken\ from\ state\ s\ under\ the\ optimal\ policy$

This equation can be turned into an update equation in order to iteratively solve the equation by relating the value of each state or state-action pair to that of its neighbors, calculated as:

$V_{i+1}^*(s) \leftarrow \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V_i^*(s')]$ (Eqn 3.9)

or:

$Q_{i+1}^*(s,a) \leftarrow \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma \max_{a'} Q_i^\pi(s',a')]$ (Eqn 3.10)

The corresponding backup diagrams for these optimal policies are shown in Figure 3.5.

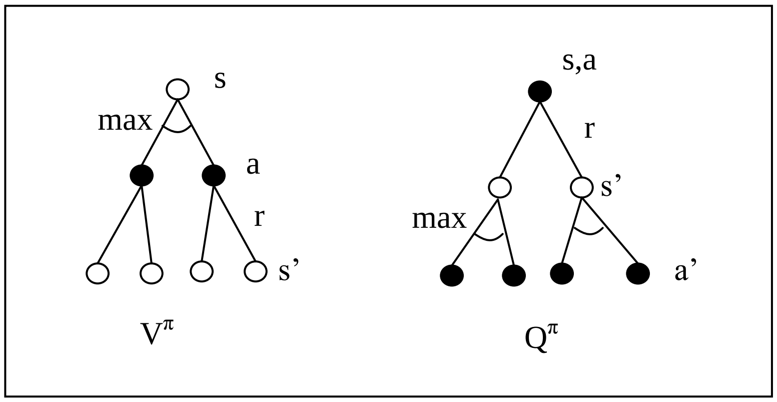


Figure 3.5 – Bellman optimality backup diagrams

It is necessary for the reinforcement learning agent to use iteration to solve the overall value function due to the nonlinear nature of the "max" operator in the second half of the Bellman equations shown in Eqn 3.9 and 3.10. Because of the iterative nature of the process, the correct values for each state or state-action pair are discovered only within a user-specified limit of convergence. This approach is known as value iteration. Value iteration essentially represents a single iteration of policy evaluation and policy improvement as used in the policy iteration approach (Sutton and Barto 1998; Russell and Norvig 2003).

It is possible to use hybrid methods to achieve faster convergence to an optimal policy. Some methods use a number of policy evaluation steps between each policy improvement step. This type of method is known as modified policy iteration. It is also possible to achieve optimality by conducting policy iteration and policy improvement as independent processes, with each gaining information from the other but not necessarily in a strict sequential process as with policy iteration or value iteration. This type of method is known as asynchronous policy iteration. The overall notion of policy evaluation and policy improvement processes interacting in some fashion to find an optimal solution to an MDP is known as generalized policy iteration (GPI) (Sutton and Barto 1998). The various policy iteration approaches are sometimes implemented using methods known as “actor-critic methods”. These methods essentially store the policy and value functions separately. The policy function is used for action selection, forming the “actor” of the pair. Meanwhile, the value function is used as a “critic” to evaluate the actions taken. A variety of actor-critic methods exist, and there can be advantages to implementing GPI using an actor-critic system, including minimizing the computational effort necessary for action selection (Sutton and Barto 1998; Szepesvári 2010).

The methods described thus far are generally known as dynamic programming (DP) solution methods, but they form the basis for solving the reinforcement learning problem. The DP methods described all rely on developing estimates of the value of each state or state action based on estimates of neighboring states, a property known as “bootstrapping.” They suffer from a problem known as the “curse of dimensionality,” whereby the difficulty in solving the equations grows exponentially with the number of possible states, but with the aid of modern computing technology they are often feasible and sometimes one of the best ways to solve an MDP (Sutton and Barto 1998; Russell and Norvig 2003).

In reinforcement learning, it is not necessary for an agent to have perfect knowledge of the environment, or in other words, for the Markovian transition model T to be fully specified in equations 3.3-3.10. This deviation from the classical DP methods described above is generally made by allowing the agent to learn the state or state-action value functions by interactive experience. In one form of this solution method, the agent learns the value function for a state or state-action pair by averaging the future returns experienced each time it encounters the state or takes a certain action from a state. The methods that provide this type of solution are generally called “direct utility estimation” or Monte Carlo (MC) methods, and provide distinct advantages over classical DP methods. As previously noted, this solution does not require a transition model (completely specified model of the environment). The methods can be used with either real environmental experience, or alternatively with simulation or sample models, which are sometimes more readily available than complete transition models. These methods can be focused on a narrower group of states within the entire domain of possible states (state space), providing computational advantages by eliminating the need to completely solve an MDP for the entire state space. Finally, these methods calculate state or state-action values by averaging actual future returns, and therefore do not rely on bootstrapping as with classical DP methods. This provides the methods with some flexibility in dealing with problems where adherence to the Markov property is not necessarily strictly enforced (Sutton and Barto 1998; Russell and Norvig 2003).

Generally, MC methods require that there be some sort of end state that is reached, making the learning environment episodic. In this way, an agent may follow a policy π, and once the end of an episode is reached, the observed returns can be credited to each state (or state-action pair) the agent passed through. By doing this many times, the average returns will eventually converge to the expected value for the state (or state-action pair), carrying out the policy evaluation step of GPI. By maintaining a certain amount of exploration of non-greedy actions, MC methods can also carry out the policy improvement step, completing the requirements for GPI and eventually converging to an optimal policy. It is also possible to use MC methods to gradually evaluate the usefulness of a policy that is not being followed by an agent, thereby actively searching for potentially optimal policies while following a reasonable policy. This is known as an "off-policy" method, whereby the agent keeps track of the values of states-action pairs where an occasional non-greedy action was taken, while generally following a greedy policy. A simple MC update equation is calculated as:

$$V(s_t) \leftarrow V(s_t) + \alpha[R_t - V(s_t)] \qquad \text{(Eqn 3.11)}$$

where:

$V(s_t) = Value\ of\ state\ s\ at\ time\ t$

$R_t = Actual\ return\ observed\ after\ time\ t$

$\alpha = Step\ size\ parameter$

The backup diagram for on-policy MC methods is shown in Figure 3.6.

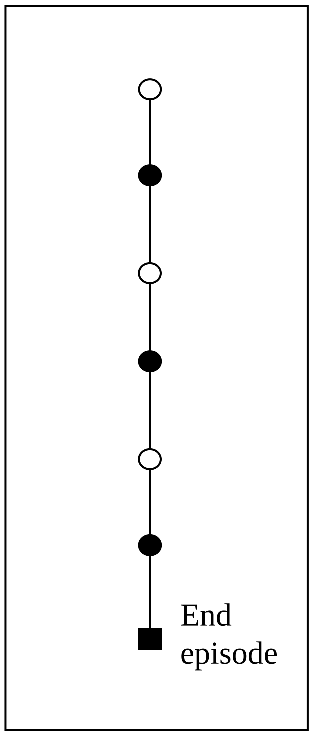


Figure 3.6 – Monte Carlo backup diagram

Another set of reinforcement learning methods that employ interactive experience with the environment are known as Temporal Difference (TD) learning methods. TD learning is similar to MC learning because examples from the environment or a simulation model are used to estimate the value of states or state-action pairs, but the requirement to reach the end of an episode before updating the value is relaxed by allowing the update to occur based on the value of the following state. In this sense, TD learning uses bootstrapping in a manner similar to DP methods, but maintains the benefits of learning from experience as with MC methods. The update equation for a simple one-step TD method is calculated as:

$$V(s_t) \leftarrow V(s_t) + \alpha[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)] \qquad \text{(Eqn 3.12)}$$

where:

$r_{t+1} = Value\ of\ immediate\ return\ at\ time\ t + 1$

The TD equation shown above can be converted to an action-value function by simply replacing the state-value terms with action-value terms, as:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \qquad \text{(Eqn 3.13)}$$

The basic backup diagram for a one-step TD algorithm is shown in Figure 3.7.

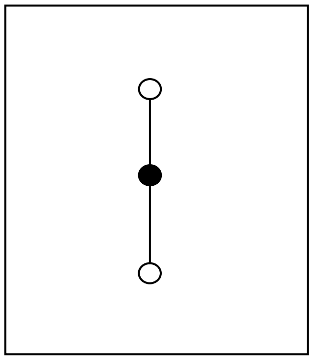


Figure 3.7 – TD(0) backup diagram

Like MC learning, TD methods have an advantage over classical DP methods because they do not require a complete transition model of the environment, can work with the actual environment or simulation models, and do not require a full solution for the entire state space. On the other hand, the fact that TD methods use bootstrapping and therefore do not require completion of a full episodic task sometimes gives these methods an advantage over MC methods, particularly on tasks with longer episodes. Both TD and MC methods have been shown to converge to an optimal solution, and thus depending on the learning problem being addressed, one or the other can prove to be the more efficient solution (Sutton and Barto 1998; Russell and Norvig 2003).

Like MC and DP methods, TD methods can also be used to improve agent control with the GPI paradigm, first evaluating the state-value or action-value functions, then improving the agent's policy by making it greedy with respect to those value functions. As with MC methods, TD learning can be used while following a specific policy (on-policy) or alternatively, based on an off-policy algorithm. A widely used on-policy method is known as "SARSA." This is essentially a 1-step DP method. The name stems from the sequence of events in each of the agent's transitions from one state-action pair to another (state, action, return, state, action). The basic update equation for this method is provided in Equation 3.13. Alternatively, a popular off-policy method is known as "Q-learning," in which the agent learns the optimal action-value function or "Q function" as it progresses through the state space, regardless of the policy the agent is following. This process is effectively carried out by evaluating each state-action pair's value (through bootstrapping) based on the value of the proceeding state-action pair that would be encountered if an optimal action was chosen, even if the agent's current policy dictates that it will not currently follow that action. The one-step Q-learning update equation is calculated as:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] \qquad \text{(Eqn 3.14)}$$

The Q-learning backup diagram is shown in Figure 3.8. The arc across the three possible actions represents the "max" function, selecting the action that results in reaching the next state-action pair with the highest value.

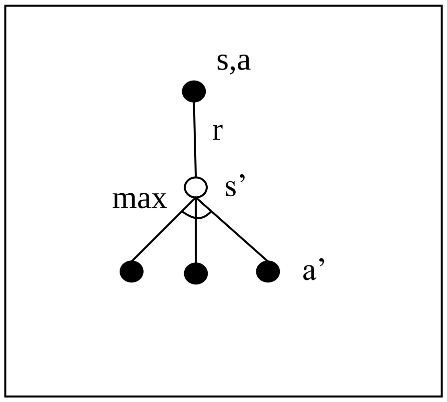


Figure 3.8 – Q-learning backup diagram

MC and TD methods share the ability to learn from interactive experience, but differ in their use of sample returns and bootstrapping. As presented thus far, MC methods must wait to process an update until the end of an episode based on all future returns, whereas TD methods update values on each timestep using bootstrapping. It is possible to utilize a hybrid approach by using actual returns from several future timesteps as provided for in MC methods, but then applying bootstrapping to the subsequent timestep in order to gain information on returns from that point onward. This approach is known as an n-Step TD method, with the update equation calculated as:

$$V_t(s_t) \leftarrow V_t(s_t) + \alpha\left[R_t^{(n)} - V_t(s_t)\right] \qquad \text{(Eqn 3.15)}$$

where:

$$R_t^{(n)} = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} \dots + \gamma^{n-1} r_{t+n} + \gamma^n V_t(s_{t+n})$$

It is further possible to hybridize the n-Step TD methods by obtaining an update that is a weighted average of several different n-Step TD equations. As long as all of the weights sum to 1, any different combination of n-Step TD equations can be used to update a value. One particular algorithm for achieving this hybridization is known as the TD(λ) algorithm. This method weights each n-Step return by λ^{n-1} (with $0 \le \lambda \le 1$), and normalizes the entire return with a factor of 1 - λ (to ensure all weights sum to 1) in order to obtain an average of each n-Step return with a decaying weight for increasing n (Sutton and Barto 1998). The resulting update equation is calculated as:

$$V_t(s_t) \leftarrow V_t(s_t) + \alpha\left[R_t^\lambda - V_t(s_t)\right] \qquad \text{(Eqn 3.16)}$$

where:

$$R_t^\lambda = \sum_{n=1}^{T-t-1} \lambda^{n-1} R_t^{(n)} + \lambda^{T-t-1} R_t$$

The TD(λ) algorithm provides a simple mechanism to bridge the entire spectrum of possibilities from a decaying MC method by using λ = 1, to the classic one-step TD algorithm described previously by using λ = 0. The algorithm also gives rise to an implementation mechanism known as the "eligibility trace". An eligibility trace is a parameter that defines which recently visited states are eligible to have their values updated, and the weight of the update. The eligibility trace is calculated as:

$$e_t(s) = \begin{cases} \gamma\lambda e_{t-1}(s) & if\ s \neq s_t \\ \gamma\lambda e_{t-1}(s) + 1 & if\ s = s_t \end{cases} \qquad \text{(Eqn 3.17)}$$

The eligibility trace is then used in the update equation, calculated as:

$$V_t(s_t) = V_t(s_t) + \alpha e_t(s)[r_{t+1} + \gamma V_t(s_{t+1}) - V_t(s_t)] \qquad \text{(Eqn 3.18)}$$

As calculated above, each time a state is visited, its eligibility trace is increased by a value of 1 and subsequently fades away with the amount of time that the state is not visited. A slight variation of this algorithm resets the trace to a value of 1 rather than increasing the value, thereby preventing the trace from becoming too strong. This variation is known as a “replacing trace” method, whereas the former is known as an “accumulating trace” method. The use of eligibility traces provides a more efficient mechanism for implementing the TD(λ) algorithm, and thus both the classical one-step TD method as well as the complete MC method. Due to the fact that eligibility traces create a sort of “blend” between TD and MC methods, they provide many of the benefits from both methods. They are most beneficial in tasks that violate the Markov property or have long delays before a substantial reward is earned, and can often provide a faster learning mechanism (Sutton and Barto 1998).

The eligibility trace concept can also be extended to include agent control under GPI. In the most straightforward case, eligibility traces can be applied to the SARSA algorithm by simply switching Equation 3.18 to apply to state-action pairs instead of just the state-value function. The rest of the GPI algorithm is then applied as previously described. Eligibility traces can also be applied to Q-learning through several mechanisms. These mechanisms generally focus on resetting all eligibility traces to 0 whenever a non-greedy action is selected, or alternatively, using the non-greedy selection as the final “nth-step” return (Sutton and Barto 1998).

All of the reinforcement learning methods and algorithms discussed to this point focus on generating values for discrete states or state-action pairs. This can become computationally inefficient in the case of large state spaces, potentially making the solution for these cases intractable. In these cases it is useful to generalize the values for all states or state-action pairs by approximating with a continuous function, creating an actual value function rather than tablized values. The creation of a value function is a supervised learning task, using each update to a value as an added point to the training set supplied to the supervised learning algorithm. Most of the common supervised learning methods will work for this task, with gradient-descent methods applied to artificial neural networks or linear regression being two popular alternatives. Agent control under GPI has been shown to work reasonably well with function approximation methods, using similar techniques to those already developed. There are some known issues with the use of this type of system, and research in this arena is ongoing (Sutton and Barto 1998).

As outlined previously, DP methods generally require a model of the environment, whereas MC and TD methods learn more directly from experience. Model-based methods like DP are generally referred to as planning methods, or indirect reinforcement learning methods, due to the fact that they are able to plan independently of gaining new experience and do not directly learn the utility of states from experience. In these methods, learning can be applied directly to the model of the environment which is then used to plan or update utility information. One example of this using classic DP methods is known as adaptive dynamic programming (ADP). Methods not based on models like MC and TD are more commonly known as learning methods, or direct reinforcement learning methods. They learn utility information and policy directly from experience, but do not independently plan in the absence of new experience. It is possible to combine the use of both learning and planning methods into a single agent, directly using experience for update and action, while learning a model of the environment and using that model for planning purposes simultaneously to improve agent performance. There also exist different planning strategies and backup strategies to improve performance of both the direct and indirect reinforcement learning methods used in such a hybridized agent. Generally speaking, combinations and hybridizations are possible with most of the different methods and algorithms presented thus far, providing the ability to improve agent performance for a particular task environment (Sutton and Barto 1998; Russell and Norvig 2003).

3.2 – Development of an Intelligent Agent for Water Quality Management

3.2.1 – Background and Assumptions

The concepts discussed in Section 3.1 of this document were used to develop an intelligent agent structure to address issues related to water quality management. The development of that agent is detailed in this section, and application of that agent structure to the case study on the Truckee River is covered in the proceeding chapter. For the development of this intelligent agent, it is assumed that the agent will have the ability to store and release some limited amount of water that can be dedicated to instream flow. The agent’s primary focus will be on the management of this specific instream flow water within the context of the broader operation of the river and reservoir system, with the agent taking into consideration the effects of other reservoir operations for other objectives (water supply, hydropower, etc.) on water quality as well. Because the water controlled by the agent will be used to augment the water released for other objectives, it is assumed that flow augmentation does have an effect on the water quality issues of interest to the agent. The agent is designed to consider one or more water quality objectives at one or more points of interest within the river downstream of the reservoir or reservoirs being considered.

3.2.2 – Agent Specification

As previously discussed, the agent receives state and reward information from its environment, and acts on that information based on an agent function that is implemented through an agent program on the agent's architecture. For the development of the water quality rational agent, the state information from the environment is limited to inputs on any given timestep which define the state of the environment that the agent should be concerned with, depending on the overall goals of the agent. For an agent concerned with the immediate improvement of water quality in general, or improving water quality once the water quality has exceeded a preferred threshold for the river, the state inputs include the amount of dedicated water in storage in reservoirs that the agent currently has available to utilize for improvement of water quality, and the predicted water quality at some downstream control point for the current timestep if no additional releases are made by the agent (in other words, the downstream water quality based on releases made only for other objectives such as water supply, hydropower, etc.). For an agent concerned with not only the general improvement of water quality, but also avoiding exceeding a certain threshold for a certain period of time, the state inputs would include those outlined above (storage of dedicated water and predicted downstream water quality), and also a signal to indicate that the threshold will have been exceeded for more than the desired length of time unless some action is taken. For an agent concerned with longer-term planning for the general improvement of water quality over a number of years, the state inputs would not necessarily include information about the immediate state of downstream water quality, but rather would include more general information about the overall state of the river and reservoir system, and would generally only be provided to the agent on an annual or seasonal basis. That information would include the amount of dedicated water the agent has to use for a season, and some indicator of the likely flow augmentation needs for the season.

The agent is also provided the reward calculation from the action taken by the agent in the previous timestep, so that it may update its knowledge of the usefulness of that previous action. In the case of an agent focused on the longer-term outcome, the reward would be calculated and provided to the agent on a seasonal or annual basis. The action taken by the agent on the environment is a direction to the environment regarding the amount of additional water to release into the river for water quality augmentation on top of the water already released for other purposes. Alternatively, for an agent with a longer-term focus, the action would be the specification of the overall daily release policy to be used for the entire upcoming season or year. For the water quality agent, the most straightforward implementation is a reward function that is inversely proportional to the quality of the action taken by the agent. In other words, this is a reward function that focuses on punishment values for not meeting the goals, and one which should be minimized to achieve improved actions. This type of implementation provides for the development of a reward function that increases with increasing levels of violation of water quality standards or goals.

The agent's policy function (agent function or agent program) is a mapping of the current state of the system to its next action, where the current state of the system from the point of view of the agent is defined as the current amount of dedicated water for water quality improvement available in storage, and depending on the agent's focus, some combination of the predicted water quality in the river downstream assuming no additional water released by the agent, the threshold exceedence signals, and/or some indication of the flow augmentation needs in the near future. Two types of agent policy functions were developed as part of the system implementation. One agent policy function is a tabular implementation, providing a direct lookup mechanism for the agent to select a release or policy selection action based on current dedicated water available in storage, predicted water quality at some downstream point, threshold exceedence signals, and/or flow augmentation requirement indicators. The second is a neural network which allows the agent to calculate the approximate action that should be taken given the current state of the environment. The user has the ability to select which type of agent policy function is to be used for a particular run.

The tabular case can be used to verify proper operation of the environment simulation model, and proper data connections between the agent and the environment simulation model. It is also used to evaluate operation of the reward signal being specified from the environment model, and to evaluate the usefulness of the reward function in providing information to the agent that will aid in the learning and discovery of improved action policies. The tabular case provides an excellent tool for testing and debugging of the system, since exact values and calculations are used for each computational operation of the agent. In addition, its use is efficient and appropriate when the state and action space are small to moderately sized, and can be reasonably discretized into meaningful partitions, also called “bins” or “tiles”.

The neural network case can be used to aid in the generalization of the agent policy function to deal with all possible specifications of the environment. It allows the state and action signals to essentially be represented in a continuous manner, rather than requiring discretization of the state variables. In addition, it can be a tool to mitigate against the “curse of dimensionality” (Sutton and Barto 1998) associated with the use of the tabular case by serving as an interpolator of the complex agent policy function. This dimensionality problem arises from the fact that as the number of possible states or actions available to the agent increase, the memory requirements and computing intensity rapidly increase in a multiplicative manner, creating performance issues for the agent. As the number of state variables increases, the problem becomes exponential and sometimes causes problems to become intractable and practically unsolvable using conventional computing technology. The neural network case provides the flexibility to deal with the potential for an increased number of state variables and potential actions without the full impact of the dimensionality problem. This allows the system to adapt to a larger number of potential water quality and reservoir system state variables, making the system flexible to deal with different water quality and river/reservoir simulation models or real-time water quality and river/reservoir system inputs. A diagram of the overall water quality agent is shown in Figure 3.9.

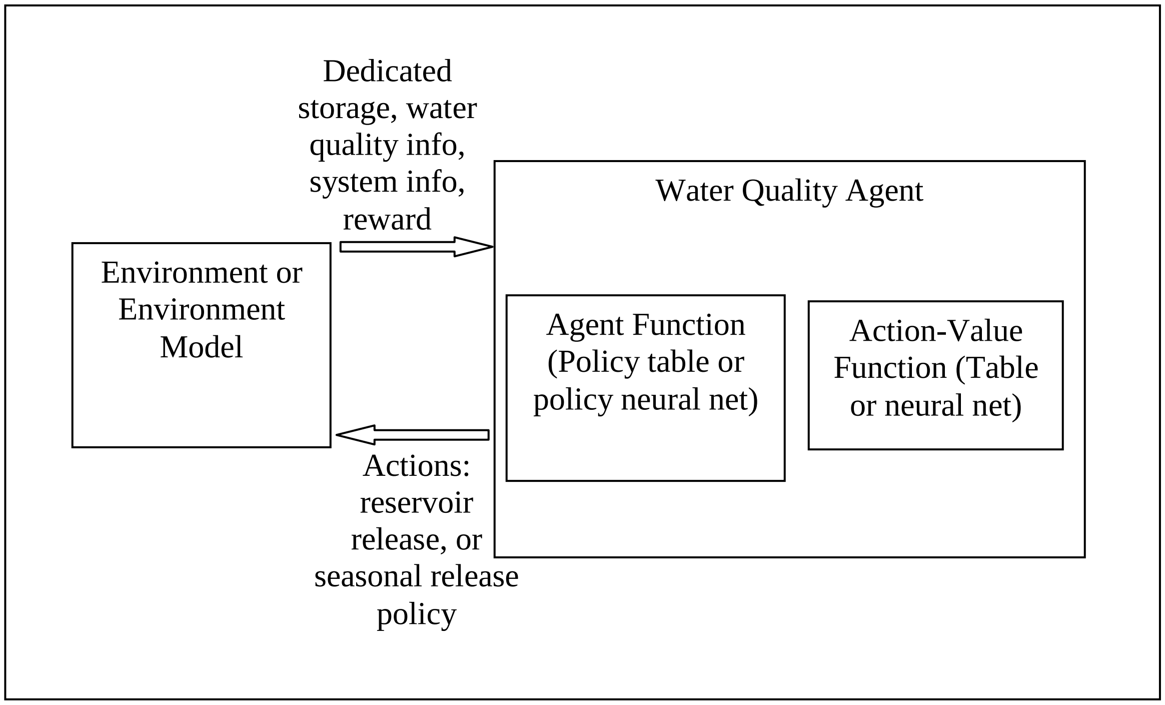

Dedicated storage, water quality info, system info, reward
Water Quality Agent
Environment or Environment Model
Agent Function (Policy table or policy neural net)
Action-Value Function (Table or neural net)
Actions: reservoir release, or seasonal release policy

Figure 3.9 – Diagram of the water quality intelligent agent

3.2.3 – Learning Methods

The water quality agent was designed as a decision-theoretic agent as previously defined. It operates based on the principle of Maximum Expected Utility, attempting to make rational decisions based on its knowledge of the expected rewards and overall utility of the state it is in and the actions it takes. This is due to the addition of a learning element based on the concept of reinforcement learning. Using this concept and methodology, the agent takes into account the tradeoff between immediate reward and future reward, and attempts to take actions that will produce the best overall results considering the impacts on long term reward when attempting to maximize immediate reward.

The water quality agent was designed to provide several learning method options to a user. The agent can utilize the one step TD method known as SARSA, as discussed in detail in Section 3.1.3, working with an action-value function as calculated by Equation 3.13. The agent was constructed for the control case using GPI as described in Section 3.1.3. Alternatively, the agent can utilize the one step TD method known as Q-learning, as discussed in detail in Section 3.1.3, working with an action-value function as calculated in Equation 3.14, using GPI as well. The agent was also designed to optionally expand to the n-Step versions of both SARSA and Q-learning through the use of eligibility traces as shown in Equations 3.17 and 3.18, substituting the values for state-action pairs instead of the state-value. Options were implemented to allow for either the "replacing trace" or the "accumulating trace" methods of the eligibility trace concept. For the Q-learning version of the eligibility trace mechanism, an algorithm known as "Watkin's Q(λ)" was utilized (Sutton and Barto 1998). Using this algorithm, the eligibility traces are calculated as previously discussed, but are set to zero whenever a non-greedy action is taken.

The policy improvement process of GPI for the agent was designed to provide improvements at every policy evaluation step, as followed in a value iteration approach. Alternatively, the policy improvement process may occur at user-specified intervals, allowing the user to adjust the interval, as generally followed in a modified policy iteration scheme (Russell and Norvig 2003). Additionally, the user may elect to complete policy improvement after the action-value function has stabilized using the current policy. To do this, the user may specify that the policy improvement will only occur when all recent policy evaluation calculations have only changed the action-value function within a certain threshold percentage. In general, the agent's design is based on the concept of asynchronous policy iteration, allowing for the selection of any specific type of desired policy iteration.

The system is designed to allow adjustment of the step size parameter and discount rate for the update equation as presented in Equation 3.13. In addition, options were introduced to allow either a constant step size parameter, or one that reduces with increased experience. A variety of reduction algorithms have been evaluated in the literature, although none were identified as the best for implementation (Sutton and Barto 1998; Szepesvári 2010). For the water quality agent, a similar equation was used to the generalized equation noted in Szepesvari (Szepesvári 2010) to provide for a reducing step size parameter. That equation reduces the step size as follows:

$$\alpha_{s,a} = \frac{1}{t^{\eta}} \qquad \text{(Eqn 3.18)}$$

where:

$t = Number\ of\ times\ action\ a\ taken\ in\ state\ s$

$\eta = Reduction\ parameter$

For any choice of reduction parameter greater than zero, Equation 3.18 results in a step size parameter that asymptotically approaches zero. A higher choice of reduction parameter results in a more rapid decrease in step size, effectively placing a higher weight on earlier training experiences in the development of the action-value function. The reduced step size parameter is specific to each state and action, to ensure that significant updates are appropriately applied to states and actions that are not frequently experienced. The effects of the use of different reduction parameter values are shown in Figure 3.10.

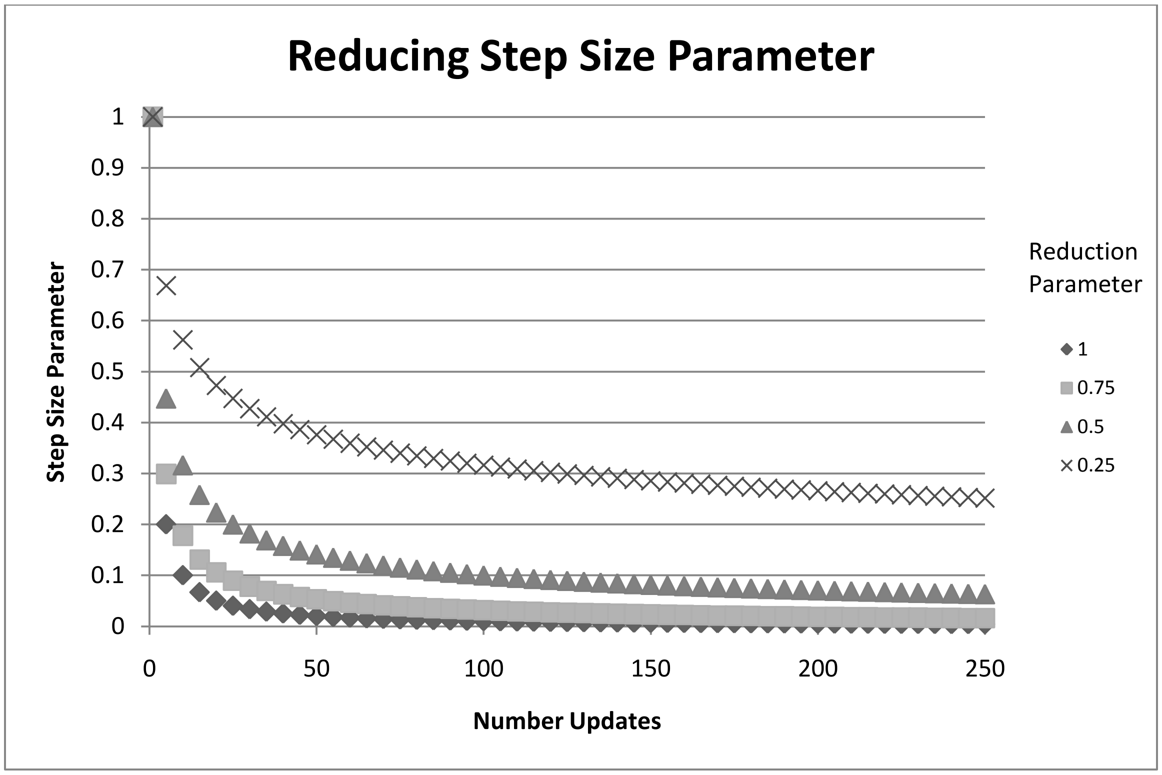


Figure 3.10 – Reducing step size parameter options

3.2.4 – Exploration Methods

The water quality agent was designed to take advantage of several different methods to balance the need for exploration during the development of the action-value function and agent's policy function. These methods include the ε-greedy, softmax, and optimism in the face of uncertainty (OFU) methods, as discussed in Section 3.1.3 (Sutton and Barto 1998; Russell and Norvig 2003; Szepesvári 2010).

The ε-greedy method generally follows the current policy of the agent, selecting an exploratory action with the probability of ε, where $0 \leq \varepsilon \leq 1$. For the case of a relatively stationary environment, the water quality agent design also allows for a gradually reducing value of ε, which provides for more stable agent behavior after the agent has sufficiently explored the state and action space (Sutton and Barto 1998; Szepesvári 2010). The reduction of the probability of an exploratory action is calculated as:

$$\varepsilon_s = \varepsilon_s \times \left(1 - \frac{\zeta}{1000}\right)^{t_s} \qquad \text{(Eqn 3.19)}$$

where:

$t_s = Number\ of\ times\ state\ s\ has\ been\ experienced$

$\zeta = Reduction\ parameter$

For higher choices of the reduction parameter, the likelihood of exploration decays more rapidly, asymptotically approaching zero. The reduced likelihood of exploration is specific to each state, ensuring adequate exploration actions are taken in states that are rarely experienced. The reduction in the probability of exploration associated with various selections of the reduction parameter ζ is shown in Figure 3.11.

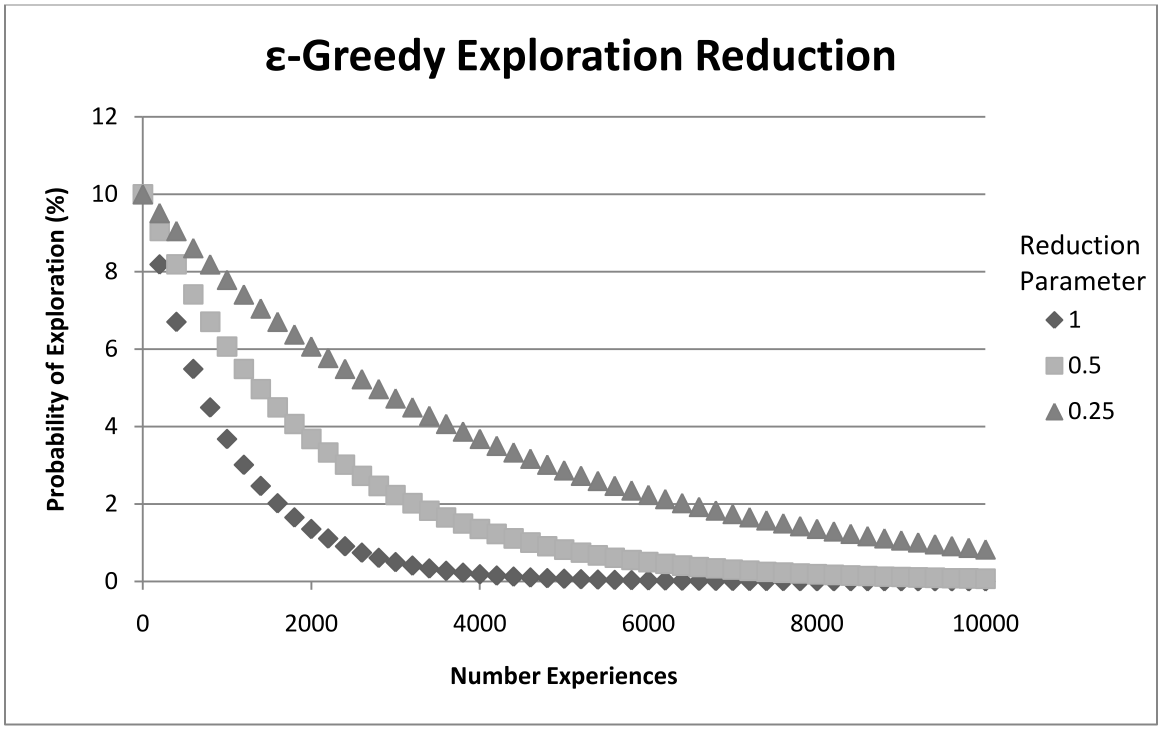


Figure 3.11 – Reduction in the probability of exploration associated with various reduction parameter values, beginning at an exploration rate ε = 10%

The softmax (or Boltzmann) method of exploration is actually an action selection method, technically superseding the need for an explicit agent function or policy during training. However, the water quality agent still produces an agent policy, which is used for testing agent behavior, and for agent operation in the actual environment. The softmax method selects actions based on their action-value estimates. In this manner, less desirable actions are generally selected less often than superior actions. Using the softmax method, actions are selected with the probability:

$$\frac{e^{\frac{Q(s,a)}{\tau}}}{\sum_{b=1}^{n} e^{\frac{Q(s,b)}{\tau}}} \qquad \text{(Eqn 3.20)}$$

where:

$n = Number\ of\ possible\ actions\ in\ each\ state\ s$

$\tau = Temperature\ parameter$

The degree to which less desirable actions are selected is determined through the value selected for the "temperature parameter". As τ approaches zero, less desirable actions are selected less often until only the most desirable action is selected, representing the final agent policy function. Because of this, τ can be reduced with increased agent experience to prevent the agent from taking actions that are found to be undesirable through experience. The water quality agent provides for the reduction of τ in a similar manner to the reduction of ε in the ε-greedy method, as:

$$\tau_s = \tau_s \times \left(1 - \frac{\zeta}{1000}\right)^{t_s} \qquad \text{(Eqn 3.21)}$$

where:

$t_s = Number\ of\ times\ state\ s\ has\ been\ experienced$

$\zeta = Reduction\ parameter$

The reduction in the temperature parameter τ associated with various selections of the reduction parameter ζ is shown in Figure 3.12.

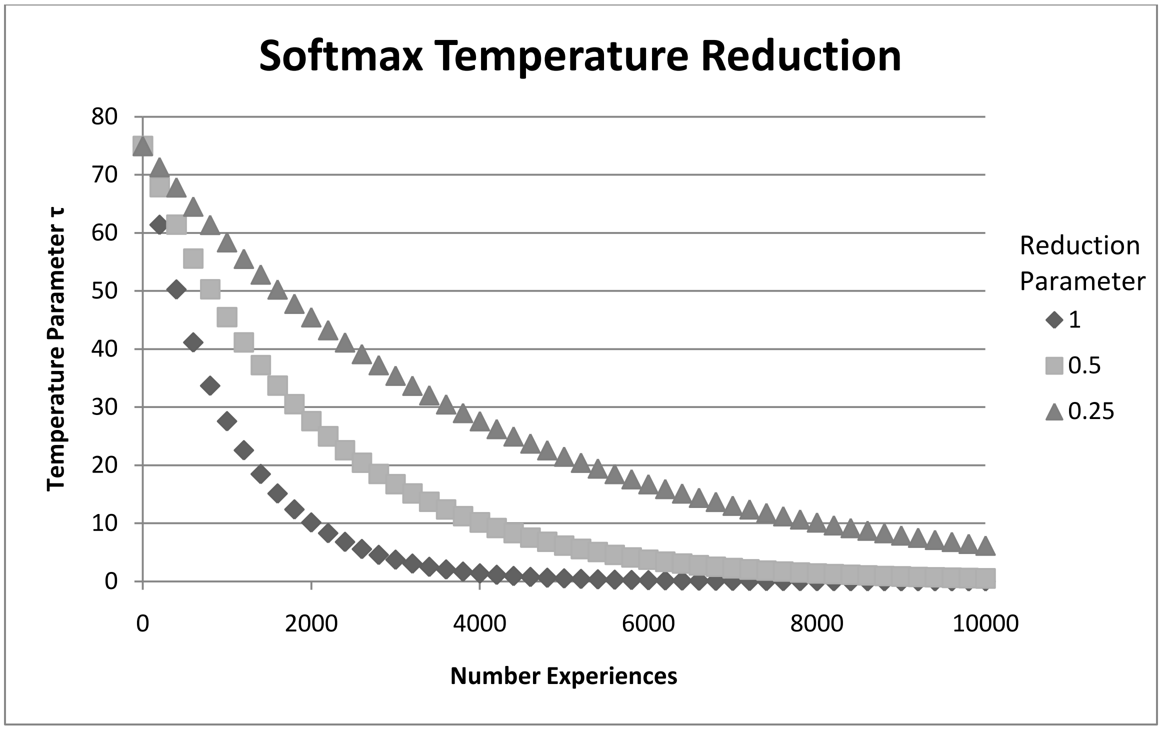


Figure 3.12 – Reduction in the temperature parameter τ associated with various reduction parameter values, beginning at a temperature τ = 75

Through experimentation with the case study summarized in the proceeding chapter, it was found that a reasonable initial value for the temperature parameter τ is approximately one half of the maximum expected action-value. An estimate for the maximum expected action-value is developed later in this section, as shown below in Equation 3.28. Based on that selection mechanism for an initial value, the progression of action selection under a softmax scheme is shown in Figure 3.13 for actions taken in a particular state with a maximum expected action-value of 150. The figure shows that as the temperature parameter τ is reduced with increased experience, the probability of selection of the action with the best action-value is increased, reaching 100 percent in the limit, while the probabilities of selecting actions with poorer action-values decreases with increased experience. For this example, the release action associated with the best action-value is 175 cfs.

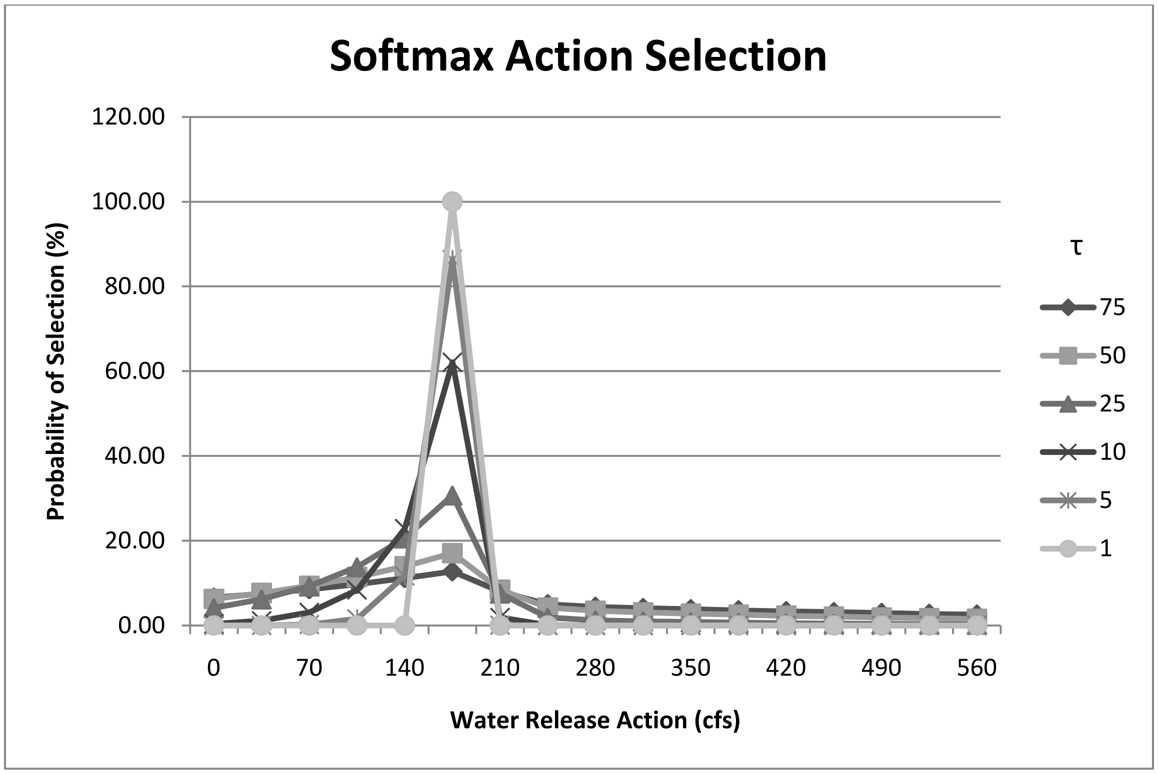


Figure 3.13 – Example of the progression of action selection probabilities with decreasing temperature parameter values

Similar to the softmax method, the OFU method represents an action selection policy. Also in a similar manner, OFU attempts to avoid undesirable actions. Some of the primary differences between the two methods are that OFU does not require a user-specified temperature parameter, and OFU focuses its attention on the mean experienced reward of an action rather than the estimated action-value. In addition, OFU takes into consideration the amount of experience the agent has with particular actions, providing a higher likelihood of action selection for actions that have not been tested as much as other actions. OFU generally selects actions based on the best upper confidence bound in the reward estimate. Since the water quality agent is designed to prefer actions with lower rewards, OFU implementation for this agent focuses on the best lower confidence bound. The action with the lowest confidence bound on any given timestep becomes the selected action. The confidence bound for each action is calculated as (Szepesvári 2010):

$$r(s,a) - R_s\sqrt{\frac{2\times\log(t_s)}{t_{s,a}}} \qquad \text{(Eqn 3.22)}$$

where:

$t_s = Number\ of\ times\ state\ s\ experienced$

$t_{s,a} = Number\ of\ times\ action\ a\ taken\ in\ state\ s$

$r(s,a) = Mean\ experienced\ rewards\ for\ action\ a\ taken\ in\ state\ s$

$R_s = Range\ of\ rewards\ experienced\ in\ state\ s\ (full\ range\ is\ {}^{+}/{-}\ R)$

For the water quality agent, the domain of possible rewards lies in the bounds from zero to the maximum expected reward, which can be specified by the user a priori. Using this approach, Equation 3.22 becomes:

$$\left(r(s,a) - \frac{\max r}{2}\right) - \frac{\max r}{2}\sqrt{\frac{2\times\log(t_s)}{t_{s,a}}} \qquad \text{(Eqn 3.23)}$$

where:

$\max r = Maximum\ expected\ reward$

In order to prevent the need to estimate the maximum expected reward *a priori* and to provide an added level of protection against the selection of highly undesirable actions, an alternative option provided by the water quality agent is to replace range R_s in Equation 3.22 with the actual difference between the maximum and minimum experienced reward for each state. This is similar in concept to methods discussed in Szepesvari (Szepesvári 2010). For the water quality agent, this can be beneficial by providing added protection against the agent attempting to release large quantities of stored water through exploratory actions when it is known that those releases will likely be of minimal value to the health of the river downstream. Using this approach, Equation 3.22 becomes:

$$\left(r(s,a) - \frac{\max r_s - \min r_s}{2}\right) - \left(\max r_s - \frac{\max r_s - \min r_s}{2}\right)\sqrt{\frac{2 \times \log\,(t_s)}{t_{s,a}}} \qquad \text{(Eqn 3.24)}$$

where:

$\max r_s = Maximum\ experienced\ reward\ in\ state\ s$

$\min r_s = Minimum\ experienced\ reward\ in\ state\ s$

Through experimentation on the case study described in the proceeding chapter, it was noted that the OFU methods worked well when the discount rate in the Bellman update equations (Equations 3.13 and 3.14) was low and the action-value estimate for any particular state-action pair was similar to the mean experienced rewards as used in the confidence bound equations used in OFU action selection (Equation 3.22 through 3.24). However, the OFU action selection method produced less desirable action selections at higher discount rates, due to the relative disconnect between the mean immediate experienced reward and the overall action-value estimate. For this reason, the water quality agent provides the option to use the action-value estimate in place of the mean experienced rewards in Equations 3.22 through 3.24, resulting in the following confidence bound equations:

$$q(s,a) - Q_s\sqrt{\frac{2 \times \log\,(t_s)}{t_{s,a}}} \qquad \text{(Eqn 3.25)}$$

$$\left(q(s,a) - \frac{\max q}{2}\right) - \frac{\max q}{2}\sqrt{\frac{2 \times \log\,(t_s)}{t_{s,a}}} \qquad \text{(Eqn 3.26)}$$

$$\left(q(s,a) - \frac{\max q_s - \min q_s}{2}\right) - \left(\max q_s - \frac{\max q_s - \min q_s}{2}\right)\sqrt{\frac{2 \times \log\,(t_s)}{t_{s,a}}} \qquad \text{(Eqn 3.27)}$$

where:

$q(s,a) = State - action\ value\ estimate\ for\ action\ a\ taken\ in\ state\ s$

$Q_s = Range\ of\ state - action\ values\ in\ state\ s\ (full\ range\ is\ ^{+}/- Q)$

$\max q = Maximum\ expected\ state - action\ value$

$\max q_s = Maximum\ state - action\ value\ in\ state\ s$

$\min q_s = Minimum\ state - action\ value\ in\ state\ s$

The maximum expected action-value found in the above equation can be fairly easily estimated with basic knowledge of the reward structure for the agent, and an estimate of the maximum expected reward. If the maximum expected reward can be estimated, then the maximum expected action-value can be calculated using the discount rate as:

$$\max q = \frac{\max r}{1-\gamma} \qquad \text{(Eqn 3.28)}$$

This calculation is derived by taking the limit of the Bellman update equation as t approaches infinity, as:

$$\lim_{t\to\infty} Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)] \qquad \text{(Eqn 3.29)}$$

In Equation 3.29, the Q values for the current and successive timesteps become equal, and Equation 3.28 is derived through algebraic solution of the update equation, assuming max r in the place of r_{t+1}. It should be noted that the estimate of the maximum action-value in Equation 3.28 does not apply if eligibility traces are in use.

3.2.5 – Physical Implementation

The basic functionality of the water quality agent is carried out by a number of different standalone computer applications designed to independently use a centralized set of data files. These various applications include environment model interface programs, database update programs, and neural network training programs. The programs were created within the Microsoft Visual Studio 2010 integrated development environment, using the Microsoft Visual Basic .Net programming language, and operate in a Microsoft Windows-based personal computing environment. The system of programs generally forms an actor-critic type implementation as described in Section 3.1.3.

The primary application that represents the bulk of the functionality of the water quality agent is a program called the “environment model interface” or “interface program”. The interface program acts on the model of the environment by providing the model with an action selection at each timestep based on the current state of the environment, and then conducting the policy evaluation step of GPI based on the reward signal received from the environment for the previous timestep’s action. For the action selection step, the interface program obtains the current state information from the environment model, selects an action based on either the ε-greedy, softmax, or OFU exploration methods, and provides the action back to the environment model. For the purposes of keeping track of the number of times that a particular action has been selected in a particular state and the mean experienced rewards for OFU calculations, the interface program then updates a tracking file located in the centralized file space. The interface program then conducts the policy evaluation step of GPI by calculating the action-value update in either Equation 3.13 or Equation 3.14, depending on whether the SARSA or Q-Learning methodology is utilized. The update value is then written to an update file in the centralized data file space, along with pertinent data required for the use of eligibility traces, and any desired debugging information. The basic functionality of the interface program is shown in Figure 3.14.

```
Observe state information s from environment

If exploration method is ε-Greedy:
    If reducing ε: calculate current ε
    If exploration action selected with probability ε:
        Select action a with lowest number of explorations for state s
    Else:
        Select action a from agent policy function
Else if exploration method is softmax:
    Calculate τ and generate action selection probability distribution
    Randomly select action a from action selection probability distribution
Else if exploration method is OFU:
    Calculate lower confidence bounds on mean reward/value for all actions
    Select action a with lowest confidence bound

Return action a to environment and update tracking file for state s

If reinforcement learning method is SARSA:
    Calculate update q(s,a) using Equation 3.13
Else if reinforcement learning method is Q Learning:
    Calculate update q(s,a) using Equation 3.14

Update eligibility trace e(s,a)
Write update q(s,a) to update file
Write eligibility and debugging information to update files
```

Figure 3.14 – Pseudocode for the “interface program”

After the interface program has completed its functions for each timestep, an action-value training data update program or “Q update program” completes the remainder of the processes associated with the policy evaluation step of GPI. The Q update program actively monitors the centralized data file space for action-value update files generated by the interface program. Its primary function is to collect those files, and assimilate the data into the action-value dataset, which represents the action-value function. For the tabular case, this dataset is the action-value function itself. For the neural network case, this dataset represents the training data for the neural network. In addition to providing direct updates to the action-value dataset, if n-Step learning methods are being used by the agent, the Q update program conducts updates to those action-values which have an active eligibility trace. The basic functionality of the Q update program is shown in Figure 3.15.

```
Check file space for value update files; when one or more are present:

Read all $q(s,a)$ updates, and action-value table

For each $q(s,a)$ update:
    If multiple updates for same state $s$ and action $a$: average update values
    If action-value estimate exists or is w/i threshold of $s,a$ pair in action-value table:
        Replace action-value estimate with $q(s,a)$ update
        Calculate change in action-value estimate
        Update tracking file with relative changes in action-value estimates
    Else:
        Insert action-value estimate into table
    If eligibility traces used:
        Calculate and update action-value estimates with active eligibility trace
        Update eligibility traces

Write new action-value table
Update number iterations since policy update tracking file
```

Figure 3.15 – Pseudocode for the “Q update program”

As the agent is actively updating the action-value function, an agent function update program or “policy update program” carries out the policy improvement step of the GPI process. The policy update program keeps track of how many action-value updates have occurred since the last policy improvement iteration, and when a user-specified threshold is reached, conducts a complete sweep of the state space, updating the agent’s policy function with the current estimate of the optimal policy. Alternatively, the policy update program has the ability to keep track of the maximum amount of relative change in any of the action-value updates within a user-specified threshold number of recent updates, and once maximum relative change in recent updates falls below a threshold value (percent change), the policy update is conducted. For the tabular case, this policy update action is carried out by sweeping through the entire action-value table, identifying the best (lowest) action-value for each state and using the associated action as the action policy for that state. For the neural network case, the program evaluates actions at user-specified intervals for each state that has been experienced by the agent, and selects the action at each state with the best action-value based on the action-value function computed by the action-value neural network. Those actions are then provided to the agent's policy function neural network as training data. The basic functionality of the policy update program is shown in Figure 3.16.

```
If number of policy evaluations exceeds threshold, or action-value function is stable:

If neural networks in use:
    Read action-value training data table
    For each state-action pair s,a in action-value training data:
        For each action a at user-specified intervals in user-specified range:
            Calculate action-value estimates from neural network
        Find action a associated with minimum action-value estimate
Else:
    Read action-value data table
    For each state s in user-specified range of values:
        For each action a at user-specified intervals in user-specified range:
            Find action-value estimates from action-value data table
        Find action a associated with minimum action-value estimate

Write new agent policy table based on actions associated with minimum action-value est.

Reset number iterations since policy update tracking file
```
Figure 3.16 – Pseudocode for "policy update program"

For the neural network case, two neural network programs conduct the training necessary to estimate the action-value function and the agent function. Each of the neural network training programs use the training data generated by the Q update program and the policy update program to produce estimates of their respective functions. The neural network programs are trained with genetic algorithms, as discussed in Section 3.1.2.1.1. The basic functionality of each neural network program is shown in Figure 3.17.

```
Construct overall neural network object with vectors of layers and neurons for each
population member, each with initial connection weights

Assemble array of weight vectors (one vector per population member) from neural
network object

Read existing superior weight vectors (if any) into weight array

Disassemble weight vectors back into neural network object

Do while training data updates occurring or until function stabilizes:
    Read training data
    Scale input and output training data
    For each data point:
        For each genetic population member:
            For each hidden layer in neural network object:
                For each neuron:
                    Calculate activation from Equation 3.1
    Assemble array of weight vectors, calculating sum squared output error
    Calculate fitness for each weight vector
    Rank population by fitness, and assign selection probability based on fitness
    If elite segment is desired: hold elite segment out of crossover operation
    For remainder of population:
        Randomly select 2 weight vectors based on selection probability
        Perform crossover of weight vectors
        Apply mutation of weights with desired probability
    Write new file of superior weight vectors
    Disassemble weight vectors back into neural network object
```

Figure 3.17 – Pseudocode for “neural network programs”

Each of the different programs discussed above can operate independently and concurrently, allowing the system to take advantage of the parallel processing capabilities of many newer computing systems. This also allows the system to operate on more than one machine in a networked system, further expanding the capability of the system to allow for faster processing times. Additionally, because the basic reinforcement learning mechanism is based on an MDP and the agents learn from incremental experience, the system can be operated with multiple instances of the interface program, each working concurrently with their own version of the task environment model on different machines or on the same machine with multiple central processing units (CPUs) to accelerate the learning process. Each of the interface programs utilizes the shared knowledge of the overall "system of agents", and contributes to the combined learning of the system of agents. The entire process is shown in Figure 3.18.

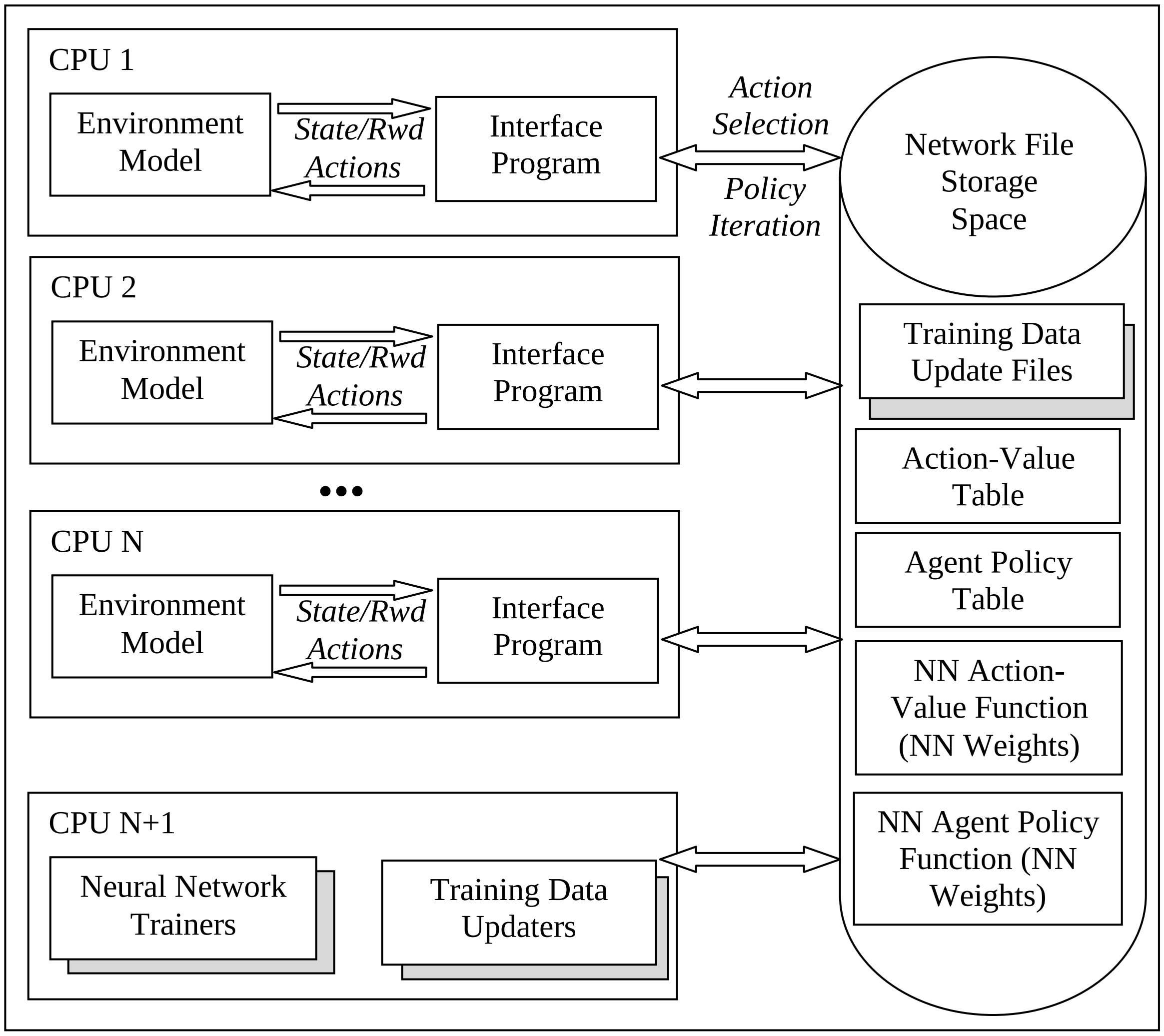


Figure 3.18 – Diagram of implementation

3.2.5.1 – Tabular Implementation

For tabular implementation, the basic steps outlined above are carried out. All action-value estimates are stored in a single text file database. Similarly, all policy values are stored in a different text file database. For the action-value database, each line in the database contains the current action-value estimate for a state-action pair. The state-action pairs are specified with the action variable at the beginning of each line, followed by each state variable, all separated by commas. The action-value is found at the end of each line, separated from the action and state variables by a semicolon. The policy value database is similar, with the differences being that there is no action variable at the beginning of the line, and the final value on each line is the policy value for the state identified by the state variables in that line. The files are sorted in ascending order by each action and state variable for rapid computational searching, as well as human review and evaluation.

The various programs in the system generally obtain and replace data from the databases using a recursive binary search method, whereby the programs incrementally bisect the database to find the value associated with a particular state or state-action pair. Using this mechanism, a particular value can be found in the database rapidly, with the worst case number of search iterations being $\log_2 N$, where N is the number of lines in the database. The mechanism generally allows for a value to be found or replaced in a database of 100,000 entries in 17 actions or less, and 1,000,000 entries in 20 actions or less. For particular processes within the Q-Learning mechanism, it is necessary to search the action-value database for the optimal action and action-value. This is carried out by conducting a binary search to determine the database location (line number) of the first action for a given state, and then obtaining and evaluating the action-values for the remaining actions in that state by simply incrementing the location (database line number) by the number of possible state variables until all actions have been evaluated.

For SARSA implementation, the values used in the action-value update equation are taken directly from the centralized data file representing the action-value function. For Q-Learning implementation, the interface program uses the mechanism described above to complete a sweep of the action-values on each timestep for the current state in order to calculate the off-policy optimal action as used in Equation 3.14. Depending on the rate of generation of the update files from the interface program, the Q update program sometimes obtains multiple update files with updates for the same state-action pair. In those cases, the mean of the updates is used to update the action-value dataset.

Since the Q update program also completes eligibility trace updates, action-value updates completed by the program are immediately returned to the action-value table so that they may be immediately updated by successive eligibility trace updates, even if those updates are contained in the collection of update files currently being processed by the updater. Eligibility trace updates are completed by utilizing a series of eligibility trace tracking files. One file is maintained for each instance of the environment model being acted upon. In this fashion, eligibility trace updates are only completed using the action-value updates from that particular environment model instance, to ensure that reward credit is only given to state-action pairs that are in the direct sequence of events proceeding them. In other words, if an action is taken in a particular state in a particular instance of the simulated environment, its eligibility trace only becomes active for that instance of the simulated environment and it is only credited with a portion of the action-value estimates of actions taken in successive states in that particular simulated environment, not by the unrelated succession of events in any other instance of the simulated environment being used by the system concurrently. In order to maintain reasonable eligibility trace information for a limited number of state-action pairs, eligibility traces are removed from the tracking files once a particular trace value drops below a user-specified threshold.

In order to properly carry out each of the various exploration algorithms, a system of tracking the exploration of the state and action space was developed. This system maintains a database of the number of times each action is taken in each possible state, as well as the mean rewards for an action taken in any particular state. The system provides for proper calculation of the decay of ε-greedy and softmax exploration using Equations 3.19 through 3.21, as well as the calculation of the confidence bounds required by OFU exploration in Equations 3.23 through 3.27. The system utilizes separate tracking files for each possible state, with each file containing a line for each possible action, storing the number of times the action has been explored, and the mean rewards experienced. By using separate files for each possible state, a very large number of files are generated. However, the amount of time and computing memory required to store and retrieve the data is greatly reduced over that required if the entire database was stored in a single file. Additionally, in recent years, increasing amounts of file storage space have become inexpensive, allowing for large numbers of small text files to be stored for minimal cost, facilitating this type of mitigation against the “curse of dimensionality”.

Through testing described in the proceeding chapter, it was found that it can be less ideal to allow values to be updated when an exploratory action is taken in a successive timestep, particularly for the ε-greedy exploration algorithm. For this reason, the water quality agent was designed to allow for these updates to be excluded from the system. However, for exploration methods such as softmax and OFU, almost every action can be considered exploratory in early training, thus exclusion of updates is not necessarily desired and that feature may be turned off.

As previously described, the system conducts policy improvement either after a user-specified number of policy evaluation updates, or when policy evaluation updates result in little change to the action-value function. In order to track the number of policy evaluation updates, a single file is updated by the Q update program, incrementing the number of policy evaluation updates since the last policy improvement sweep. When the policy improvement process takes place, the value in this file is reset. The value iteration process described in Section 3.1.3 can be accomplished by setting the number of policy evaluation updates between policy improvement cycles to a value of one. Alternatively, for the case where a stable action-value function is desired before policy improvement takes place, a file is maintained by the Q update program showing the percent change that occurred in each recent action-value update. The file also tracks the number of updates since the change occurred. If a subsequent update to a particular action-value is made, that record in the file is updated. After a user-specified number of updates have occurred since a particular action-value was updated, that action-value's tracking ends and it is removed from the tracking file to prevent rarely visited states and actions from unnecessarily delaying a policy improvement cycle. This removal process also keeps the file size manageable. Once all tracked values in the file fall below a user-specified threshold for the percent of change in recent policy evaluation updates, the policy improvement process is carried out.

The system was designed to allow for the specification of the initial values to be used for the action-value function as well as the agent policy. For the tabular case, if a single value is specified, the action-value and agent policy database files are generated with that value for each state or state-action pair. Alternatively, the user may specify files containing an initial database of action-values and agent policy values to create a more complex initial action-value or agent policy function.

For the tabular implementation, discretization of the state and action space is required. The water quality agent completes this in the interface program, where continuous values provided by the environment model are rounded to discrete values at user-specified intervals between minimum and maximum values. Any values below or above the predetermined minimum and maximum values are set to those minimum or maximum values. In this manner, basic partitioning of the state and action space is accomplished, placing each state and action variable into a "bin" or "tile" that have some meaningful value with respect to the problem at hand.

To improve on the basic partitioning of the state and action space, a technique known as "feature extraction" can be employed (Bertsekas 1996). Using this method, information from the state and action space can be condensed into a vector of features that better represent important and relevant areas of the state and action space. This could also be considered a system of categorization, where each state variable is placed into a category with similar variables. In certain cases, this could also be considered another method of partitioning, with irregularly sized partitions to better capture the important areas of a particular state variable. Those important areas might include portions of the state space where the system state is more likely to commonly exist, and where decision breakpoints are more likely to occur. The water quality agent allows for this type of feature extraction, whereby the variables are placed into integer-coded categories by the environment model or the interface program and then processed as if they were state or action variables by the system, partitioned into integer values.

3.2.5.2 – Neural Network Implementation

The neural network implementation of the water quality agent essentially utilizes many of the techniques discussed in the previous section on tabular implementation. The primary differences are that neural networks are used to estimate the action-value and policy functions. Action-value updates as shown in Equation 3.13 and 3.14 are calculated by the interface program using the action-value for $Q(s_t,a_t)$ and $Q(s_{t+1},a_{t+1})$ or $\max_a Q(s_{t+1},a_{t+1})$ as computed by the neural network, depending on whether SARSA or Q-Learning methodologies are being used. The action-value updates resulting from Equation 3.13 or 3.14 are then provided to the neural network training program by the Q update program as an additional piece of training data. The neural network weight vector is then adjusted by the neural network training program to account for this update to the action-value function training data. This presents several challenges because previous updates would also be found in the neural network training dataset, creating a bias away from the ideal approximation of the function and more towards samples from previous approximations. This problem is magnified if the overall issue being solved would result in a non-stationary action-value function. Another issue with this methodology is that the training dataset for the neural network has a much higher density of training points at states and actions encountered most often when following a particular policy. Because neural network training, like most function approximation methods, usually focuses on the minimization of the squared error in its predictions, this is likely to create better approximations of the action-value function in these commonly visited states, and worse approximations in rarely visited states (Sutton and Barto 1998).

To attempt to overcome these issues, the water quality agent is designed to replace previous points in the neural network training dataset with the updated training points. This is implemented by a search and replace algorithm within the Q update program that replaces any training point within a user-specified threshold of all state and/or action variables of the update point, generally using the same mechanisms outlined in the tabular implementation section above. If no point is found within the threshold of each of the new point’s state and action variables, the point is appended to the dataset without removing any other training data. This method attempts to solve the issues of point density, non-stationarity, and bias due to older data.

In the neural network implementation, policy improvement is conducted by computing the minimum action-value from the action-value neural network for each potential action in the states that have been visited in the task environment. The range of potential actions is determined *a priori* by user-specified minimum and maximum permissible actions and a typical action interval (i.e. likely reservoir release intervals such as 0 cms, 1 cms, 2 cms, etc.). The action associated with the minimum action-value function for each of the experienced states is provided as an additional piece of training data to the training dataset used by the neural network for the agent's policy function. By only generating training data based on states that have been visited, the policy function creates better approximations of the desired function in the state space most often visited. To avoid excessive training point density in these regions of the state space, as well as the non-stationarity issues and bias issues discussed previously, a similar search and replace algorithm is employed by the policy update program as is used by the Q update program in the development of training data for the action-value neural network.

In the water quality agent, both the action-value neural network and the agent policy function neural network are trained using genetic algorithms. The genetic algorithms generate a population of network weights, and continuously perform crossover and mutation operations to improve the optimal weight sets, as discussed in Section 3.1.2.1.1.

In the neural network implementation, the interface program utilizes the policy neural network to calculate the agent's current policy, rather than obtaining policy from a table, depending on the exploration or action selection method in use. All exploration and action selection methods are implemented in the same manner as discussed in the section on tabular implementation above. In addition, eligibility trace methods are implemented in a similar manner, with the eligibility updates calculated using values from the neural network in a manner similar to normal updates.

Initial values for the action-value neural network and policy function neural network can be provided by the user, or a file of initial values may be specified, in a manner similar to the tabular implementation. The primary difference is that in the case where a single initial value is specified for the action-value and agent policy functions, initial values are only generated for the corner points, or essentially, outer boundaries of the state or state-action space, rather than throughout the entire state or state-action domain. These initial values can then optionally be ignored after actual updates begin to occur, to prevent the initial values from having a lasting impact on the estimated function if they are not replaced by updated values. In the absence of user-defined initial values, the system will generate a single initial value of zero at the origin of the state or state-action space.

Partitioning of the state or state-action space and feature extraction can be utilized by the water quality agent under neural network function approximation as well as the tabular case, and there can be benefits to their use in certain processes under the neural network implementation. In particular, partitioning is included as a mechanism to sample the actions for a particular state to identify the optimal action for that state, both for use in the Q-learning update equation (Equation 3.14) and for generation of training data by the policy update program. Feature extraction can be used as a means of reducing the state or state-action space to a more meaningful set of partitions for use with the neural network implementation, in a similar manner to the tabular implementation.

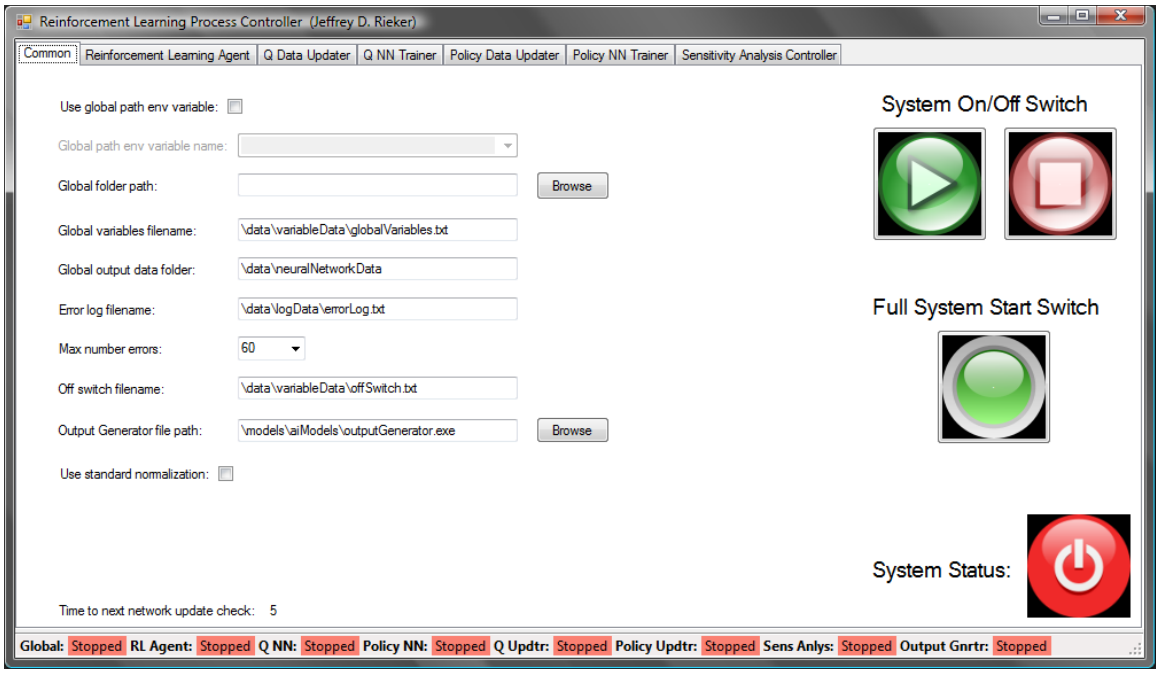
3.2.5.3 – Process Control

For the purposes of controlling the system of applications discussed above, an overall process controller was developed. This process controller helps ensure that processes are properly initialized, operated, and terminated. In addition, the process controller facilitates the setting of various options, variables, and parameters associated with each of the programs in the system. Also, the process controller allows the user to monitor each of the programs, and ensure proper functionality. Finally, the process controller provides for automated, independent, and iterative operation of the system. This allows the user to initiate a series of system runs to evaluate a variety of different options and parameter settings, and then allow the system to complete those runs without supervision. It also provides for an automated sensitivity analysis of the various parameters and settings for each of the learning mechanisms implemented in the programs.

The process controller implements a graphical user interface, allowing the user to manipulate various controls on the interface to specify the options, variables, and parameters associated with the different programs. The process controller utilizes a consistent naming convention for each of its controls, providing for a systematic mechanism to link the user-specified values of the controls to the variable names within each of the system programs. When buttons are pushed on the interface to invoke the various programs in the system, the control values are transferred to a global variable file in the centralized data file space, where they can be obtained by the initialization routines in each of the system's programs.

The process controller allows the user to monitor the various system programs to ensure proper operation. As the programs are running, the process controller constantly provides the user with feedback information such as model run progress, estimated time of run completion, CPU usage information, and iteration speed among others. The process controller contains visualization tools to examine cross sections of the action-value function and agent's policy function. In addition, it has the ability to visualize cross sections of the state and action space to examine the number of times a particular state or state-action pair has been explored. This visualization can be used to evaluate the various exploration algorithms and their functionality. For the neural network programs, the process controller allows the user to examine the neural network function, review error statistics, and track key information regarding the genetic algorithm population of solutions such as the overall population fitness or population homogeneity. A portion of this output generation is facilitated through the use of a neural network post-processing program that was created to obtain sample outputs from the neural network function at regular intervals across the entire state or state-action space.

For completion of sensitivity analyses and iterative operation of the system, the process controller allows the user to specify the options, variables, and parameters that should be included in the iterative runs, as well as a minimum bound, maximum bound, and interval for the variables and parameters as appropriate. The controller then iteratively completes training and testing runs of the reinforcement learning system, constantly monitoring the system, and initiating a new run when the previous one is complete. The graphical user interface of the process controller is shown in Figures 3.19 through 3.24.


Reinforcement Learning Process Controller (Jeffrey D. Rieker)

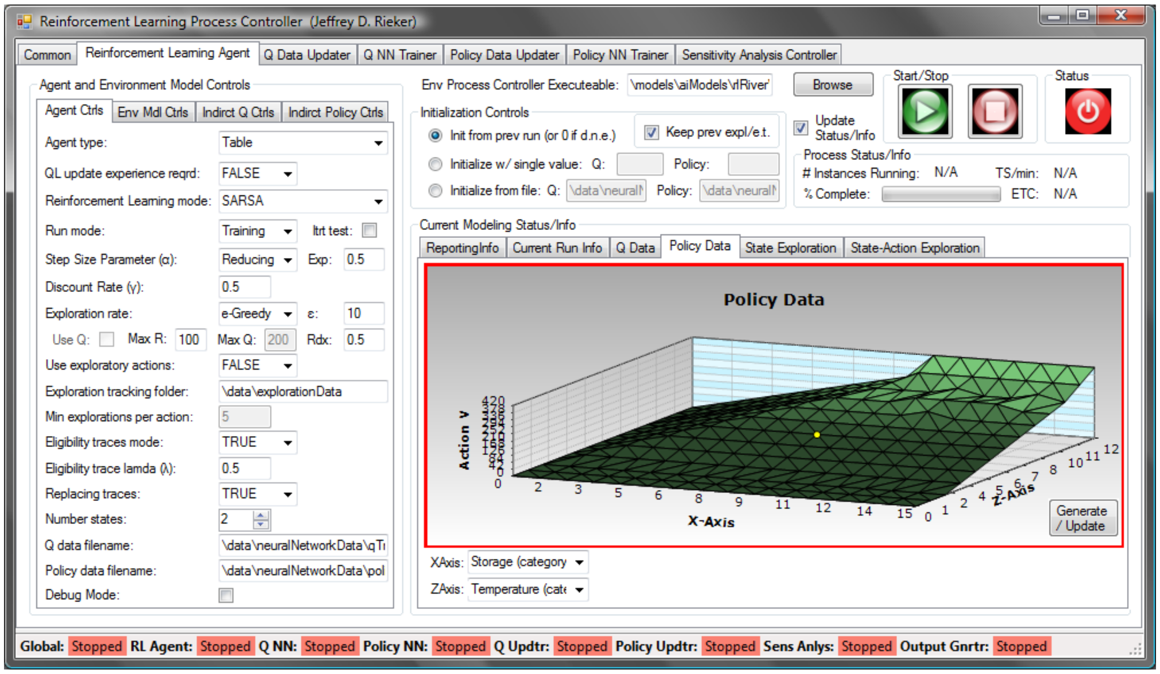
Common | Reinforcement Learning Agent | Q Data Updater | Q NN Trainer | Policy Data Updater | Policy NN Trainer | Sensitivity Analysis Controller

Use global path env variable:	☐
Global path env variable name:	
Global folder path:	
Global variables filename:	\data\variableData\globalVariables.txt
Global output data folder:	\data\neuralNetworkData
Error log filename:	\data\logData\errorLog.txt
Max number errors:	60
Off switch filename:	\data\variableData\offSwitch.txt
Output Generator file path:	\models\aiModels\outputGenerator.exe
Use standard normalization:	☐

Time to next network update check: 5

Global: Stopped RL Agent: Stopped Q NN: Stopped Policy NN: Stopped Q Updtr: Stopped Policy Updtr: Stopped Sens Anlys: Stopped Output Gnrtr: Stopped

Figure 3.19 – Process controller user interface, showing common controls for all processes


Reinforcement Learning Process Controller (Jeffrey D. Rieker)

Common | Reinforcement Learning Agent | Q Data Updater | Q NN Trainer | Policy Data Updater | Policy NN Trainer | Sensitivity Analysis Controller

Agent and Environment Model Controls

Agent Ctrls | Env Mdl Ctrls | Indirct Q Ctrls | Indirct Policy Ctrls

Agent type:	Table		
QL update experience reqrd:	FALSE		
Reinforcement Learning mode:	SARSA		
Run mode:	Training	Itrt test:	☐
Step Size Parameter (α):	Reducing	Exp:	0.5
Discount Rate (γ):	0.5		
Exploration rate:	e-Greedy	ε:	10
Use Q: ☐ Max R: 100	Max Q: 200	Rdx:	0.5
Use exploratory actions:	FALSE		
Exploration tracking folder:	\data\explorationData		
Min explorations per action:	5		
Eligibility traces mode:	TRUE		
Eligibility trace lamda (λ):	0.5		
Replacing traces:	TRUE		
Number states:	2		
Q data filename:	\data\neuralNetworkData\qT		
Policy data filename:	\data\neuralNetworkData\pol		
Debug Mode:	☐		

Env Process Controller Executeable: \models\aiModels\rlRiver

Initialization Controls

- ● Init from prev run (or 0 if d.n.e.)
- ☑ Keep prev expl/e.t.
- ○ Initialize w/ single value: Q: Policy:
- ○ Initialize from file: Q: \data\neuralN Policy: \data\neuralN

☑ Update Status/Info

Process Status/Info

Instances Running: N/A TS/min: N/A

% Complete: ETC: N/A

Current Modeling Status/Info

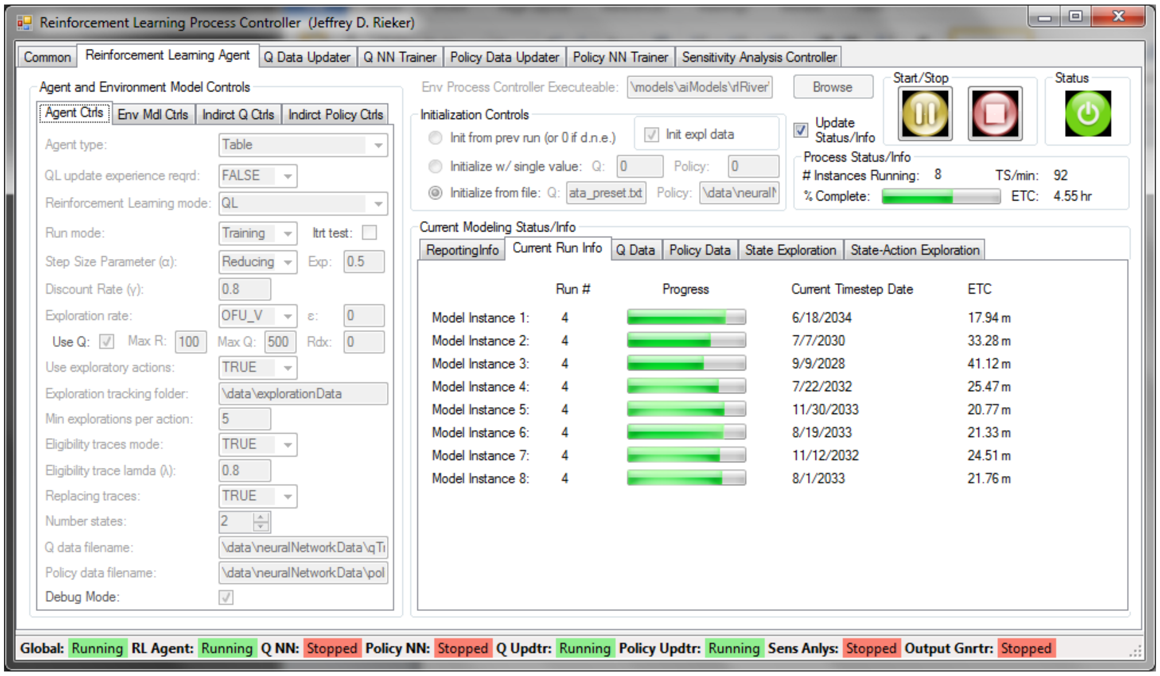
ReportingInfo | Current Run Info | Q Data | Policy Data | State Exploration | State-Action Exploration

XAxis: Storage (category

ZAxis: Temperature (cate

Global: Stopped RL Agent: Stopped Q NN: Stopped Policy NN: Stopped Q Updtr: Stopped Policy Updtr: Stopped Sens Anlys: Stopped Output Gnrtr: Stopped

Figure 3.20 – Process controller interface, showing primary reinforcement learning agent controls, and graphical representation of policy function table data



Agent and Environment Model Controls

Agent type:	Table
QL update experience reqrd:	FALSE
Reinforcement Learning mode:	QL
Run mode:	Training
Itrt test:	☐
Step Size Parameter (α):	Reducing
Exp:	0.5
Discount Rate (γ):	0.8
Exploration rate:	OFU_V
ε:	0
Use Q:	✓
Max R:	100
Max Q:	500
Rdx:	0
Use exploratory actions:	TRUE
Exploration tracking folder:	\data\explorationData
Min explorations per action:	5
Eligibility traces mode:	TRUE
Eligibility trace lamda (λ):	0.8
Replacing traces:	TRUE
Number states:	2
Q data filename:	\data\neuralNetworkData\qTı
Policy data filename:	\data\neuralNetworkData\pol
Debug Mode:	✓

Env Process Controller Executeable: \models\aiModels\rlRiver

Initialization Controls

- Init from prev run (or 0 if d.n.e.)
- Init expl data ✓
- Initialize w/ single value: Q: 0 Policy: 0
- ● Initialize from file: Q: ata_preset.txt Policy: \data\neuralN

Update Status/Info ✓

Process Status/Info

Instances Running: 8 TS/min: 92

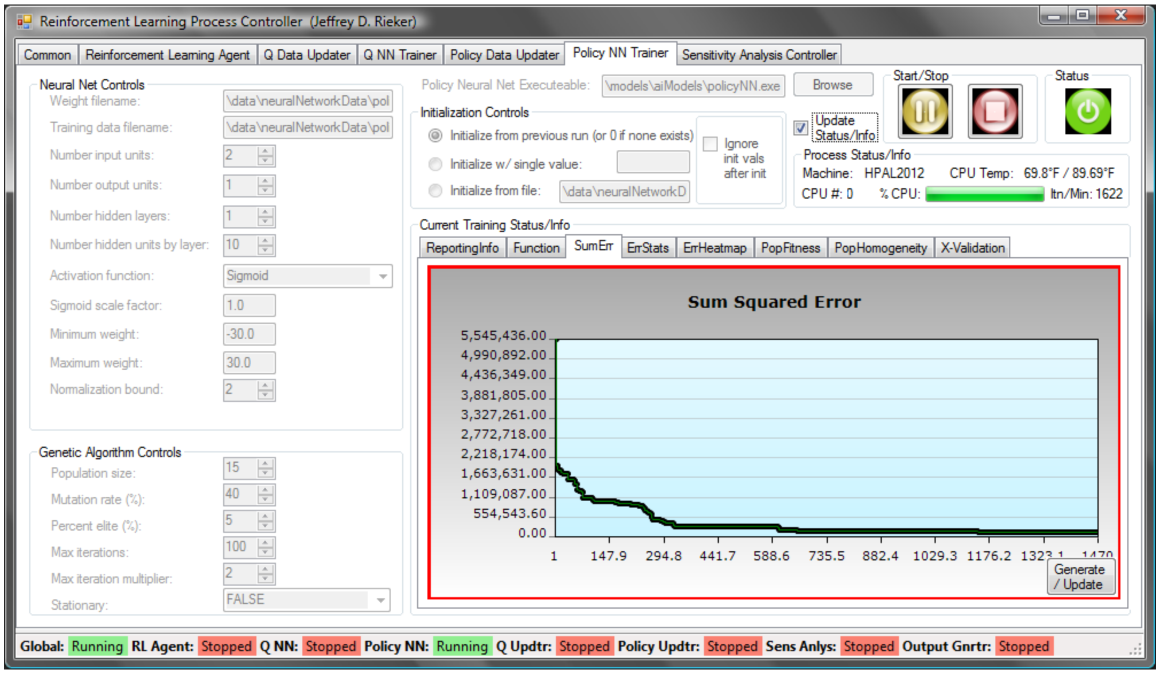
% Complete: ETC: 4.55 hr

Current Modeling Status/Info

	Run #	Progress	Current Timestep Date	ETC
Model Instance 1:	4		6/18/2034	17.94 m
Model Instance 2:	4		7/7/2030	33.28 m
Model Instance 3:	4		9/9/2028	41.12 m
Model Instance 4:	4		7/22/2032	25.47 m
Model Instance 5:	4		11/30/2033	20.77 m
Model Instance 6:	4		8/19/2033	21.33 m
Model Instance 7:	4		11/12/2032	24.51 m
Model Instance 8:	4		8/1/2033	21.76 m

Global: Running RL Agent: Running Q NN: Stopped Policy NN: Stopped Q Updtr: Running Policy Updtr: Running Sens Anlys: Stopped Output Gnrtr: Stopped

Figure 3.21 – Process controller interface, showing eight concurrent environment model instances during active simulation, with simulation status and estimated time of completion



Neural Net Controls

Weight filename:	\data\neuralNetworkData\pol
Training data filename:	\data\neuralNetworkData\pol
Number input units:	2
Number output units:	1
Number hidden layers:	1
Number hidden units by layer:	10
Activation function:	Sigmoid
Sigmoid scale factor:	1.0
Minimum weight:	-30.0
Maximum weight:	30.0
Normalization bound:	2

Genetic Algorithm Controls

Population size:	15
Mutation rate (%):	40
Percent elite (%):	5
Max iterations:	100
Max iteration multiplier:	2
Stationary:	FALSE

Policy Neural Net Executeable: \models\aiModels\policyNN.exe

Initialization Controls

- ● Initialize from previous run (or 0 if none exists)
- Initialize w/ single value:
- Initialize from file: \data\neuralNetworkD
- ☐ Ignore init vals after init

Update Status/Info ✓

Process Status/Info

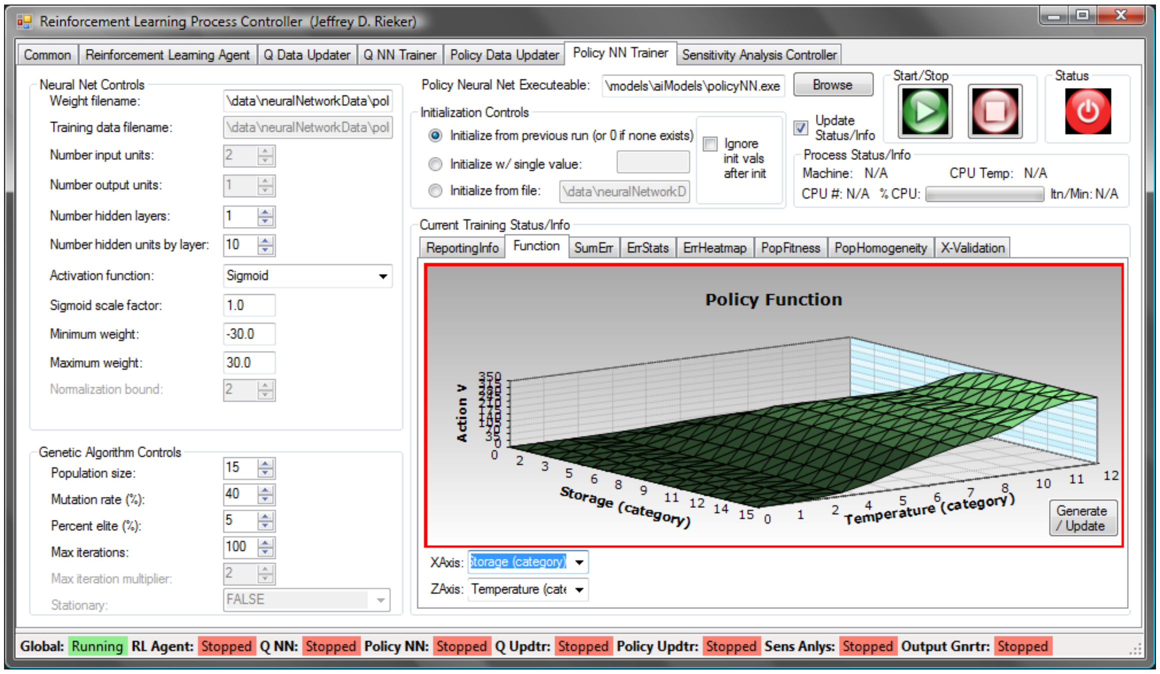
Machine: HPAL2012 CPU Temp: 69.8°F / 89.69°F

CPU #: 0 % CPU: Itn/Min: 1622

Current Training Status/Info

Global: Running RL Agent: Stopped Q NN: Stopped Policy NN: Running Q Updtr: Stopped Policy Updtr: Stopped Sens Anlys: Stopped Output Gnrtr: Stopped

Figure 3.22 – Process controller interface, showing neural network training controls, and graphical network performance display



Reinforcement Learning Process Controller (Jeffrey D. Rieker)

Common | Reinforcement Learning Agent | Q Data Updater | Q NN Trainer | Policy Data Updater | Policy NN Trainer | Sensitivity Analysis Controller

Neural Net Controls

Weight filename:	\data\neuralNetworkData\pol
Training data filename:	\data\neuralNetworkData\pol
Number input units:	2
Number output units:	1
Number hidden layers:	1
Number hidden units by layer:	10
Activation function:	Sigmoid
Sigmoid scale factor:	1.0
Minimum weight:	-30.0
Maximum weight:	30.0
Normalization bound:	2

Genetic Algorithm Controls

Population size:	15
Mutation rate (%):	40
Percent elite (%):	5
Max iterations:	100
Max iteration multiplier:	2
Stationary:	FALSE

Policy Neural Net Executeable: \models\aiModels\policyNN.exe Browse

Initialization Controls

- [✓] Initialize from previous run (or 0 if none exists)
- Initialize w/ single value:
- Initialize from file: \data\neuralNetworkD

Ignore init vals after init

[✓] Update Status/Info

Start/Stop

Status

Process Status/Info
Machine: N/A CPU Temp: N/A
CPU #: N/A % CPU: Itn/Min: N/A

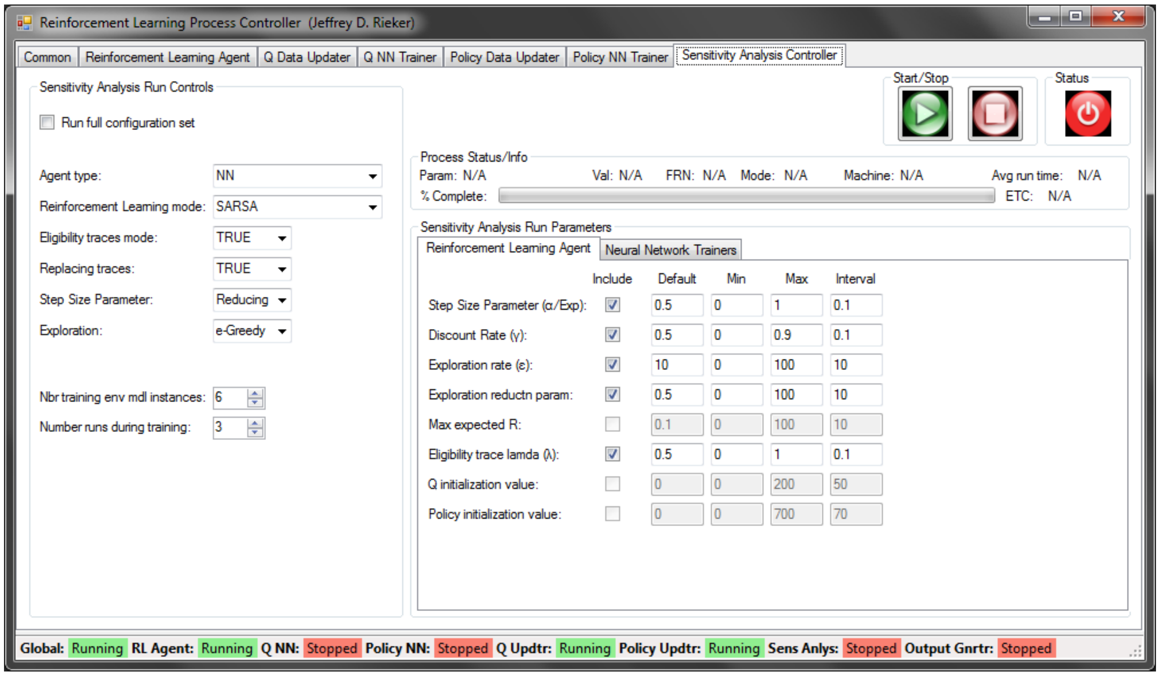
Current Training Status/Info

ReportingInfo | Function | SumErr | ErrStats | ErrHeatmap | PopFitness | PopHomogeneity | X-Validation

XAxis: Storage (category)
ZAxis: Temperature (cate

Global: Running RL Agent: Stopped Q NN: Stopped Policy NN: Stopped Q Updtr: Stopped Policy Updtr: Stopped Sens Anlys: Stopped Output Gnrtr: Stopped

Figure 3.23 – Process controller interface, showing neural network training controls, and graphical representation of agent policy function approximation



Reinforcement Learning Process Controller (Jeffrey D. Rieker)

Common | Reinforcement Learning Agent | Q Data Updater | Q NN Trainer | Policy Data Updater | Policy NN Trainer | Sensitivity Analysis Controller

Sensitivity Analysis Run Controls

Run full configuration set

Agent type:	NN
Reinforcement Learning mode:	SARSA
Eligibility traces mode:	TRUE
Replacing traces:	TRUE
Step Size Parameter:	Reducing
Exploration:	e-Greedy
Nbr training env mdl instances:	6
Number runs during training:	3

Start/Stop

Status

Process Status/Info
Param: N/A Val: N/A FRN: N/A Mode: N/A Machine: N/A Avg run time: N/A
% Complete: ETC: N/A

Sensitivity Analysis Run Parameters

Reinforcement Learning Agent | Neural Network Trainers

	Include	Default	Min	Max	Interval
Step Size Parameter (α/Exp):	✓	0.5	0	1	0.1
Discount Rate (γ):	✓	0.5	0	0.9	0.1
Exploration rate (ε):	✓	10	0	100	10
Exploration reductn param:	✓	0.5	0	100	10
Max expected R:		0.1	0	100	10
Eligibility trace lamda (λ):	✓	0.5	0	1	0.1
Q initialization value:		0	0	200	50
Policy initialization value:		0	0	700	70

Global: Running RL Agent: Running Q NN: Stopped Policy NN: Stopped Q Updtr: Running Policy Updtr: Running Sens Anlys: Stopped Output Gnrtr: Stopped

Figure 3.24 – Process controller interface, showing sensitivity analysis controls

4 – TRUCKEE RIVER CASE STUDY

For the purposes of testing and evaluating the system developed in Section 3.2, the reinforcement learning methodology was applied to the case study on the Truckee River. Application of the reinforcement learning system to the case study is covered in this chapter. Results from the application are discussed in the proceeding chapter. The first section of this chapter provides an overview of the study area, including reservoir operating procedures and water quality issues. The second section discusses previous and current water quality modeling on the Truckee River. The third section summarizes water quality monitoring and data collection efforts on the river, including past efforts producing data that can be used in models, as well as current data that may be available for real-time decision making. The fourth section reviews the development of reservoir operations models and decision support systems for the Truckee River. The fifth section of this chapter details the development of simulation models and performance functions of the Truckee River environment. The sixth and seventh sections discuss the development of the rational agent for the case study and its physical implementation.

4.1 – Truckee River Study Area

The Truckee River flows from the outlet of Lake Tahoe high in the Sierra Nevada mountain range of California to its terminus at Pyramid Lake in the desert plains of Nevada. The river is approximately 170 kilometers (105 miles) long, and contains two distinct regions with differing physical characteristics (Nevada Division of Water Planning 1997). The upper half of the basin is characterized by cold, rapid flowing water in mountainous valleys and canyons. In the lower portion of the basin, the river slows as it progresses towards its terminus, passing through the metropolitan areas of Reno-Sparks and the smaller cities and towns of Fernley, Wadsworth, and Nixon before terminating in Pyramid Lake (United States Department of the Interior 2002). The study area is shown in Figure 4.1.

The Truckee River Basin is a hydrographically closed basin, and encompasses approximately 7,925 square kilometers (3,060 square miles) (USDOI 2008). Altitudes within the basin range from approximately 3,320 meters (10,900 feet) in the headwater areas to approximately 1,160 meters (3,800 feet) at Pyramid Lake. Precipitation amounts vary widely from the mountainous areas to the desert. Much of the precipitation within the basin is in the form of mountain snow, with annual precipitation measurements reaching more than 76 centimeters (30 inches) of liquid or “snow water equivalent”. The desert areas of the basin average less than 13 centimeters (5 inches) of precipitation, due largely to the fact that the basin lies within the rainshadow of the Sierra Nevada mountain range (U.S. Geological Survey 1996; U.S. Geological Survey 1997).

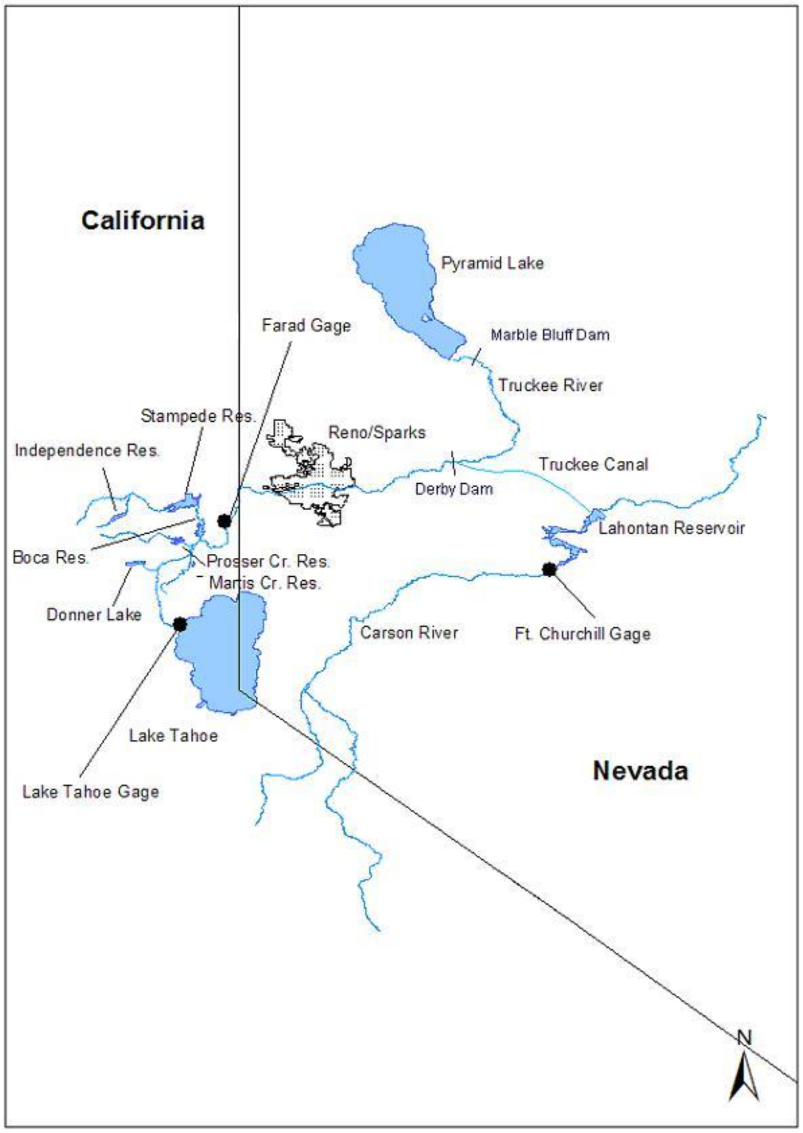


Figure 4.1 – Map of the Truckee River Basin (Rieker 2006).

There are seven large storage reservoirs located in the upper reaches of the river in California. These reservoirs control approximately 70% of the flow in the Truckee River (USDOI 2008), however they have a maximum effective total capacity that is less than two years of the total average natural river flow within the basin (Berris, Hess et al. 2001). Donner and Independence Lakes are natural mountain lakes which have small dams controlling water storage above the natural rim of the lakes. Independence Lake is owned by the Truckee Meadows Water Authority (TMWA) and Donner Lake is owned by both TMWA and the Truckee-Carson Irrigation District (TCID). The lakes provide water storage for the cities of Reno and Sparks as well as irrigation for the Newlands irrigation project near Fallon, Nevada. Martis Reservoir is a small flood control reservoir owned by the US Army Corps of Engineers (COE). Lake Tahoe, Prosser Creek Reservoir, Stampede Reservoir, and Boca Reservoir are all Reclamation facilities used to provide storage for agricultural irrigation, municipal use, and industrial use as well as water for the preservation of endangered fish in Pyramid Lake. Like Donner and Independence Lakes, Lake Tahoe was originally a natural lake, and now has a small dam controlling storage above the natural rim of the lake (Nevada Division of Water Planning 1997). Lake Tahoe is the tenth deepest lake in the world, at approximately 500 meters (1650 feet) deep. Lake Tahoe is also known for the clarity of its water. Earlier in the 20th century, it was possible to see objects at over 30 meters (100 feet) of depth (Resources 1991).

There are two large structures on the middle and lower reaches of the main stem of the river. These are Derby Diversion Dam and Marble Bluff Dam. Derby Diversion Dam is located downstream of Reno, and diverts water out of the Truckee River into the Truckee Canal to provide irrigation water for a portion of Reclamation’s Newlands Project along the Truckee Canal near the city of Fernley, Nevada. Additionally, water taken from the Truckee at Derby Dam flows through the Truckee Canal out of the basin into Lahontan Reservoir on the Carson River, and is used to irrigate agricultural lands in Reclamation’s Newlands Project within the Carson basin, as well as supply water to a national wildlife refuge and the Fallon Paiute Shoshone Indian Reservation. Marble

Bluff Dam was constructed to check the downcutting and erosion of the river channel upstream from Pyramid Lake (Nevada Division of Water Planning 1997).

Hydroelectric generation on the river is limited to four run-of-the-river power plants located along the river (Berris 1996), and a small generation plant at Stampede Dam. The run-of-the-river power plants have a maximum combined capacity of approximately 10 megawatts, and the power plant at Stampede has a capacity of 3.6 megawatts (Reclamation 2010).

Pyramid Lake represents the terminus of the Truckee River, located on the Pyramid Lake Indian Reservation. Water generally only leaves the lake by evaporation. Due to increased consumptive use of the river's water supply over the last one hundred years, as well as the transbasin diversion of water away from the Truckee River through the Truckee Canal to the Newlands Project, Pyramid Lake's water surface elevation has declined greatly at times. The lowest recorded lake elevation occurred in 1967, when the lake was almost 29 meters (95 feet) lower than its highest recorded elevation in 1891. The lake is home to two fish that are on the federal threatened and endangered species list, the cui-ui and the Lahontan cutthroat trout (LCT) (Nevada Division of Water Planning 1997). Additionally, seven other fish occur naturally within the Truckee basin, and approximately fourteen non-native species are known to reside in the Truckee river and its lakes and reservoirs (United States Department of the Interior 2002).

The average annual discharge of the Truckee River from California into Nevada is approximately 693 million cubic meters (MCM) (561,800 acre-feet). The average diversion from the river at Derby Dam is approximately 199 MCM (161,500 acre-feet) per year (USDOI 2008). Active water rights in Nevada in 2002 indicated a demand of

271 MCM (219,691 acre-feet) per year for agricultural, municipal, and industrial uses within the basin (United States Department of the Interior 2002). Agriculture generally accounts for approximately 70% of the consumptive use within the basin (Warwick, Cockrum et al. 1999), and reports show that the demand on the river can be greater than the supply (Berris 1996). On average, approximately 525 MCM (425,700 acre-feet) of water flows into Pyramid Lake per year (United States Department of the Interior 2002). Average and maximum flows at various streamgages along the river are shown in Table 4.1.

Table 4.1 – Streamgage statistics for the Truckee River system (USGS 2010)

USGS Streamgage	**Mean Daily Streamflow**	**Maximum Streamflow of Record**
Truckee River @ Tahoe City (10337500)	6.4 cms (227 cfs)	76.2 cms (2,690 cfs)
Truckee River @ Farad (10346000)	21.5 cms (758 cfs)	495.5 cms (17,500 cfs)
Truckee River @ Reno (10348000)	19.2cms (679 cfs)	589.0 cms (20,800 cfs)
Truckee Canal nr Wadsworth (10351300)	6.3 cms (222 cfs)	26.1 cms (921 cfs)
Truckee River nr Nixon (10351700)	15.4 cms (544 cfs)	600.3 cms (21,200 cfs)

Water storage and flow in the Truckee River are governed by a large number of court decrees, court decisions, laws, and regulations. These include the Truckee River General Electric Decree (1915), Truckee River Agreement (States, District et al. 1935), Orr Ditch Decree (1944), Tahoe-Prosser Exchange Agreement, Newlands Project Operating Criteria and Procedures (OCAP) (Reclamation 1997), Interim Storage Agreement, and the Water Quality Settlement Agreement (Reno, Sparks et al. 1996). These governing documents are well described in current literature (United States

Department of the Interior 2002; Boyer 2006; USDOI 2008). The documents are summarized in Appendix A.

A detailed overview of reservoir storage and release operations on the Truckee River is provided in Appendix B. Generally, the Truckee River Agreement is the primary controlling instruction for the current operation of the river and reservoir system. The Truckee-Carson-Pyramid Lake Water Rights Settlement Act of 1990 (Public Law 101-618) has resulted in a new agreement, known as the Truckee River Operating Agreement (TROA) (States, California et al. 2008). The agreement was signed on September 6, 2008, and made effective in 2009, however has not yet been implemented due to a number of legal challenges against the agreement. The agreement, once implemented, will alter the way water is stored and released in the Truckee Basin. The agreement is designed to increase the flexibility and efficiency of water operations within the basin, while conforming to current operating procedures, laws, and water rights (USDOI 2008).

4.1.1 – Water Quality Issues

Water quality issues in the Truckee River have been studied extensively throughout the last century, primarily due to the fact that the water quality in the river has been heavily impacted by human use almost from the time of the arrival of the very first settlers. The first permanent settlement along the Truckee River was built in 1852, and by 1859, logging operations in the basin had already caused severe degradation of river water quality, resulting in the first man-made impediments to native fish spawning (Nevada Division of Water Planning 1997). Many anecdotal reports during the early and mid-twentieth century illustrate the wide variety of water quality problems in the river caused by human activities in the basin (Nevada Division of Water Planning 1997). Reports and studies completed in the past three decades have provided a more detailed understanding of the exact nature and causes of the degradation of water quality (Brown, Nowlin et al. 1986; U.S. Geological Survey 1996; U.S. Geological Survey 1997).

The primary water quality constituents of interest on the Truckee River include temperature, dissolved oxygen, nitrogen, phosphorus, and total dissolved solids (Brown, Nowlin et al. 1986; U.S. Geological Survey 1996; USDOI 2004). Temperature in the river has impacts on the levels of dissolved oxygen, and high water temperatures can harm or even kill the threatened and endangered native species in the river (U.S. Army Corps of Engineers 1995; U.S. Geological Survey 1996). Dissolved oxygen is required for the growth and survival of fish and other aquatic life (United States Department of the Interior 2002). Low levels of dissolved oxygen can harm or kill the threatened and endangered species of fish in the river (U.S. Geological Survey 1997), as well as inhibit their ability to successfully spawn (Hoffman and Scoppettone 1989; U.S. Geological Survey 1997). Nitrogen and phosphorus are two naturally occurring nutrients that stimulate the growth of algae and other aquatic plants. When excess amounts of these nutrients are introduced to the river, an overabundance of aquatic growth can occur, causing low dissolved oxygen levels due to both nighttime respiration and decay of dead algae and vegetation (U.S. Environmental Protection Agency Office of Water 1994; U.S. Geological Survey 1997). Measurements of total dissolved solids in the water are an indication of the mineral content within the river. This is of particular importance on the Truckee, since high levels of dissolved solids are harmful to most beneficial uses of the river, including municipal, industrial, agricultural, and instream uses (Brown, Nowlin et al. 1986). Additionally, since Pyramid lake is a terminus lake, the total mass of minerals reaching the lake will be permanently stored there, increasing the mineral content of the lake as a whole.

Standards for many water quality constituents have been developed by the State of Nevada and the Pyramid Lake Paiute Tribe for selected locations on the river. A summary of key standards is shown in Table 4.2.

Table 4.2 - Summary of selected water quality standards for the Truckee River (Nevada 2010)

Water Quality Constituent	**Standard**
Maximum Temperature	≤7°C Nov-Mar, upstream of Lockwood Bridge ≤13°C Nov-Mar, downstream of Lockwood Bridge ≤13°C Apr-May, upstream of Lockwood Bridge ≤21/22°C Apr/May, downstream of Lockwood Bridge ≤14°C Apr-Jun, downstream of Derby Dam ≤17/21/22°C Jun/Jul/Aug, upstream of Lockwood Bridge ≤23°C Jul-Oct, downstream Lockwood Bridge ≤25°C Jul-Oct, downstream Wadsworth ≤23°C Sept-Oct, upstream of Lockwood Bridge
Minimum Dissolved Oxygen	≥6.0 mg/l S.V. Nov-Mar (Nov-June downstream Wadsworth) ≥5.0 mg/l S.V. Apr-Oct (Jul-Oct downstream Wadsworth)
Total Phosphates	≤0.10 mg/l A-Avg. upstream of Lockwood Bridge ≤0.05 mg/l A-Avg. downstream of Lockwood Bridge
Nitrogen	≤2.0 mg/l S.V. Nitrate ≤0.4 mg/l S.V. Nitrite ≤0.75 mg/l A-Avg. TN downstream of Lockwood Bridge ≤1.2 mg/l S.V. TN downstream of Lockwood Bridge
Total Dissolved Solids	≤500 mg/l A-Avg.
Suspended Solids	≤25 mg/l S.V. upstream of Lockwood Bridge ≤50 mg/l S.V. downstream of Lockwood Bridge

Generally, the quality in the river degrades with increasing distance downstream of Lake Tahoe with respect to most of the water quality constituents of interest (Brown, Nowlin et al. 1986; U.S. Geological Survey 1997). The reach of the Truckee River from its headwaters at Lake Tahoe to the United States Geological Survey (USGS) flow gaging station at Farad (Farad gage) located on the California-Nevada state line is characterized by low temperatures and generally high water quality. The only water quality issues noted in this reach of the river are occasional temperature problems. The issues with temperature are primarily associated with release of water from upper basin reservoirs when reservoir elevations are low, particularly late in the summer. The outflow from the reservoirs can occasionally be warm under these conditions, degrading the water temperatures in the river. (Neumann 2001; USDOI 2004). None of the reservoirs have selective withdrawal capabilities that might be able to mitigate these issues.

Water quality degradation generally begins to occur in the reach of river extending from the Farad gage through the Reno-Sparks metropolitan area (also known as the Truckee Meadows). The bulk of the degradation in this section is due to decreased velocity caused by decreased stream gradient, decreased streamflow due to diversions, and increased pollutant loadings from agricultural return flows and urban runoff from the Reno-Sparks metropolitan area (United States Department of the Interior 2002). The agricultural return flows and urban runoff cause an increase in both temperature and dissolved solids (Brown, Nowlin et al. 1986). Decreased streamflow and velocity in association with warmer air temperatures result in increased temperature (Neumann, Rajagopalan et al. 2003).

Upon exiting the Truckee Meadows area, the river's quality becomes increasingly degraded due to a number of issues. There are three point-source loads of water quality constituents that enter the river near the Vista USGS streamgage at the downstream end of the Truckee Meadows. These are Steamboat Creek, the North Truckee Drain, and the outfall from a wastewater treatment plant known as the Truckee Meadows Water Reclamation Facility (TMWRF). Waters from Steamboat Creek contain an increased loading of total dissolved solids and nutrients from agricultural returns, urban runoff, and geothermal springs (Brown, Nowlin et al. 1986). The North Truckee Drain is primarily an agricultural return flow containing increased loads of dissolved solids and nutrients (Brown, Nowlin et al. 1986). The TMWRF primarily provides the river with increased nutrients and some dissolved solids (Brown, Nowlin et al. 1986; U.S. Geological Survey 1997). All three of these inflows to the river generally increase river temperature (Brown, Nowlin et al. 1986). The reach of river extending from the Vista gage to Derby Dam is generally characterized by low quality water due to these loadings (Warwick, Cockrum et al. 1999), and the river at Lockwood has been placed on the Nevada 303 (d) list as not supporting its designated uses due to increased concentrations of total nitrogen, total phosphorus, and total dissolved solids (U.S. Environmental Protection Agency Office of Water 1994). The 303 (d) listing is a provision of the Clean Water Act which requires that states must develop total daily allowable loadings to the river for dischargers, to ensure that rivers are capable of meeting their designated uses (Bhimani, McDonald et al. 2002).

Upon reaching Derby dam, a large portion of the river's flow is sometimes diverted from the river through the Truckee Canal. This removes many of the constituent loadings from the river, however diversions are highly dependent on hydrologic conditions in the neighboring Carson River basin. A summary of the number of years when full diversion through the Truckee Canal to Lahontan Reservoir on the Carson River was required between 1987 and 2006, as well as average daily flow into the canal in those years, is shown in Table 4.3. A summary of the number of years of diversion only to the local irrigators on the Truckee Canal between 1987 and 2006, as well as average daily flow into the canal in those years is shown in Table 4.4.

Table 4.3 - Number of years between 1987 and 2006 when full transbasin diversion was required through the Truckee Canal (USGS 2010)

Month	Number of Years (1987-2006)	Average Monthly Volume, MCM (AF)	Average Daily Flow, cms (cfs)
Jan	14	17.52 (14,200)	6.53 (231)
Feb	13	21.22 (17,200)	8.49 (300)
Mar	13	34.66 (28,100)	12.93 (457)
Apr	11	35.03 (28,400)	13.51 (477)
May	10	26.89 (21,800)	10.02 (354)
Jun	9	22.33 (18,100)	8.6 (304)
Jul	9	16.78 (13,600)	6.28 (222)
Aug	5	18.38 (14,900)	6.85 (242)
Sept	5	18.87 (15,300)	7.3 (258)
Oct	7	16.65 (13,500)	6.2 (219)
Nov	14	14.19 (11,500)	5.67 (200)
Dec	15	16.9 (13,700)	6.33 (223)

Table 4.4 - Number of years between 1987 and 2006 when no transbasin diversion was required through the Truckee Canal, based on USGS streamflow data (USGS 2010)

Month	Number of Years (1987-2006)	Average Monthly Volume, MCM (AF)	Average Daily Flow, cms (cfs)
Jan	6	1.11 (900)	0.4 (14)
Feb	7	0.62 (500)	0.27 (9)
Mar	7	0.99 (800)	0.36 (13)
Apr	9	3.21 (2,600)	1.24 (44)
May	10	6.41 (5,200)	2.41 (85)
Jun	11	6.04 (4,900)	2.31 (82)
Jul	11	7.03 (5,700)	2.63 (93)
Aug	15	6.66 (5,400)	2.48 (88)
Sept	15	5.18 (4,200)	2.01 (71)
Oct	13	3.95 (3,200)	1.47 (52)
Nov	6	1.73 (1,400)	0.68 (24)
Dec	5	0.62 (500)	0.25 (9)

The water quality in the reach of river between Derby Dam and Pyramid Lake continues to be impacted from many of the pollutants encountered upstream, although some reduction in nutrients occurs due to biological assimilation (Brown, Nowlin et al. 1986; U.S. Geological Survey 1997), particularly in late summer (Galat 1990). Constituent loadings to the river and increased temperature in this reach come from agricultural return flows (Brown, Nowlin et al. 1986). Groundwater inflows to the river generally result in a decrease in temperature, but also contribute dissolved solids (Brown, Nowlin et al. 1986; Pohll, McGraw et al. 2001). Increased water temperature may also be due to reduced flows (Wagner and Lebo 1996; Neumann, Rajagopalan et al. 2003) and loss of shade from riparian vegetation (U.S. Army Corps of Engineers 1995).

Pyramid Lake itself suffers as a result of upstream water quality issues, primarily due to the fact that it is a desert terminus lake and water only leaves through evapotransipration (Nevada Division of Water Planning 1997). Dissolved solids are permanently retained in the lake, increasing salinity. This problem is compounded by the fact that flows to the lake have been reduced due to upstream consumptive uses, reducing total lake volume (USDOI 2004).

It has been suggested that increased streamflow in the river would improve instream water quality by lowering summertime water temperatures and diluting the concentrations of various pollutants such as total dissolved solids and nutrients, which would in turn decrease the likelihood of low dissolved oxygen concentrations (Chiatovich and Fordham 1979; Protection 1994; United States Department of the Interior 2002; USDOI 2004). Reports have attempted to show relationships between streamflow and measures for the various water quality constituents (Galat 1990; U.S. Geological Survey

1997). These relationships indicate that there may be an improvement in water quality for some constituents at some locations, however in other locations the water quality can degrade as a result of increased flows. Total dissolved solids concentrations in lower portions of the river tend to improve with increased streamflow due to dilution (U.S. Geological Survey 1996). However, nitrate (a measure of nitrogen) shows an increase, then decrease with increased flow (U.S. Geological Survey 1997). This is attributed to a "flushing" response in the lower reaches of the river (Hoffman 1990; U.S. Geological Survey 1997). Another problematic issue with increased flows is that algae attached to the streambed, also known as periphyton, can become dislodged during higher flows and decay as it flows towards Pyramid lake (Warwick, Cockrum et al. 1999). Reports have cited the nighttime respiration of periphyton as being the primary cause of low dissolved oxygen in the river, and have stated that the limiting nutrient for periphyton growth is nitrogen rather than phosphorus (Chen 2002). These same reports note that increased streamflow will not improve water quality due to dilution, however the improvement to water quality would only come through the reduction of nutrients (Chen 2002).

4.1.2 – Water Quality Settlement Agreement

In response to the ongoing degradation of water quality in the Truckee River entering the Pyramid Lake Indian Reservation, the Pyramid Lake Paiute Tribe (Tribe) threatened and filed litigation against Reno, Sparks, the Environmental Protection Agency (EPA), and the Nevada Division of Environmental Protection (NDEP). To settle the litigation, an agreement was reached by Reno, Sparks, Washoe County, the US

Department of the Interior (DOI), the US Department of Justice (DOJ), EPA, NDEP, and the Tribe. The agreement is known as the Truckee River Water Quality Settlement Agreement (WQSA), and was signed by all involved parties in 1996 (Reno, Sparks et al. 1996). The agreement states that a total of twenty-four million dollars ($24,000,000) is to be expended by several of the involved parties to purchase water rights on the Truckee River. The United States federal government was obligated to purchase twelve million dollars worth of water rights, while the remaining twelve million dollars worth of purchases were to be pursued by Reno, Sparks, and Washoe County. The water purchased under the agreement is to be stored in the federally owned reservoirs in the upper Truckee basin, and used as necessary to augment instream flows in the Truckee River to improve water quality in the river and increase flows to Pyramid Lake. Specifically, the water will be used to:

- Assist in compliance with water quality standards
- Improve water quality
- Maintain and preserve the lower Truckee River and Pyramid Lake for the following purposes:
    - Fish and wildlife
    - Threatened and endangered species
    - Recreation

The agreement stipulates that the water will be stored in the Truckee River reservoirs, and will be released from the reservoirs according to a release schedule cooperatively developed by Reno, Sparks, Washoe Co., and DOI. The releases will be scheduled to meet the purposes of the agreement, and will use the following priority order of goals to gain the most benefit from the available water:

- Meet water quality standards in the reach from the USGS gage at Vista to Pyramid Lake
- Improve water quality in that reach when sufficient water is not available to meet the standards
- Maintain habitat for fish and riparian species downstream of Derby Dam
- Add to aesthetic and recreational value of the river from Reno-Sparks to Pyramid Lake

(Reno, Sparks et al. 1996). Though improvement of conditions at Pyramid Lake itself is not a specific goal of the WQSA, a study has shown that, in addition to improving water quality in the river, the augmented flows would result in an increase in the surface elevation of Pyramid Lake (Pratt 1997).

Previous studies have shown that using the WQSA purchased water rights to maintain target flows at the USGS streamgages at Sparks and Nixon during June through September would address the WQSA flow enhancement goals. These target flows are 7.8 cubic meters per second (cms) (275 cubic feet per second (cfs)) and 3.8 cms (135 cfs) at the respective gages (USDOI 2002). These preliminary targets were developed with the understanding that real-time water management operations may provide more beneficial flows by reacting to physical conditions within the basin (United States Department of the Interior 2002). Preliminary studies indicated that the most likely acquisition of water rights would provide 10.5 MCM (8,500 acre-feet) to 16.5 MCM (13,350 acre-feet) of water to help meet the goals of the WQSA, however the final volume acquired may be less (United States Department of the Interior 2002), due to the fact that market prices for water rights have sharply increased in recent decades (Colby, McGinnis et al. 1991). Some reports show that the potential acquisition volume may be as high as 29.6 MCM (24,000 acre-feet) (Nevada Division of Water Planning 1997; Lovell 2000), or slightly less at 21.0 MCM (17,000 acre-feet) (Neumann 2001). As of

January, 2011, the total amount of water rights acquired under the WQSA was 6.7 MCM (5,422.2 acre-feet), of which 6.2 MCM (4,993.2 acre-feet) were eligible for transfer for the purposes of the WQSA, with the remainder representing water rights that were previously inactive or otherwise not eligible to be put to use under the WQSA (Great Basin Land and Water 2011). The total amount that was transferred for in-stream flow purposes in the 2010 season was 4.9 MCM (3,973.02 acre-feet) (Federal Water Master's Office 2010).

Only one previous study has evaluated an improved strategy or policy for the release of this water (Neumann 2001; Neumann, Rajagopalan et al. 2003; Neumann 2006). The details of that study are summarized in Section 4.2 of this document.

4.2 – Truckee River Water Quality Modeling

Water quality has been modeled in various forms on the Truckee River for well over three decades. Many models are simple regression relations, however some previous studies have produced complex water quality simulations. Several previous reviews outline the history of water quality modeling on the Truckee (Taylor 1998; Neumann 2001).

One of the earliest water quality models on the Truckee was a simulation model for total dissolved solids (TDS), developed in the late sixties and early seventies (Sharp, Bateman et al. 1970). This model used elements of mass balance and relationships between flow and TDS concentrations, and was later modified for use with a reservoir operating model to show that flow augmentation would help meet TDS standards except during drought conditions (Chiatovich and Fordham 1979). A daily water temperature prediction model was completed in 1975 using the concepts of heat transfer. The model was used to show that a minimum flow in the river below Derby Dam should be kept above 2.8 cms (100 cfs) during summer months when diversions at the dam were low to ensure survival of LCT in the river. The reason that the restriction occurred at low diversion rates was that lower diversion rates required less flow through the Truckee Meadows above Derby Dam, resulting in higher stream temperatures by the time the water reached the downstream reaches (Rowell 1975). A model was created in 1978 as the primary product of a series of workshops involving stakeholders within the basin, led by the USGS (Andrews, Ellison et al. 1979). The model included a water supply submodel and a water quality submodel. The series of workshops appear to have been a portion of the first phases of a larger river quality assessment by the USGS. The USGS also developed a simplistic daily stream temperature model in 1986 based purely on harmonic functions (Brown, Nowlin et al. 1986). A daily timestep FORTRAN model called the Truckee River Water Quality Model (TRWQ) was created by Nowlin (Nowlin 1987) in 1987 which simulated concentrations of TDS, DO, and nutrients (among other constituents) from Reno to Marble Bluff Dam. The USGS created a nutrient loading model in 1997 that estimated nutrient loads based on a regression with instantaneous streamflow (U.S. Geological Survey 1997). Warwick et al. created a water quality model of the downstream reaches of the Truckee River in 1999 using the Water Quality Analysis Program (WASP). The program was altered to consider the effects of periphyton growth, and was used to reveal the fact that groundwater influx in the downstream river reaches was likely to have a significant impact on water quality, particularly with regard to the concentration of nutrients and their effect on periphyton growth and dissolved oxygen problems (Warwick, Cockrum et al. 1999).

In 1987, Caupp et al. created the Dynamic Stream Simulation and Assessment Model (DSSAM) based on a periphyton model developed by Runke (Caupp, Brock et al. 1991; Warwick, Cockrum et al. 1999) as well as the Nowlin model (U.S. Environmental Protection Agency Office of Water 1994; Berris 1996). The DSSAM model provided the ability to model a wide variety of water quality constituents including dissolved oxygen, TDS, and nutrients, and also had the ability to model benthic algae (periphyton) and its effects on dissolved oxygen levels (Protection 1994). The model was later extended to include temperature, and was then called DSSAMt. Currently, there are 26 total water quality parameters modeled (USDOI 2004). The DSSAM model was utilized in a variety of studies throughout the 1990's and 2000's. The model was used to determine the Total Maximum Daily Loadings (TMDLs) for the Truckee River at Lockwood for the EPA in 1994 (Protection 1994; U.S. Environmental Protection Agency Office of Water 1994). The model was also used in a 1995 study by the US Army Corps of Engineers (COE) to evaluate the potential benefits of restoration measures applied to the downstream reaches of the river (U.S. Army Corps of Engineers 1995). The model was subsequently used to evaluate the target flows required to attain beneficial water quality levels during WQSA negotiations, and was later used to evaluate various scenarios presented in the WQSA Environmental Impact Statement (EIS) (United States Department of the Interior 2002). The DSSAMt model was further used to compare the various alternatives provided in the TROA EIS, and model the overall effect of TROA on water quality in the Truckee River (USDOI 2008).

In 1998 the USGS completed a daily flow routing model that had the capability to model temperature and TDS concentrations from Lake Tahoe to Marble Bluff Dam. The model was built using the USGS's Hydrologic Simulation Program Fortran (HSPF) framework (Berris, Hess et al. 1996; Taylor 1998). The HSPF framework was later used by the Cities of Reno and Sparks and Washoe County (local agencies) to create a water quality model of the Truckee River with a similar configuration to that of the USGS. The later implementation of the HSPF framework is being referred to as the "Truckee River HSPF Model" or TrHSPF. The effort by the local agencies included the TrHSPF model, the DSSAMt model created by Brock et al., and a model known as the Watershed Analysis Risk Management Framework (WARMF). The effort was geared towards managing and revising the waste load allocations placed on the TMWRF, which is managed by the local agencies (Bhimani, McDonald et al. 2002; Engineers 2003). The effort utilized the DSSAMt model to research and refine water quality modeling methods for the Truckee River, particularly the modeling of periphyton. The refined modeling algorithms were then placed into the TrHSPF model, which was used to evaluate the potential for revising TMDLs for the TMWRF, as well as evaluate impacts of structural and non-structural alternatives to waste management for improvement of water quality in the river. The WARMF was designed to provide inputs to the DSSAMt and TrHSPF models. The inputs from the WARMF include pollutant loadings to the river from the surrounding watershed, given inputs of meteorology and land use data (McDonald, Bhimani et al. 2000; Bhimani, McDonald et al. 2002). The WARMF can also be used to develop TMDLs for various point and non-point sources of pollutants along the entire Truckee River (McDonald, Bhimani et al. 2000). The suite of models (DSSAMt, TrHSPF, WARMF) are intended to also provide assessments of the benefits of WQSA water rights purchases (Reno, Sparks et al. 2003).

Neumann developed a predictive model for Truckee River temperature at Reno, based on outflows from the basin's reservoirs and daily air temperature in Reno (Neumann 2001; Neumann, Rajagopalan et al. 2003; Neumann 2006). The model was simplified into a regression table, and implemented within the RiverWare decision support framework for management of WQSA stored water (Neumann, Rajagopalan et al. 2003; Neumann 2006). This model was the first attempt at developing a decision support mechanism capable of managing the reservoirs for optimized operation of water stored under provisions of the WQSA. The model was used as a demonstration to show that flow augmentation would help reduce stream temperatures. However, the study showed that extreme temperatures were still likely to occur (Neumann 2006). The study pointed out that the wide variety of factors affecting water temperature and quality downstream of Reno would complicate the development of a water quality model in the downstream reaches. The study suggested that future development of more generalized, complex water quality models would be necessary for the appropriate management of the WQSA stored water. The study pointed out that those models would have to include dissolved oxygen, nutrient loadings, and other constituents, and would have to take long-term climate predictions into consideration. The Neumann study serves as the predecessor to the research conducted for this dissertation.

4.3 – Truckee River Water Quality Monitoring

Water quality has been monitored on the Truckee River for several decades by a variety of agencies. These agencies include the USGS, Desert Research Institute, University of Nevada-Reno, TMWRF, NDEP, EPA, US Fish and Wildlife Service, Federal Water Master, Nevada Division of Wildlife, Reclamation, TMWA, Washoe County, Reno, Sparks, Pyramid Lake Paiute Tribe, Truckee River Yacht Club, Nature Conservancy, Nevada State Health Division, and Truckee-Tahoe Sanitation Agency among others (Protection 1994; Brock 2000; Reno, Sparks et al. 2003). In general, most of the longer-term monitoring efforts on the river have consisted of non-continuous monitoring. This type of monitoring usually consists of samples taken at weekly, monthly, bimonthly, or quarterly intervals (Brock 2000). A monitoring network operated by the TMWRF is the only long-term semi-continuous effort currently in place. There are currently nine monitoring stations providing hourly data for water temperature, pH, specific conductance (an indicator of TDS), and dissolved oxygen concentration. These monitoring stations typically obtain data from April 1st through November 30th of each year, although many of the stations operate continuously throughout the year. The stations sometimes do not operate during periods of extreme high flow. Most of the stations have a period of record extending back several years, in some cases as far back in time as 1993 (Facility 2004). The TMWRF network has been used in conjunction with other monitoring networks for several coordinated monitoring activities throughout the past two decades (Association 1987). Two of the coordinated efforts were used to generate the data necessary to calibrate the DSSAMt and TrHSPF models. A coordinated monitoring effort during September of 1989 produced water quality information during a fairly stable period in river flows and constituent loadings that was used to calibrate the DSSAM model for the purposes of setting TMDL's for the river (Protection 1994). Another coordinated effort organized fourteen agencies to monitor river quality for development of the TrHSPF model as well as developing revised TMDLs and assessing compliance with water quality objectives within the river (McDonald, Bhimani et al. 2000; Bhimani, McDonald et al. 2002; Reno, Sparks et al. 2003).

4.4 – Truckee River Hydrologic, Policy, and Decision Support Modeling

A wide variety of hydrologic and policy simulation models have been constructed for the Truckee River. Several of these models have been designed as decision support systems. There are several existing reviews outlining the history of this type of modeling on the Truckee River (Cobb, Olson et al. 1990; Berris 1996; Berris, Hess et al. 2001).

One of the earliest computer models simulating the Truckee River was a monthly flow simulation model constructed by the Desert Research Institute (DRI) based on simple mass balance (Sharp, Bateman et al. 1970; Berris, Hess et al. 2001). This model was operated using a deterministic dynamic programming mechanism to propose an optimal operating policy for the river (Fordham 1972). The model was also used in conjunction with a water quality model described previously (Sharp, Bateman et al. 1970; Chiatovich and Fordham 1979). When used in conjunction with the water quality model, the model optimized the total release from all reservoirs. However, it did not focus on individual reservoirs due to the computational complexity of the problem (Chiatovich and

Fordham 1979). The model was later altered into a network flow model which used the out-of-kilter solving algorithm of linear programming to determine operating policies for the combined Truckee-Carson system. The optimal policies suggested by the model were reported to potentially save 70.3 MCM (57,000 acre-feet) of water that could be delivered to Pyramid Lake under operating conditions that existed in the early 1970's (Fordham 1972). Others from DRI later constructed an hourly flow model of the Truckee River capable of simulating flow peaks and floods, however the model did not include reservoir operations (Berris 1996).

Another network flow model was developed many years later by Israel (Israel 1996). The model was developed using the COE's Hydrologic Engineering Center Prescriptive Reservoir Model (HEC-PRM). The model used a linear programming algorithm to optimize the use of Truckee River water. The model operated the Truckee system based on a set of prioritized penalties, and the study suggested a unique approach to developing penalty values (Israel and Lund 1999). The approach was later criticized due to its computational intensity (Labadie and Baldo 2001).

A monthly-mass balance model was created by Reclamation in 1975. The FORTRAN-coded model was initially developed to help provide modeling support for reservoir and canal operations related to the Newland Project Operating Criteria And Procedures (OCAP), a set of procedures promulgated by Reclamation as the federal regulation for the operation of the Newlands Project, including Derby Dam, the Truckee Canal and Lahontan Reservoir. This model became known as the "BOR Model". The model was later modified during negotiations that resulted in the Preliminary Settlement Agreement (PSA), which was a precursor to the TROA. The modified model was developed to simulate various scenarios for use in the negotiations, and became known as the “Negotiation Model”. Both the Negotiation Model and the BOR Model were operations models that were capable of simulating reservoir operations as well as water accounting. The models did not have optimization capabilities. The Negotiation Model was later developed into the Truckee River Operations Model (TROM) for use on the WQSA Environmental Impact Statement (EIS) (United States Department of the Interior 2002) and the TROA EIS (USDOI 2008). Technical representatives to the PSA negotiations generally agreed that a daily or hourly physical simulation model would be necessary to fully implement the provisions set forth in the PSA (Cobb, Olson et al. 1990; Berris 1996). Others have asserted that any new model would need to be better documented to ensure its integrity (Pratt 1997).

The USGS developed a daily flow-routing model using the well documented HSPF framework as a first step towards implementing the TROA (Berris 1996). The model was later modified to include reservoir operations using conditional “if-then” logic statements (Berris, Hess et al. 1996). The model was developed to simulate existing conditions at the time of development, as well as potential conditions under the provisions of both the TROA and the WQSA (Bohman 1996; Berris, Hess et al. 2001). The model was designed to interact with a scenario generation program known as GENSCN to test and compare a variety of operating scenarios (Bohman, Berris et al. 1995; Berris, Hess et al. 2001), as well as with a precipitation-runoff model for the generation of input data (Jeton 2000).

Following the completion of the studies by the USGS, Reclamation, in conjunction with the TROA Implementation Coordinator’s Office, began development of a hydrologic model and decision support system to fully implement the TROA. The system was also developed to assist with operational forecasting and administration of current basin operating procedures prior to the implementation of TROA. The system was designed to provide operational forecasts to basin stakeholders during the spring runoff period. The system was also intended to provide current water accounting calculations for the Federal Watermaster's Office (FWM). The system is a combination of three distinct daily models, each developed with the RiverWare modeling framework (Zagona 2001), working in conjunction with a near real-time data collection utility developed by the TROA Implementation Coordinator's Office. The three models were designed to provide current accounting calculations, basin naturalized streamflow forecasts, and regulated streamflow simulations, with each model carrying out one portion of the operation. The system is currently being reconfigured for the full implementation of the TROA (Rieker, Coors et al. 2005; Boyer 2006; Boyer 2006; Boyer 2006; Coors 2006; Coors 2006; Coors 2006; Mann 2006; Mann 2006; Rieker 2006; Rieker 2006; Scott 2006). The system was utilized in the previously described water quality study by Neumann to model the use of WQSA purchased water (Neumann 2001; Neumann 2006). The system is important to this study due to the fact that the system is intended to implement the TROA, which means that it will also likely be used to implement the WQSA.

4.5 – Development of the Truckee River Task Environment

In order for an agent to be able to learn how to act from experience, a task environment must be specified and provided to the agent. The task environment may be either a real environment that the agent may act upon, using actuators based on state and reward information received from sensors of the environment, or it may be a simulation of a real environment. For the case study that is the focus of this research, the real environment that the agent is expected to act upon in the future is the Truckee River reservoir system. Due to the fact that experience in the real environment would likely cause highly undesirable outcomes in the training and exploratory phases, a simulated version of the task environment was used for training and exploration by the agent.

The task environment (real or simulated) that the Truckee River agent interacts with is one of the most complex as defined by the classifications for a task environment outlined in the previous chapter. The environment is partially observable rather than fully observable; only key variables are provided to the agent that may not provide it with all the information necessary to know how the state of the river system will evolve from one day to the next. A full specification of the environment would potentially involve hundreds of state variables, including, but not limited to, reservoir storage values, individual water owner account values within each reservoir, streamflows, forecasted streamflows, temperatures, forecasted temperatures, current operating policies, and actions by other water users of the system. The only information provided to the agent includes the status of its own water account in the system's reservoirs, as well as key state variables pertaining to the quality of water in the river.

The environment is stochastic rather than deterministic; the values of a variety of variables such as reservoir inflow, streamflow, temperature, and water quality for the system do not change in a deterministic manner. The environment is sequential rather than episodic; i.e., there are no "stopping points" or episodes that signal termination of the task being conducted by the agent. The agent attempts to improve water quality in the river system continuously by maximizing the immediate as well as longer-term performance. The environment is dynamic rather than static. The agent must determine the action continuously at each timestep. In addition, water availability and streamflow in the system change from year to year, sometimes passing through longer periods of abundant supply or drought. There is the potential for long-term non-stationarity due to the effects of climate change. The environment is continuous, rather than discrete. Essentially all variables in the environment, both those observed by the agent and those hidden from the agent, have an infinitely broad spectrum of potential values.

Finally, the environment contains multiple agents, rather than a single agent. The focus of this research is on the agent that will be operating a certain portion of the stored water in the Truckee River reservoir system that is dedicated to improvement of water quality. Many other agents exist which control other segments of the stored Truckee River water, as well as components of the river system, with each agent assigned a unique objective. Some of the agents are cooperative, such as the agent controlling water for the recovery of threatened and endangered fish species in the river which may take actions that provide an ancillary benefit to water quality. Some of the agents are competitive, such as the case of the agent controlling transbasin diversions out of the river at Derby Dam, which will cause an increased need to augment the flow of the lower river flowing to Pyramid Lake for both habitat and water quality. Some of the agents have the ability to damage the interests of other agents, even when those agents are often cooperative. An example of this would be if the water quality agent (the focus of this research) attempts to increase flows for the benefit of water quality at a time when the agent controlling flows for fish species recovery is attempting to suppress flows which would initiate fish spawning behaviors. If the water quality augmentation flows result in a fish spawning run, the fish recovery agent is forced to use its stored water to complete the undesired spawning run over a much longer time frame than that which the water quality agent had planned to augment flows, thereby potentially reducing the stored water available to conduct a fish spawning run at a more desirable time. In total, it is estimated that half a dozen or more agents may be acting on the system at any given time.

4.5.1 – Development of Truckee Hydrologic and Reservoir Operations Model

4.5.1.1 – Model Structure

For the training of the agent, a simulated version of the Truckee River hydrologic and reservoir operating environment was used. The model is a river and reservoir operations model that simulates the Truckee River system as a series of nodes and links. The model was created within the RiverWare modeling and decision support framework (Zagona 2001). The modeling system operates on a daily timestep, using a combination of basic hydrologic methods for river and reservoir routing, and a set of operating "rules" to simulate reservoir operations. The system was originally developed as a single year accounting and operational forecasting model by a team of engineers and hydrologists from the Reclamation as well as the TROA Implementation Coordinator's Office (Rieker, Coors et al. 2005; Boyer 2006; Boyer 2006; Boyer 2006; Coors 2006; Coors 2006; Coors 2006; Mann 2006; Mann 2006; Rieker 2006; Rieker 2006; Scott 2006). The Truckee River model has been converted to also function as a long-term planning model, with the capability of simulating several decades of reservoir operations in a single run. The model rules simulate current operating policy, and all major regulations, decrees, and policies currently governing the river system are included in the model ruleset. The functions of the various agents currently operating the river system are carried out through the various rules and guidelines embedded in the model, with the exception of the functions of the water quality agent that are the focus of this research. The model has been programmed with the ability at each timestep to output any number of state variables to an external process, and wait for a response from that external process before continuing with the simulation. In this manner, the water quality agent can serve as the external process, receiving information on the state variables, and returning an action to the simulated environment.

For this study, the capability to store WQSA water was introduced to the model using storage mechanisms originally proposed by Neumann (Neumann 2001). These mechanisms had historically been included in the model, but were removed after completion of that study since WQSA water was not as yet stored in the actual environment. The mechanisms proposed by Neumann were generally reintroduced to the model, with minor changes to address structural changes made to the hydrologic and reservoir operations model since the time the Neumann mechanisms existed in the model, and also to address a more current understanding of how the storage of WQSA water will be implemented in actual practice. The basic mechanism for storage begins with the transfer of water from the general water supply in the river to a specific type of water that is required to remain in the river all of the way to Pyramid Lake as instream flow. The amount of water not released from upstream reservoirs for the purpose of augmenting flows for fish recovery is then converted to stored WQSA water in the upstream reservoirs for later use by the water quality flow augmentation agent. This mechanism forms a type of exchange between the fish recovery water and the WQSA water. As an alternative, the model allows for the optional conversion of fish recovery water in the reservoirs to stored WQSA water without the requirement of an exchange, providing for the storage of the entire amount of purchased water each year.

Neumann proposed an exchange of water between Boca and Stampede reservoirs to account for the potential release of unacceptably warm water from Boca Reservoir when the reservoir is at particularly low levels in the late summer (Neumann 2001). Though a valuable finding from that study, this exchange mechanism was not reintroduced to the model primarily because of the increased complexity, as well as the limited impact on optimal release policies of the WQSA stored water.

4.5.1.2 – Model Data

Input data to the hydrologic and reservoir operations environment simulation model generally consists of initial values for each of the reservoirs and water accounts within the reservoirs, as well as hydrologic inflows to the reservoirs and uncontrolled river reaches. Approximately thirty years of daily historic hydrologic data was originally developed (Rieker 2005), and later extended by others to encompass over fifty years of historic hydrology on the river system (Fritchel 2009). For operation of the planning model, a spreadsheet system was developed by Coors (Coors 2010) to allow user input of hydrologic exceedence probabilities for each year in a multiple year run, and generate synthetic hydrologic sequences as input to the model. The sequences are generated by averaging three similar-year hydrographs for each year in the sequence based on April to July runoff volume for the river, and then fitting the overall hydrograph to match the total runoff volume associated with the user-entered hydrologic exceedence probability. Alternatively, the spreadsheet system allows users to enter actual historic year values and generate hydrologic sequences based on historic data measurements. These sequences may either replicate an actual historic period by entering successive year values, or be used to create a synthetic sequence of data based on a selection of non-successive year values.

For agent training and testing in the Truckee River case study, two primary datasets were developed and utilized. The first dataset consisted of the approximately 30 year period of historic data from October 1, 1970 through December 31, 1999. These data are referred to as the “30-year historic dataset.” It represents the best available historic hydrologic data for a timespan found to be appropriate for use within the limits of the physical memory of the computers used for training and testing of the Truckee River case study, when utilizing multiple instances of the simulation model as discussed in Section 3.2.5. The dataset covers the period of time beginning at the completion of the last major reservoir on the Truckee River (Stampede Reservoir), and includes a wide variety of extreme events experienced in the Truckee River basin. It includes a series of significant drought years (early 1990's), many significantly wet years (early 1980's late 1990's), and one of the largest floods on the river in the past century (1997).

The second dataset constructed is a synthetic dataset composed of approximately 30 years of hydrologic data, generally consisting of a repeating 4 year cycle. The cycle includes one significantly wet year associated with a hydrologic exceedence probability of approximately 10 percent, followed by two moderately dry years and a final increasingly dry year. The first two dry years represented years with a 70 percent exceedence probability, and the final dry year represented an 80 percent exceedence probability. The intent of the dataset was to provide cycles of four year periods where the first year would provide the ability to fill the Truckee River reservoirs and store water for water quality purposes during a time when few or no downstream water quality issues would arise due to naturally wet conditions throughout the river system. The second through fourth years were designed to introduce conditions with increasingly poor downstream water quality on the river, whereby the water quality agent would be required to develop longer-term strategies to ensure the conservation of enough stored water to prevent more serious water quality issues over the course of many years of simulation. A limited number of other datasets were developed for various testing purposes, generally with a similar intent and design as the primary synthetic hydrology dataset.

4.5.2 – Development of Truckee Water Quality Model

To simulate water quality in the Truckee River task environment, several water quality models were considered. An initial effort was made to link the TrHSPF water quality model under development by the Cities of Reno and Sparks to the Truckee River hydrologic and reservoir operations model using data connections between the two models on a daily timestep. This was necessary due to the dependence of the simulated water quality on the flow rate in the river. The TrHSPF model was initially selected to serve as the water quality simulation model for the system because it was the newest and most current water quality model of the river, had ongoing developmental support by the cities, and had the ability to model water quality for the entire reach of river from the California-Nevada state line to Pyramid Lake, which is the area of concern for this study.

Difficulties were encountered linking the TrHSPF model to the Truckee River hydrologic and reservoir operations model due to several issues. The first major issue was that the model timesteps are different, with the TrHSPF model operating on an hourly timestep and the reservoir operations model operating on a daily timestep. This was overcome by introducing an intermediate data processing step into the model linkage to provide the water quality model with the average daily flow on an hourly timestep. The second major issue involved the fact that the TrHSPF model utilized an older, generally unsupported database format known as “ANNIE” (Lumb, Kittle et al. 1990) as its external data storage mechanism, making it incompatible with most current data transfer mechanisms. To overcome this issue, the model was modified to interface with a newer data storage mechanism. All existing input data was converted into the newer mechanism, and the models were successfully linked together. The final major issue with the TrHSPF model was its need for a large number of water quality data inputs, many of which were dependent on flow, weather, and a variety of other hydrometerological conditions.

The database of inputs had been developed for a historic period of time significantly shorter than the available historic input data for the hydrologic and reservoir operations model. Attempts were made to explore ways to extend the database to a longer period of time, but the effort proved intractable. Additionally, for the water quality agent to be able to perform in a near-real time control situation as intended, this large number of water quality inputs would need to be available to the agent in a near-real time and ongoing basis. Many of the inputs were of a type that would not support that need. For these reasons, the effort to link the TrHSPF model to the hydrologic and reservoir operations model was terminated. In addition, the effort as a whole became a large undertaking that diverted energy away from the true goal of the research, which was the development of the agent itself.

4.5.2.1 – Model Structure

In light of the difficulties encountered in the development of a complex water quality environment model, a more straightforward approach to water quality modeling was taken. This involved the use of the temperature prediction water quality model employed by Neumann in the predecessor study to this effort (Neumann, Rajagopalan et al. 2003). The temperature model was developed to predict the water temperature of the Truckee River near Reno, Nevada. This was done to address one key component of the water quality issues facing the Truckee River that was intended to be solved through the use of the WQSA stored water. The model was based on a detailed regression analysis relating river temperature to both river flow and maximum daily air temperature in Reno, and was found to produce reasonable results (Neumann, Rajagopalan et al. 2003). The model was integrated directly into the Truckee River hydrologic and reservoir operations model as a set of rules that utilized the developed regression parameters to predict the water temperature in the river near Reno on a daily basis.

4.5.2.2 – Model Data

Model input data utilized by the temperature prediction water quality model includes only two variables; the estimated maximum daily air temperature in Reno, and the flow passing the California-Nevada state line. The flow data are generated by the hydrologic and reservoir operations model as simulated flow for each timestep, with the estimated maximum daily temperatures in Reno obtained from the historic record. For this analysis, a static dataset of daily air temperatures was utilized. The data were obtained from the National Oceanic and Atmospheric Administration’s (NOAA) National Climatic Data Center (NCDC). Temperature data obtained from the station at the Reno Weather Bureau Office (WBO) were utilized for dates prior to 1973. Temperature data obtained from the station at the Reno Airport were utilized from 1973 to present. The dataset was made up of approximately 30 years of data, beginning in 1970 and ending in 1999 to match the time period of the 30-year historic hydrologic dataset used in the hydrologic and reservoir operations environment simulation model. For ease of model operation, any data points in the dataset representing temperatures colder than 0° C were replaced with a data point at 0° C since these data points had little bearing on the model simulation and results. Though it is recognized that temperature and hydrology data may be related, for this study the two variables were assumed to be independent, and the temperature data was not rearranged or otherwise reprocessed in a manner similar to the processes used to create synthetic or non-historic hydrologic datasets as discussed in Section 4.6.1.2.

4.5.2.3 – Temperature Targets and Water Quality Violations

As discussed above, water temperature was selected as the primary focus of water quality for the Truckee River case study. For the purposes of determining the desired water quality in the river, the needs of the two fish species listed as threatened or endangered under the Endangered Species Act were considered. These fish species were the cui-ui and Lahontan cutthroat trout (LCT). Of the two, the cui-ui generally only use the river for spawning during spring runoff, when temperature is not a major concern, and then return to Pyramid Lake after spawning (SCOPPETTONE, COLEMAN et al. 1983). For this reason, primary consideration was given to the needs of the LCT.

A system of target temperatures and water quality “violations” in the Truckee River were utilized by Neumann in the earlier study of water quality issues on the Truckee River (Neumann 2001). Those targets were based on earlier studies and interactions with water quality specialists. Under that system, the primary temperature target for the Truckee River flowing through Reno was considered to be 22° C. This was also referred to as the “preferred maximum”, and though too high for spawning or juvenile fish rearing, was considered to be the upper bound of temperatures suitable for extended habitation by adult fish. A secondary target was established, which allowed for temperatures to temporarily exceed the primary target for up to four days, but not exceed 23° C. This secondary target was considered to be stressful but not fatal to the fish, as long as the duration in that temperature range did not exceed four days. The temperature of 23° C was thus considered to be a “chronic maximum”. A tertiary target was also established, which provided for temperature fluctuations of up to 24° C for one day or less. This was considered the “absolute maximum”. Temperatures exceeding any of these thresholds or durations were considered to be “violations”, not necessarily in a legal sense, but for the purposes of performance measurement. For the purposes of this study, the target temperature and violation system developed by Neumann as described above was adopted and utilized.

4.5.3 – Development of Reward Function

The reward function has been referred to as one of the most critical elements of the environment for the development of a successful agent. The development of the reward function provided to the water quality agent focused on the operation of the river for the improvement of water quality. Two reward functions were developed for the Truckee River case study. The first reward function was designed to focus on the immediate issue of selecting daily release actions to improve downstream water quality on individual days when water quality issues are predicted to arise. The second reward function was designed to focus on the longer-term issue of preventing water quality issues over the course of a number of years. This reward function was intended to be used in association with the selection of seasonal or annual release policies that would potentially allow certain lower-level water quality issues to occur, but ensure that sufficient storage was maintained to be able to mitigate more significant water quality issues during a drought period.

The first reward function, focusing on the immediate release action selection, incorporates three distinct goals for the agent, including 1) improvement of river temperature when the temperature is above the preferred maximum temperature for the threatened and endangered species of fish in the river, 2) conservation of stored WQSA water when the water is not needed for improvement of water temperature in the river, and 3) acting on the environment within the domain of admissible actions. To fully encapsulate these goals into a reward function, the reward function was developed to be inversely proportional to the quality of the action taken by the agent, focusing on punishment values for not meeting the goals. The reward function was designed to be minimized to achieve improved actions.

To evaluate each of the agent's actions against the first and primary goal of the agent (maintenance of river temperature below the preferred maximum), Equation 4.1 was developed:

$$r_x(s,a) = \begin{cases} \left[b_x * \left(t_{Reno} - t_{target}\right)\right]^{c_x} & if\ t_{Reno} > t_{target} \\ 0 & if\ t_{Reno} \leq t_{target} \end{cases} \qquad \text{(Eqn 4.1)}$$

where:

$b_x = Reward\ function\ multiplier$

$c_x = Reward\ function\ exponent$

$t_{Reno} = River\ water\ temperature\ in\ Reno$

$t_{target} = Target\ maximum\ water\ temperature$

$s = Vector\ of\ state\ values$

$a = Action\ value$

The vector of state values contains two primary values; the amount of water stored for the purpose of augmenting flow for water quality improvements, and the downstream water quality of the river. The action value is the amount of water released from storage for the specific purpose of augmenting flow for water quality improvement in addition to water already scheduled for release for other purposes. The user is permitted to adjust the multiplier and exponent factors in the equation to provide better feedback to the agent regarding the value of its actions.

To evaluate each of the agent's actions against the second goal of the agent (conservation of stored water), Equation 4.2 was developed:

$$r_y(s,a) = \begin{cases} \left[b_y * \left(t_{target} - t_{Reno}\right)\right]^{c_y} & if\ t_{Reno} \leq\ t_{target}\ and\ a_{t-1} > 0 \\ 0 & if\ t_{Reno} >\ t_{target} \end{cases} \quad \text{(Eqn 4.2)}$$

The user is permitted to adjust the multiplier and exponent factors to provide better feedback to the agent regarding the value of its actions. By maintaining a multiplier and/or exponent factor less than that used in Equation 4.1, achievement of this goal can be given a lower priority than the first.

To evaluate each of the agent's actions against the third and final goal of the agent (actions within the admissible domain), Equation 4.3 was developed:

$$r_z(s,a) = \begin{cases} \left[b_z * \left(a_{t-1} - max\left\{\begin{matrix} sto_{t-1} \\ a_{t-1} - d_z \end{matrix}\right\}\right)\right]^{c_z} & if\ a_{t-1} >\ sto_{t-1} \\ 0 & if\ a_{t-1} \leq\ sto_{t-1} \end{cases} \quad \text{(Eqn 4.3)}$$

where:

$sto_{t-1} = Storage\ available\ to\ release\ (converted\ to\ flowrate)$

$d_z = Maximum\ reward\ limiter$

This equation focuses only on taking actions that do not attempt to release water that is not currently held in storage. In other words, it attempts to prevent the agent from directing the release of water that it does not have available. The equation does not focus on preventing the agent from releasing more water than the reservoir is physically capable of releasing, because that amount will be highly dynamic and depend on a multitude of other factors beyond the control of the agent. The user is permitted to adjust the multiplier, exponent, and maximum reward limiter factors to provide better feedback to the agent regarding the value of its actions. The multiplier serves as a scaling factor to make this evaluation commensurate to that obtained from equations 4.1 and 4.2. The maximum reward limiter acts as a limitation on the upper value of this particular reward equation, to prevent the result of the equation from growing to a point where it becomes a higher priority than Equations 4.1 and 4.2 for the agent, or negatively impacts the agent's learning ability.

The final reward function is the sum of all three equations presented above, implemented as Equation 4.4:

$$r(s,a) = r_x(s,a) + r_y(s,a) + r_z(s,a) \qquad \text{(Eqn 4.4)}$$

The second overall reward function, focusing on the seasonal or annual policy selection action, incorporates only one goal for the agent, which is the improvement of river temperature when the temperature is above the preferred maximum temperature for the survival of the threatened and endangered species of fish in the river, as viewed across an entire season or year. Because the agent is selecting an overall policy for the season or year, the agent is no longer concerned with the day-to-day release actions, and it is assumed that the range of potential policies presented to the agent will be crafted in a way that ensures that the water will not be used on days when it is not needed to carry out the seasonal policy, or in a way that attempts to release more water than the agent has available. Similar to the first overall reward function shown in Equation 4.4, the seasonal reward function was developed to be inversely proportional to the quality of the action taken by the agent, requiring minimization to achieve improved actions.

To evaluate each of the agent's policy selection actions against the goal of the agent (overall seasonal/annual maintenance of river temperature below the preferred maximum), Equation 4.5 was developed:

$$r_x(s,a) = \begin{cases} \sum_{t_{target}}^{t_{max}} \#Days_{t_{Reno}} * \left[b_x * \left(t_{Reno} - t_{target}\right)\right]^{c_x} & if\ t_{Reno} > t_{target} \\ 0 & if\ t_{Reno} \leq t_{target} \end{cases}$$

(Eqn 4.5)

where:

$\#Days_{t_{Reno}} = Number\ of\ days\ during\ year\ with\ water\ temperature\ t_{Reno}$

$t_{max} = Maximum\ experienced\ water\ temperature$

The remaining variables and coefficients are the same as those shown for Equation 4.1. Equation 4.5 essentially multiplies the sum of the number of days during a season or year at a particular temperature by a penalty function for that particular temperature. The penalty function for the particular temperature increases with increasing temperature. For this reason, the reward function is designed to motivate the agent to select policies that eliminate downstream temperature issues, but in the event that all issues cannot be eliminated, the agent is more motivated to select a policy action that reduces higher temperature extremes by allowing more temperature violations to occur in the lower temperature ranges.

4.5.4 – Testing/Validation of Environment Model

Initial completion of the Truckee River hydrologic and reservoir operations model occurred in 2004, and though ongoing development has continued to occur, the results of simulation modeling have been presented at monthly forecasting meetings to basin stakeholders since that time. Testing and calibration of the model has occurred iteratively through feedback from the members of that group. The State of California's Department of Water Resources conducted several annual reviews of the results of the model output, showing that it performed well as compared to actual conditions, even when used in the forecasting mode (Nelson 2006; Rieker 2006). The Truckee River temperature water quality model was validated through the work of Neumann (Neumann, Rajagopalan et al. 2003).

4.6 – Development of the Truckee River Rational Agent

For the Truckee River case study, two rational agents were developed, each with similar but distinct goals. The first agent was designed to provide daily release action decisions for the WQSA water held in storage, based on the current amount of storage available and the predicted downstream water temperature for that day. This agent is referred to as the “short-term agent.” The second agent was designed to provide decisions with respect to the overall release policy to be used during the course of a year based on the amount of storage available and an indicator variable to help predict the overall requirements for flow augmentation for the year. This agent is referred to as the “long-term agent.” The first agent was designed to be capable of generating daily release policies based on immediate reward, while the second agent was designed to address longer-term issues by developing an overall policy for the use of the water that would ensure proper storage to mitigate broader issues over a multi-year period.

4.6.1 – State and Action Representation

For the short-term Truckee River rational agent, the state information received from the environment includes the total storage of WQSA water that the agent currently has available to utilize, and the predicted temperature of the Truckee River water in Reno for the current timestep (day) if no additional releases are made beyond what was released for other purposes. The agent is also provided the reward calculation from the action taken by the agent in the previous timestep. The action taken by the agent on the environment is a direction to the environment regarding the amount of WQSA water to release into the river. Though the Truckee River agent operates in a multi-reservoir environment, it does not require specific information on the WQSA water stored in each reservoir, nor does it require actions that determine which reservoir to release the WQSA water from, since the WQSA water is stored and released from the multiple reservoirs in a simple priority order that is automatically determined by the environment model. This mimics the real-world priority system in place for storage and release from most water accounts in the Truckee reservoir system.

Tests were conducted using various combinations of the state representation provided to the agent. One set of testing involved just the actual values for WQSA storage and predicted river temperature. Another set of tests used those same state dimensions, but applied the “feature extraction” concepts discussed in Section 3.2.5.1. The feature extraction applied to those state variables included partitioning the WQSA storage value into 16 possible categories, as summarized in Table 4.5 below. Actual values of the WQSA storage volume were associated with the category corresponding to the first storage value in the table less than the actual value, thus producing a “round down” operation on the actual values. The categories were designed to provide a higher level of discretization of the storage variable in the lower range of potential storage values, where the agent’s actions were likely to have the potential to deplete storage or attempt an over-release. The goal behind that partitioning was to allow the agent to learn exactly what releases can be made if remaining storage is limited. With higher storage values, each category spans a broader range of potential storages, since the agent is less likely to be concerned with over-release or immediate depletion of storage in this range. In order to keep the overall process computationally manageable, particularly for the tabular implementation, the agent’s possible actions were limited to the actions that would most likely change the predicted downstream temperature by increments of 0.5° C. The volume of water necessary to facilitate those incremental releases is reflected in the categories used for the lower range of the storage discretization.

Table 4.5 – Short-term agent WQSA storage discretization

WQSA Storage	Category	Maximum Possible Release
MCM (AF)		cms (cfs)
0 (0)	0	0 (0)
0.09 (70)	1	1 (35)
0.17 (140)	2	2 (71)
0.26 (210)	3	3 (106)
0.35 (280)	4	4 (141)
0.43 (350)	5	5 (176)
0.52 (420)	6	6 (212)
0.6 (490)	7	7 (247)
0.69 (560)	8	8 (282)
1.38 (1,120)	9	16 (565)
2.76 (2,240)	10	32 (1,129)
5.53 (4,480)	11	64 (2,259)
8.29 (6,720)	12	96 (3,388)
11.05 (8,960)	13	128 (4,517)
13.81 (11,200)	14	160 (5,647)
16.58 (13,440)	15	192 (6,776)

The feature extraction concept was also applied to the state variable representing the predicted downstream temperature in the Truckee River. The predicted temperature was discretized into 13 categories, representing the temperature range of highest concern to the agent. That discretization is shown in Table 4.6. Temperature values were associated with the category corresponding to the first temperature value in the table greater than the actual value, thus producing a "round up" operation on the actual values to find the appropriate category.

Table 4.6 – Temperature discretization

Temperature (C)	Category
<=21	0
21.5	1
22	2
22.5	3
23	4
23.5	5
24	6
24.5	7
25	8
25.5	9
26	10
27	11
>27	12

For the long-term Truckee River rational agent, the state information received from the environment includes the total amount of dedicated WQSA water in storage as of June 1 of each year, as well as the predicted peak storage for Lake Tahoe for the year. Those two variables generally provide the agent with sufficient information to make a decision on the manner in which to use the WQSA water for the year. The reason for the inclusion of the predicted peak storage for Lake Tahoe is that Lake Tahoe represents the vast majority of the stored water in the overall Truckee reservoir system. Most temperature violations predicted by the models occur in years when Lake Tahoe's storage is depleted, resulting in extremely little flow in the lower Truckee River mostly attributable to groundwater seepage or return flows. The inclusion of this variable was intended to provide the agent with a representation of the likelihood of upcoming drought and thus a need for large amounts of stored water to prevent water quality issues. The peak storage of Lake Tahoe is fairly easily predicted by June 1 of each year, when

WQSA stored water releases would generally begin to occur due to increasing summertime water temperatures. The variable associated with the predicted peak storage in Lake Tahoe was discretized into broad ranges of possible storage values, as shown in Table 4.7. As with the WQSA storage discretization, actual values of Lake Tahoe storage volume were associated with the category corresponding to the first storage value in the table less than the actual value, thus producing a "round down" operation on the actual values.

Table 4.7 – Lake Tahoe storage discretization

WQSA Storage	Category
MCM (AF)	
< 0 (0)	0
123.35 (100,000)	1
246.7 (200,000)	2
> 246.7 (200,000)	3

For the long-term agent, the WQSA storage variable was discretized differently than for the short-term agent, due to the fact that the long-term agent is only interested in the overall amount of storage available to it. By providing larger categories of discretization, the agent is able to better determine the effects of its actions, whether the action is to select a policy resulting in large volumes being released during the course of the year, or in additional storage being gained. The discretization of the WQSA storage variable for the long-term agent is shown in Table 4.8. Similar to the short term agent, actual values of the WQSA storage volume were associated with the category corresponding to the first storage value in the table less than the actual value, thus producing a "round down" operation on the actual values.

Table 4.8 – Long-term agent WQSA storage discretization

WQSA Storage	Category
MCM (AF)	
0 (0)	0
6.17 (5,000)	1
12.33 (10,000)	2
18.5 (15,000)	3
24.67 (20,000)	4
30.84 (25,000)	5
37 (30,000)	6
43.17 (35,000)	7
49.34 (40,000)	8
61.67 (50,000)	9
74.01 (60,000)	10
86.34 (70,000)	11
> 98.68 (80,000)	12

The action provided to the environment by the long-term agent is the selection of a policy to be followed for the entire year for determination of the amount of WQSA water to be released. That policy determines the day-to-day releases of the water, depending solely on the predicted downstream temperature of the water for that day absent any releases of WQSA water. The range of policies provided to the agent are essentially the same as those developed by the short-term agent based on immediate reward, as will be shown in the proceeding chapter. Three release policies are available; those policies are designed to result in daily actions that reduce the downstream water temperature to either that associated with the preferred maximum, chronic maximum, or acute maximum as defined in Section 4.5.2.3. A fourth policy is available to the agent, which is a “no-action” policy. Under this policy, no WQSA water is released for the year, and all water is stored for future years. The four policies available to the agent are shown in Table 4.9.

Table 4.9 – Long-term agent policy selection options

Temperature	Preferred Policy	Chronic Policy	Acute Policy	Zero Release Policy
C (category)	Release, cms (cfs)	Release, cms (cfs)	Release, cms (cfs)	Release, cms (cfs)
< 21 (0)	0 (0)	0 (0)	0 (0)	0 (0)
21.5 (1)	0 (0)	0 (0)	0 (0)	0 (0)
22 (2)	0 (0)	0 (0)	0 (0)	0 (0)
22.5 (3)	1 (35)	0 (0)	0 (0)	0 (0)
23 (4)	2 (70)	0 (0)	0 (0)	0 (0)
23.5 (5)	3 (105)	1 (35)	0 (0)	0 (0)
24 (6)	4 (140)	2 (70)	0 (0)	0 (0)
24.5 (7)	5 (175)	3 (105)	1 (35)	0 (0)
25 (8)	6 (210)	4 (140)	2 (70)	0 (0)
25.5 (9)	7 (245)	5 (175)	3 (105)	0 (0)
26 (10)	8 (280)	6 (210)	4 (140)	0 (0)
27 (11)	10 (350)	7 (245)	5 (175)	0 (0)
> 27 (12)	12 (420)	8 (280)	6 (210)	0 (0)

4.6.2 – Agent Specification and Design

For the purposes of the Truckee River case study, the generalized agent specification outlined in Section 3.2.2 was utilized. The agent program for the development of the Truckee River agent was constructed for both the tabular implementation and neural network implementation (with genetic algorithm training mechanisms) as previously discussed. The learning methods applied to the Truckee River agent case are the same as those previously detailed in Section 3.2.3. For any particular simulation test, either the short-term or long-term agent is used; the two are not used simultaneously. A diagram of the Truckee River agent is shown in Figure 4.2.

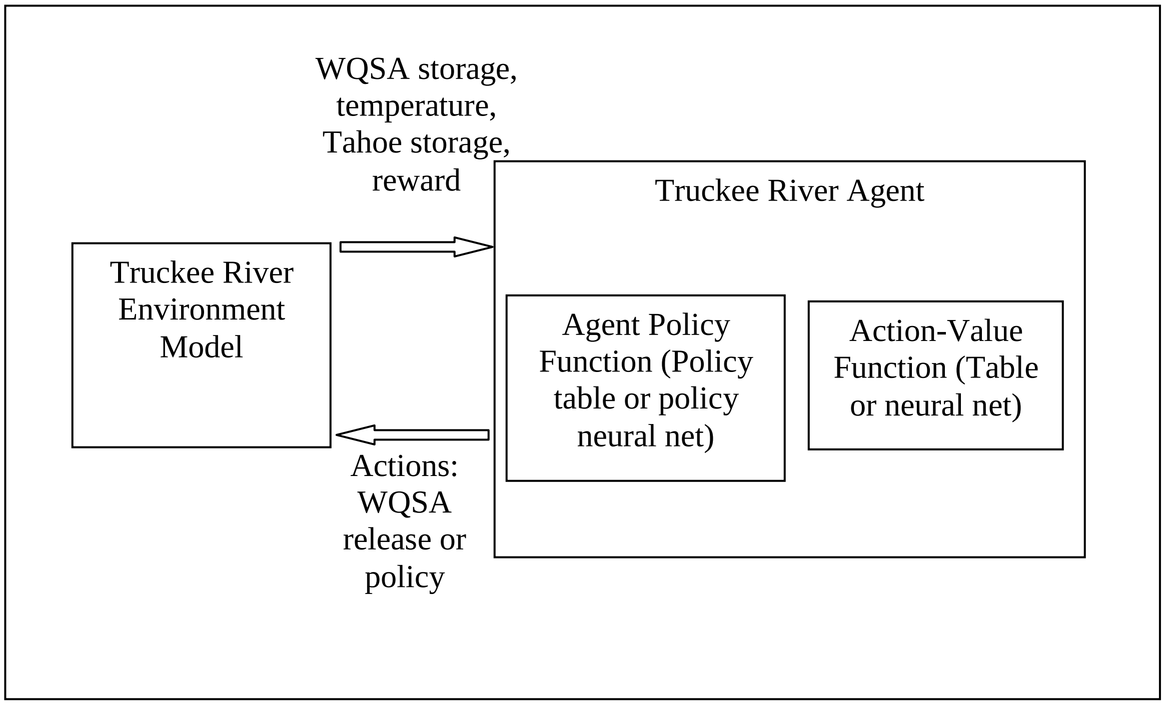

WQSA storage, temperature, Tahoe storage, reward
Truckee River Agent
Truckee River Environment Model
Agent Policy Function (Policy table or policy neural net)
Action-Value Function (Table or neural net)
Actions: WQSA release or policy

Figure 4.2 – Diagram of the Truckee River agent structure

The short-term agent for the Truckee River is designed to act on a daily basis between June 1 and December 1 of each year in a simulated model run, representing the season when essentially all water temperature issues occur. At each daily timestep, the agent receives the state information regarding the current WQSA storage available and predicted downstream water temperature absent any action, and the agent returns an action selection representing that day's intended release of WQSA water. The agent receives the reward information for the previous day for use in its action-value update calculations.

The long-term agent is designed to act on an annual basis on June 1 of each year in a simulated model run. On that timestep of the run, the agent receives the state information regarding the WQSA storage and predicted peak storage in Lake Tahoe, and returns an action selection representing the WQSA release policy to be used for the remainder of that year. The agent receives the reward information on the overall performance of the policy used in the previous year for use in its action-value update calculations.

5 – RESULTS AND ANALYSIS

The models and processes developed in Chapter 3 were utilized on the Truckee River test case discussed in Chapter 4. This chapter summarizes and analyzes the results of that application. The first section of this chapter discusses the development of release policies for the improvement of water quality on the Truckee River using both the short-term and long-term agents developed in Section 4.6. The following two sections discuss the effectiveness of many of the different technological options for applying reinforcement learning to the Truckee River case study.

5.1 – Development of Water Quality Release Policies

5.1.1 – Development of Basic Water Quality Release Policies

Testing on the Truckee River case study using the short-term agent focused on the development of daily policies for the release of the stored water specific to water quality purposes. Basic release policies were developed by applying the reward functions developed in Equations 4.1 through 4.4, and utilizing the system of discretization and feature extraction discussed in Section 4.6.1. For the development of the basic daily release policies, only the WQSA storage and predicted temperature state variables were provided to the agent, and the focus was on policies that would attempt to keep the water temperature under the preferred maximum of 22° C. It was found that these basic release policies could generally be discovered by the agent through training of the agent using many repeated runs of the simulation model using the 30-year historic hydrology. For the discovery of basic release policies that focus on the immediate improvement of water quality, a discount rate of zero was used for the update Equation 3.13. This focused the agent entirely on immediate reward and therefore the issue of immediate temperature problems.

It was determined that the amount of water eligible for conversion into storage using the exchange method described in Section 4.5.1.1 resulted in extremely little WQSA water actually being stored, due to the fact that the conditions for this type of exchange are very rare. Based on the 30-year hydrologic dataset, it was found that storage only occurred in 12 out of 30 years, and a maximum of only approximately 4.3 MCM (3,520 acre-feet) could be stored while following a policy that addresses immediate downstream water quality needs. For this reason, all subsequent modeling was conducted with the model transferring the full amount of purchased water to storage each year through conversion of that water from the water in storage dedicated to the recovery of the downstream fish species.

It was further discovered through testing based on the 30-year historic hydrology that even when WQSA water is allowed to be converted as described above, the WQSA water is not able to accrue in storage to any significant value due to the fact that it holds essentially the lowest priority for the storage rights in Stampede, Boca, and Prosser Reservoirs, where it would generally be stored. Because of this low storage priority, in years when the water is not generally needed in any significant amount to improve downstream water quality, the reservoirs where the water is stored would commonly fill to capacity, causing the WQSA water to be displaced or "spilled" by water stored for other purposes holding a higher priority storage right. Then in years when the water is needed, generally there is not enough water in storage to resolve all downstream water quality issues. Although this latter finding is similar to that made in previous studies (Neumann 2001), the period of time examined by those studies was not as extensive, and the magnitude of the spill issue and its impacts on the amount of water able to be stored for WQSA purposes were not fully known or explored in those studies. During the 30-year hydrologic test runs, it was found that generally, the WQSA water would be completely displaced in 11 out of 30 years.

As a result of the limited ability to store WQSA water, the water quality agent was generally only able to discover policies associated with lower storage volumes. This issue was magnified by the fact that the agent generally attempts to follow a greedy policy based on its current knowledge and experience while sometimes following exploratory actions according to the particular exploration algorithm in use. Both the use of a greedy policy and the occasional exploratory actions serve to reduce the agent's ability to store water, preventing it from learning policies in higher storage categories. In testing based on the 30-year historic hydrology, it was found that when following a greedy policy, storage did not exceed 25.9 MCM (21,000 acre-feet), and even if the agent attempted to constantly store all currently available WQSA water, the most it could ever store was approximately 63.0 MCM (51,100 acre-feet).

In order to more fully develop the scope of the policies learned by the agent and test the learning capabilities of the agent, the simulation models were adjusted to provide a firm amount of storage capacity dedicated to the WQSA water. The agent was able to store up to the firm amount without danger of the water being displaced by other water in the reservoir. Testing was conducted with this adjustment, and then the models were further adjusted to allow the agent to have access to any amount of water available in Stampede Reservoir that it desired, whether the water was dedicated to WQSA purposes or not. This was done to test the ability to learn policies for all conditions based on ample water supply. Under this "idealized training" condition, model runs were made which locked the storage available to the agent into each of the different storage categories identified in Table 4.5, with multiple runs per storage category, ensuring that adequate exploration could occur at all possible storage values. Access to additional water was limited to the water in Stampede reservoir because it is the primary storage reservoir for WQSA water, it is the largest of the reservoirs available to the WQSA water, and the other water in the reservoir is generally the water used for downstream fishery purposes which has similar goals to the WQSA.

The policies developed by the tests discussed are shown in Tables C.1 and C.2 in Appendix C. The policies presented are limited to the tests conducted with the full amount of WQSA water being stored through the more flexible exchange process, and with 49.3 MCM (40,000 acre-feet) of storage reserved as "firm" and not subject to spill. Tests which resulted in a state being experienced fewer than 15 times were determined to not have sufficient opportunity to form a policy, and were denoted by an asterisk (*) in the table.

An analytically derived table, showing the immediate reward policy that was expected based strictly on the amount of water required to bring predicted water temperatures below 22° C, is shown in Table C.3. This table was generated using the expected rewards that would be provided to the agent for taking actions in the various state categories listed. This overall policy is what the agent would be expected to learn through training. The similarity between the policies developed by the agent in Tables C.1 and C.2 and the analytically derived policy shown in Table C.3 illustrates that the agent was successful at learning a short-term strategy for water quality improvement, and that this strategy was similar to what could have been expected based on the release required to bring water temperatures down to the 22° C preferred target.

The rate at which basic water quality release policies were able to stabilize to the values shown in Tables C.1 and C.2 varied depending primarily on the amount of water available to the agent. "Idealized training" runs where the agent was provided access to alternate water supplies and the storage was locked into the various storage categories generally converged very nearly to the final solution by the end of the first set of runs of a 30 year simulation for each storage category. Simulation model runs made without the benefit of the additional water generally did not converge as quickly, and even though additional testing would possibly introduce enough exploration opportunities at some portions of the state and action space to add a policy data point, the runs were terminated once it was clear that the states visited most often had converged to a reasonable degree. However, that form of training generally achieved better final convergence to the analytically derived policy than the idealized training. Convergence results for the runs discussed in Section 5.1.1 are summarized in Figures 5.1 and 5.2. Figure 5.1 shows the agent's behavior under both normal and idealized training, and illustrates the speed at which the agent's policy approaches the final policy in Tables C.1 and C.2. Figure 5.2 shows how the agent's policy approaches the analytical policy in Table C.3 for both types of training. By reviewing both figures, it is noted that under normal training conditions, the agent's policy continues to shift at a steady rate once stabilizing near the analytically derived policy.

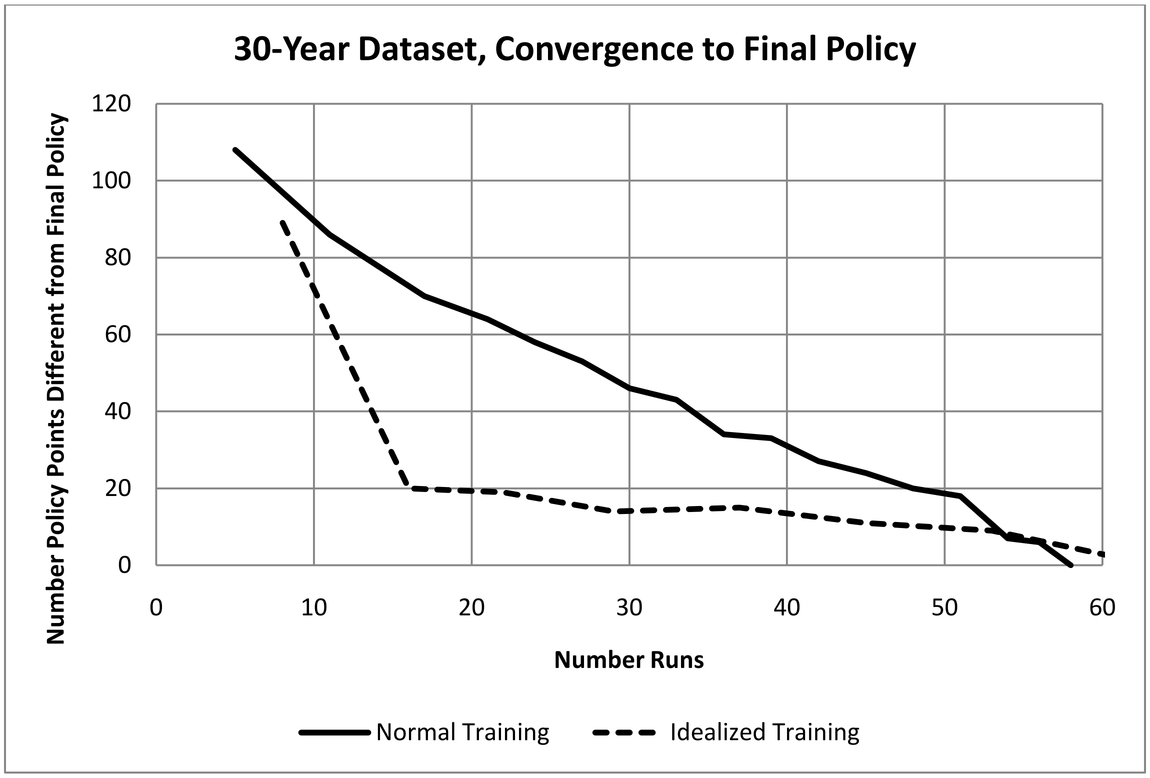


Figure 5.1 – Short-term agent training, convergence to final policy

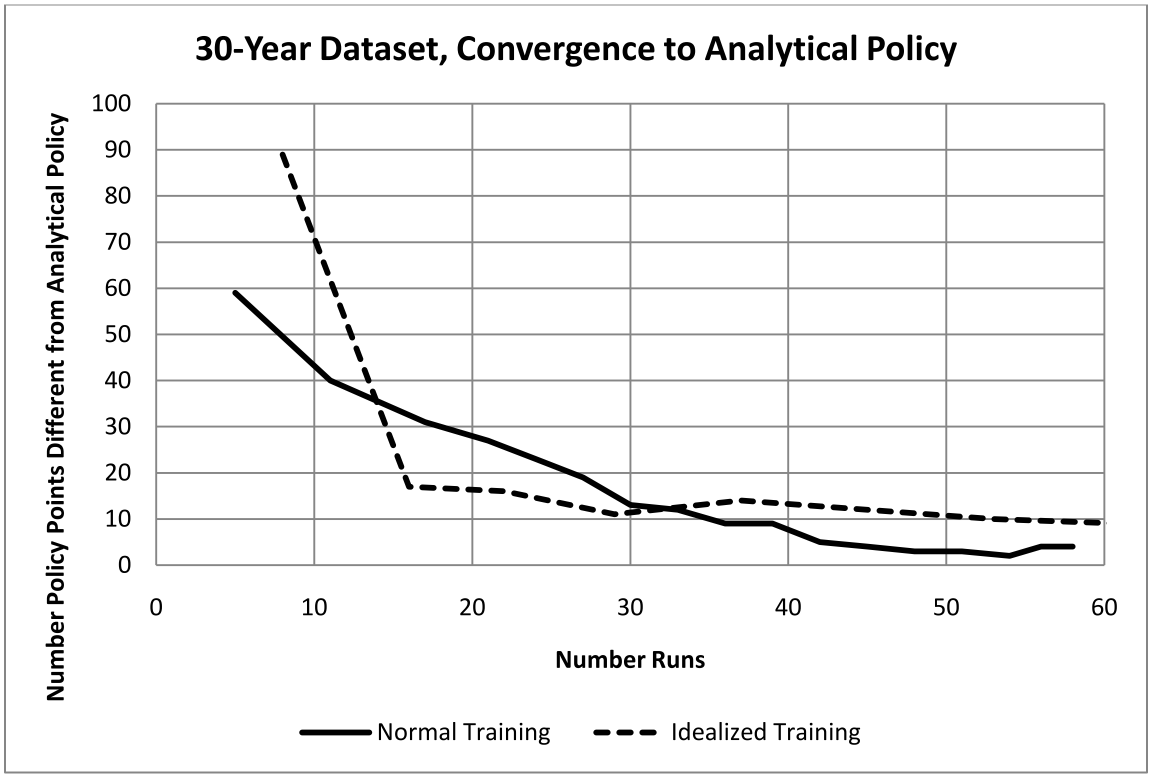


Figure 5.2 – Short-term agent training, convergence to analytical policy

Simulation model evaluations were conducted using the policies shown in Tables C.1 through C.3 using the same 30-year historic hydrology. The policies were evaluated based on the number of days resulting in predicted river temperatures at the various temperature categories identified in Section 4.6.1, and the number of violations occurring during the run as defined in Section 4.5.2.3. The results of the evaluation of those policies are shown in Table 5.1. For the evaluations, the agent was provided with 98.7 MCM (80,000 acre-feet) of firm storage, and all WQSA water was made available each year through the flexible exchange previously described. For comparison, the results for "baseline" runs with no WQSA water used are shown, as well as runs made with the strict exchange requirements in place, and runs without the benefit of any firm storage. The latter runs were made using the analytically derived policy. Values for the analytically derived policy were used in place of the missing values in Table C.1 where data was not developed by the agent due to low experience and exploration opportunities.

Table 5.1 – Evaluations of short-term agent policies over 30-year historic hydrology

		Analytically Derived Policy			Agent Policy	
Water Quality Variable	Baseline	No firm storage, strict exchange	No firm storage, full exchange	Firm storage, full exchange	Normal Training	Idealized Training
Chronic Violations	10	10	3	4	4	4
Acute Violations	39	38	16	6	6	6
Absolute Violations	416	415	329	216	214	212
22-22.5° C (# days)	120	110	99	163	159	162
22.5-23° C (# days)	92	90	29	8	7	7
23-23.5° C (# days)	67	67	42	11	12	12
23.5-24° C (# days)	85	84	39	18	21	23
24-24.5° C (# days)	74	70	51	33	33	30
24.5-25° C (# days)	53	56	46	34	31	33
25-25.5° C (# days)	73	68	52	34	35	33
25.5-26° C (# days)	82	81	55	35	35	37
26-27° C (# days)	105	111	98	54	54	53
>27° C (# days)	29	29	27	26	26	26

Table 5.1 shows that the policies learned by the agent produced results highly similar to those produced by the analytically derived policy, indicating successful agent learning. The table shows the value of the use of WQSA water through large reductions in the number of days of elevated water temperature on the river. The table also shows the importance of added flexibility in the way that water is exchanged into storage, as well as the value of having firm storage that is not subject to spill. Based on the testing conducted, without these added flexibilities in the management of the water, the ability to store and later release WQSA water has very little value, as shown in the table.

It was found that the majority of the downstream temperature issues during the 30-year historical hydrologic simulations occur during the time in the historic record associated with the drought experienced in the early 1990's. That drought causes complete depletion of all reservoirs in the Truckee reservoir system, and results in a large number of days with high water temperature and water quality violations. The results from the runs based on the 30-year historical hydrologic data illustrate the large number of days with unavoidably high temperatures and water quality violations when following an immediate reward policy, as seen in Table 5.1.

5.1.2 – Development of Improved Water Quality Release Policies

Testing was conducted to identify whether the water quality agent would be able to develop release policies for the WQSA water that produced better long-term results based on the use of non-greedy actions, or actions that did not necessarily produce the highest immediate reward. These actions would generally involve storing more WQSA water at times when it should have been released for downstream water quality improvement, so that more would be available at times of even greater need. Testing was initially conducted using the short-term agent. In addition to utilizing both the SARSA and Q-Learning methodologies and the various exploratory action selection methods described in Section 3.2.5, testing included increasing the discount rate, using eligibility traces, initializing the learning process with the basic policy and action-value functions developed in Section 5.1.1, using various configurations of state representation, dramatically increasing the earned reward, and providing for extended simulation.

Several of the tests produced policies that tended to conserve more water when the total volume of WQSA water was lower, but none of those tests produced significantly improved results from those shown by the basic release policy. In general, the tests did not result in an overall policy that was significantly different than the immediate reward policy. One of the primary reasons for this result was the extremely delayed nature of the rewards for storing additional water for use during drought periods, particularly when simulating at a daily timescale.

Additional testing was conducted using the long-term agent specification to develop improved policies over the span of many years. This testing utilized both the 30-year and synthetic hydrologic datasets, and assumed both the flexible WQSA storage exchange that allowed full storage of all WQSA water each year, and a firm storage of WQSA water that was protected from spill of 98.7 MCM (80,000 acre-feet). OFU was the primary exploration mechanism utilized. The testing showed that the agent was able to generate overall policies that were better in the long-term than the immediate reward policy at mitigating more significant temperature issues, particularly through drought periods. The performance of the agent on the 30-year historic hydrology using different agent configurations is illustrated in Figures D.1 through D.9 in Appendix D. Figure D.8 represents one of the best performing agent configurations, and is also shown as Figure 5.3 below.

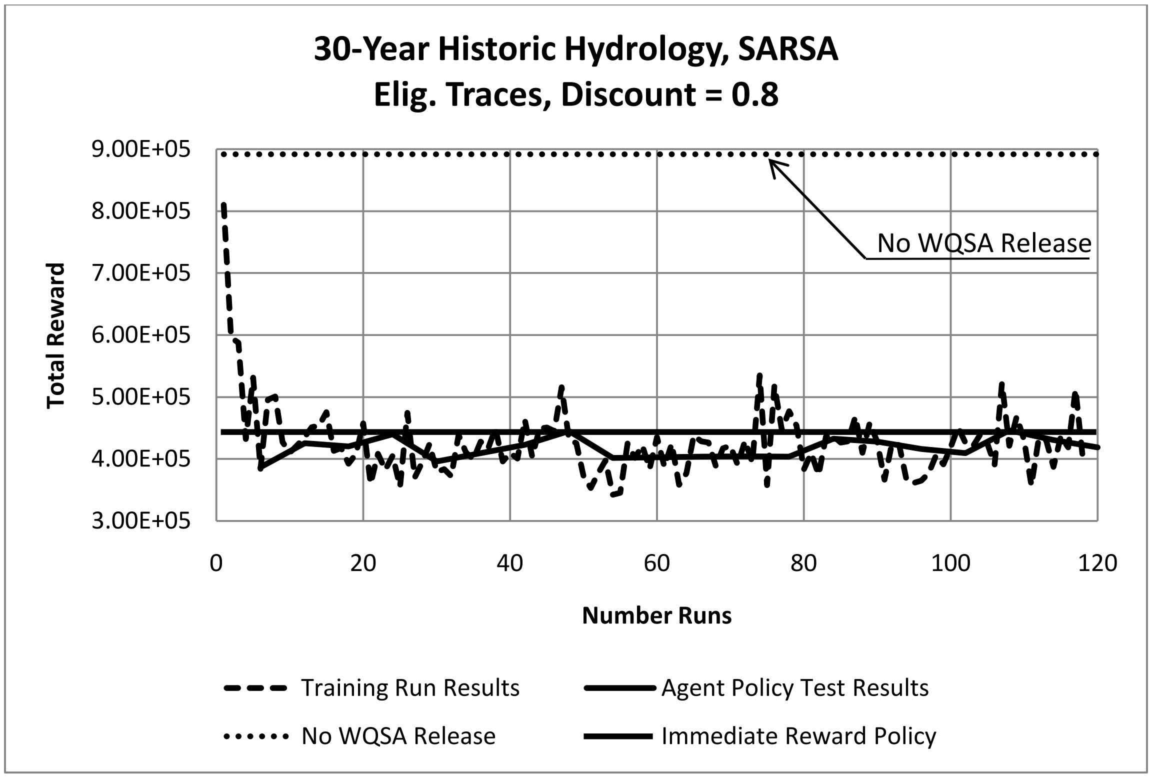


Figure 5.3 – Agent performance on 30-year historic hydrology, using SARSA with eligibility traces, λ = 0.5, γ = 0.8

Figure 5.3 above shows the sum of all rewards accrued by the agent for each 30-year historic hydrology run during agent training. The rewards are those calculated by Equation 4.5, which essentially applies a penalty for each day when water temperatures are above the preferred 22° C, using penalties that increase exponentially with increased temperature. The total reward is therefore an indicator of overall agent performance over the entire simulation, with improved performance illustrated by an agent capable of mitigating the more severe temperature violations. The reward totals for the training results shown in Figure 5.3 take into account the agent's exploratory actions as training is taking place. Also shown in Figure 5.3 is a timeseries of periodic evaluations of agent policy generated by the policy update program. This essentially shows the agent's performance if the agent were not taking exploratory actions, but rather simply following the agent's policy function at that point in the training cycle. For comparison, two horizontal lines show the values that represent the sum of all rewards for a 30-year run that would accrue in the absence of WQSA water releases, and the sum of all rewards that would be earned if the agent followed the immediate reward policy (which is the preferred policy shown in Table 4.5; essentially identical to the daily policy developed by the short-term agent in Tables C.1 and C.2).

Figure 5.3 shows that the agent improves beyond the results of the immediate reward policy at an initially high rate, and then the improvement slows, with much oscillation in results. This effect is noted in the literature when using actor-critic and GPI methods similar to the one utilized by the water quality agent (Bertsekas 1996; Szepesvári 2010). Szepesvari notes that oscillation and possible divergence can be a common occurrence with applied GPI, and the primary tool used in practice for overcoming this issue is to store the complete sequence of policies generated by the agent and then select the one illustrating the best empirical performance as measured by post-training tests (Szepesvári 2010). Based on that concept, the best policy would be that followed by the agent at training run number 54, producing the results shown in Table 5.2 below. Also shown below are the results of the best evaluation of the agent's policy function during the course of the training run. In addition, Table 5.2 shows the results for a "baseline" run without the use of WQSA water, and the results of an agent strictly following the preferred (immediate reward), chronic, or acute policy options developed in Section 4.6.1.

Table 5.2 – Evaluations of long-term agent policies over 30-year historic hydrology

		Basic Policy Options			Best Agent Policy	
Water Quality Variable	Baseline	Preferred Policy (Immediate Reward)	Chronic Policy	Acute Policy	Training	Agent Policy Evaluations
Chronic Violations	10	4	32	9	14	14
Acute Violations	39	6	44	77	47	32
Absolute Violations	416	230	97	200	116	155
22-22.5° C (# days)	120	163	133	131	131	193
22.5-23° C (# days)	92	8	339	89	63	106
23-23.5° C (# days)	67	11	167	68	87	66
23.5-24° C (# days)	85	18	45	290	155	78
24-24.5° C (# days)	74	32	16	130	69	48
24.5-25° C (# days)	53	36	10	52	33	25
25-25.5° C (# days)	73	32	16	15	8	21
25.5-26° C (# days)	82	37	18	3	6	25
26-27° C (# days)	105	53	30	0	0	29
>27° C (# days)	29	26	7	0	0	7
Total Reward	892,113	443,341	391,797	502,618	342,071	386,644

The improved agent performance shown above is evaluated through the reward calculation as computed by Equation 4.5, using a reward function multiplier of 20 and a reward function exponent of 1.5. This causes exponentially increasing rewards to be associated with the number of days with higher water temperatures, as previously discussed. For this reason, the agent is motivated to attempt to allow additional “minor violations” at lower temperatures in order to store more water so that it will be available to address more “major violations” in the long run. The agent’s general strategy is illustrated in Figure 5.4, showing the number of days in a 30-year run at water temperatures between 22° C and 24° C, and the number of days above that water temperature. The improved strategy shown in Figure 5.4 represents the best agent performance in training, as also shown in Table 5.2 above.

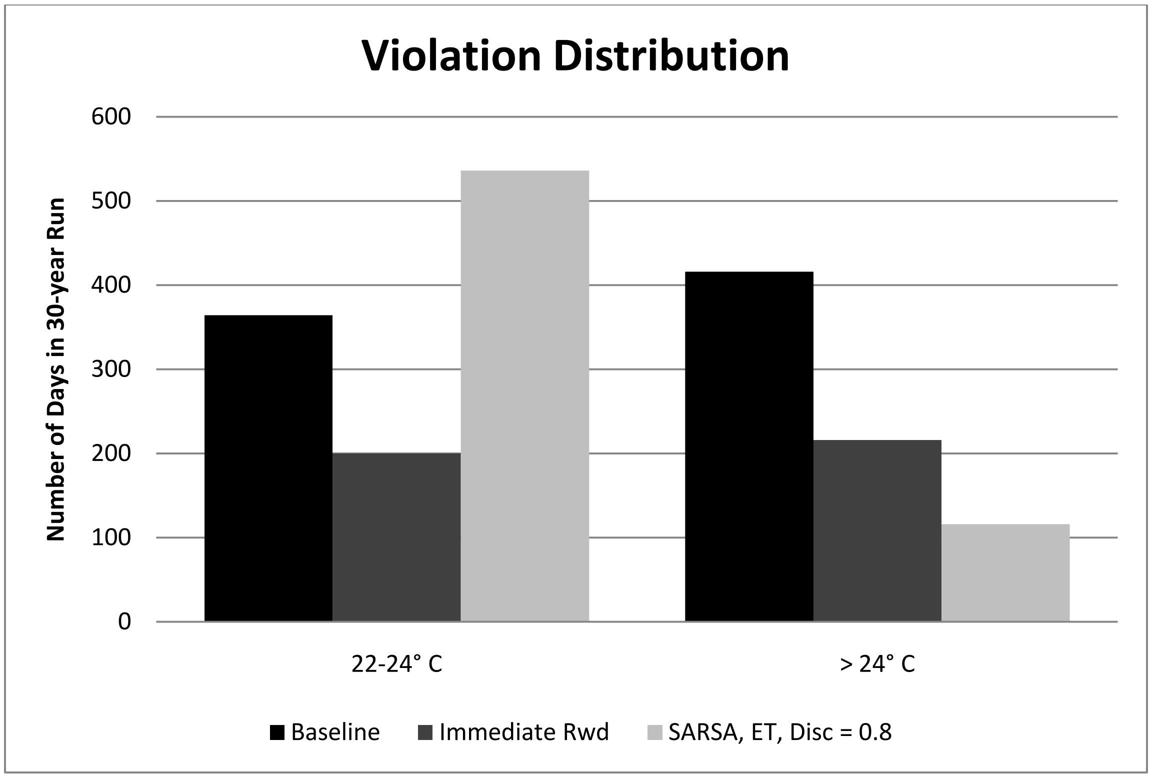


Figure 5.4 – Distribution of water temperature violations

Figure 5.4 shows that an agent following the short-term immediate reward policy does substantially better than the "baseline" case without any WQSA releases, but still tends to allow a high number of days when the water temperature would be excessively high based on the criteria discussed in Section 4.5.2.3. The long-term agent strategy allows the number of days when the temperature is in the lower end of the spectrum to actually exceed the baseline condition, but significantly reduces the number of days when higher temperatures occur.

Tests were conducted to evaluate the impacts of the level of firm storage provided to the agent. This firm storage was the maximum storage that would be protected from displacement and spill in higher water years when the reservoirs fill. The performance of the agent under different levels of firm storage is shown in Figures 5.5 and 5.6. Figure

5.5 shows performance during training, and Figure 5.6 shows evaluations of the agent policy function without exploration throughout the training cycle.

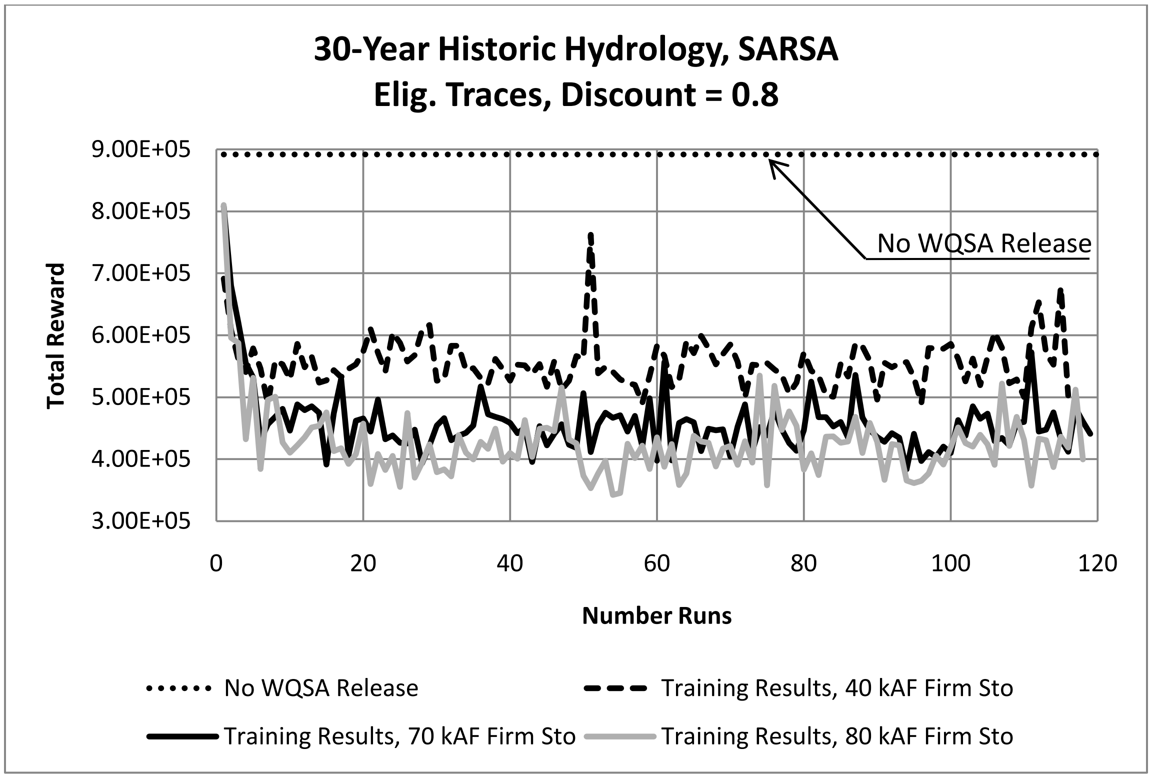


Figure 5.5 – Training results, various levels of firm storage

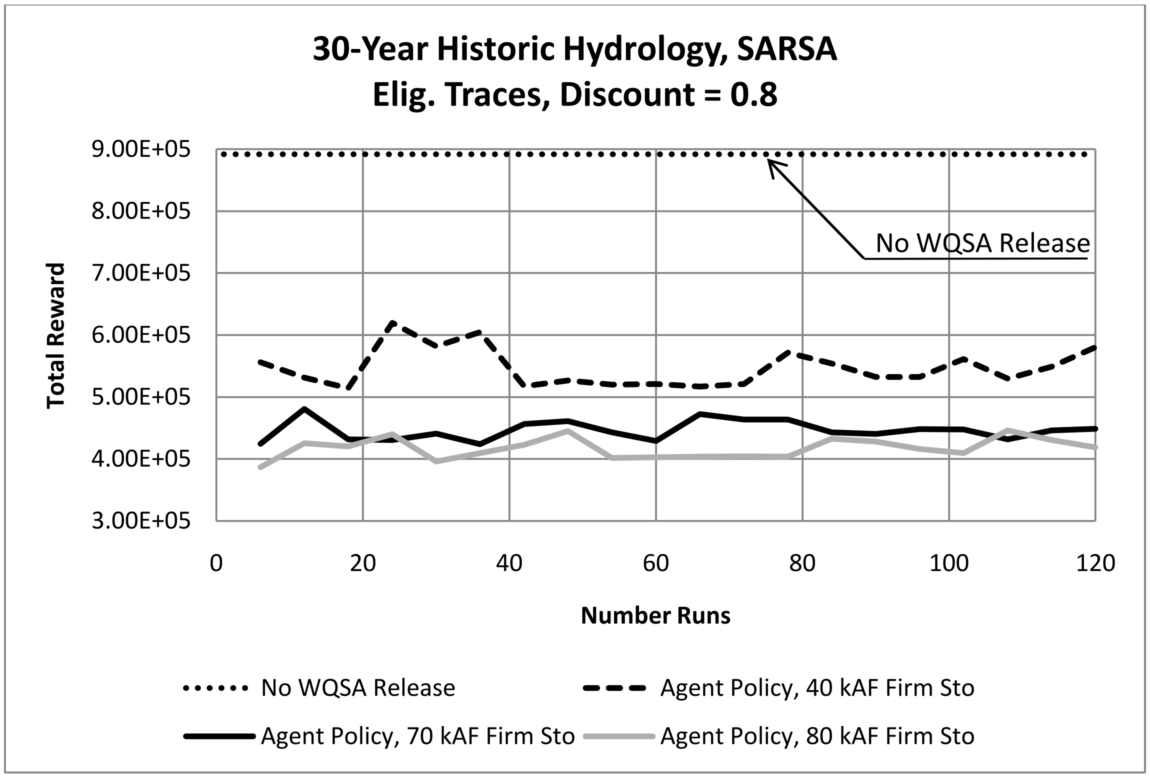


Figure 5.6 – Agent policy evaluations, various levels of firm storage

Figure 5.5 shows that agent performance during training is strongly impacted by the amount of storage that is not subject to spill. Figure 5.6 further illustrates this based on evaluations of the agent policy function during the training cycle without exploration. Test results for the best policies for each different firm storage level tested are shown in Table 5.3.

Table 5.3 – Test results for varying firm storage levels, using 30-year historic hydrology, SARSA, eligibility traces, $\lambda = 0.5$ $\gamma = 0.8$

		Firm Storage Tests		
Water Quality Variable	Baseline	49.3 MCM (40,000 acre-feet)	86.3 MCM (70,000 acre-feet)	98.7 MCM (80,000 acre-feet)
Chronic Violations	10	16	14	14
Acute Violations	39	42	61	47
Absolute Violations	416	189	138	116
22-22.5° C (# days)	120	131	148	131
22.5-23° C (# days)	92	157	130	63
23-23.5° C (# days)	67	101	80	87
23.5-24° C (# days)	85	87	196	155
24-24.5° C (# days)	74	50	85	69
24.5-25° C (# days)	53	28	39	33
25-25.5° C (# days)	73	27	12	8
25.5-26° C (# days)	82	32	2	6
26-27° C (# days)	105	37	0	0
>27° C (# days)	29	15	0	0
Total Reward	892,113	488,704	383,745	342,071

Overall, it was found during testing that the agent's performance was maximized when it learned policies that utilized the WQSA water when necessary, but also stored just enough WQSA water to address the most significant issues during periods of prolonged drought. This is intuitively reasonable, since the goal of the agent would be to store only exactly that which is necessary to make it through drought events, without any carryover storage at the end of each event. This concept is illustrated by one of the agent's best performances on the 30-year hydrologic dataset, as shown in Figure 5.7. Figure 5.7 shows the total amount of storage of WQSA water during the 30-year model run.

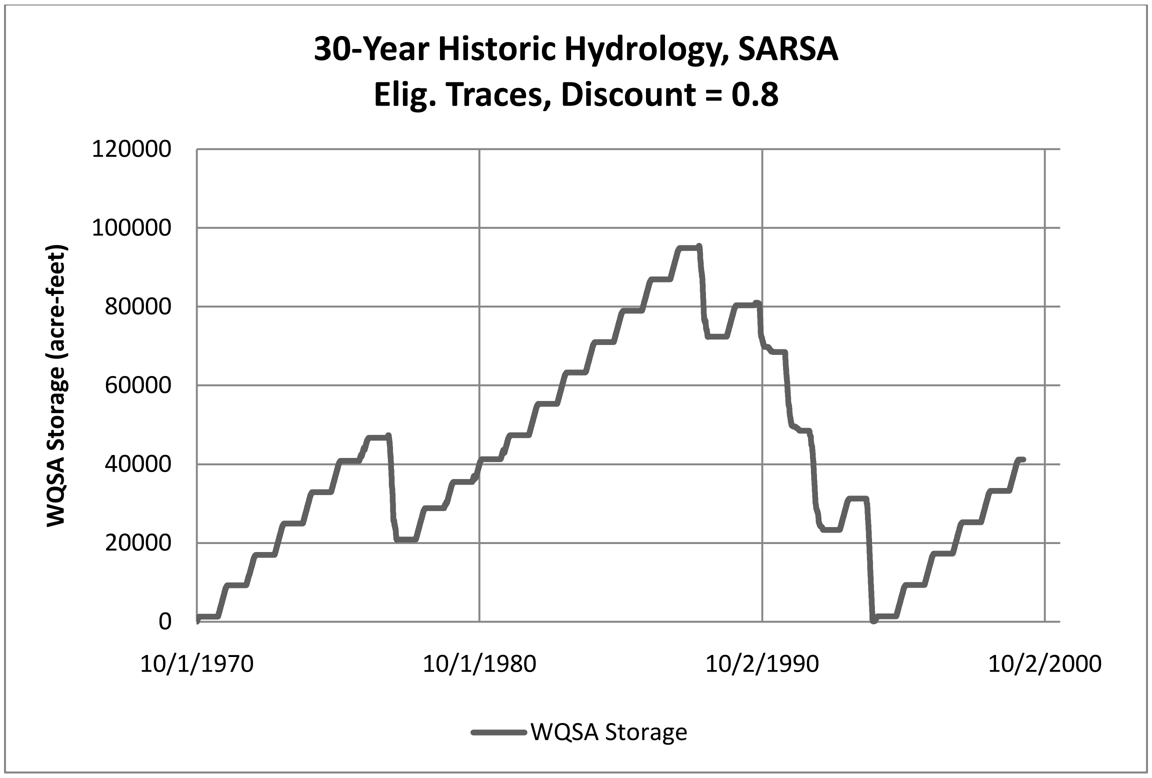


Figure 5.7 – Total WQSA storage, 30-year historic hydrology, SARSA, eligibility traces, $\lambda = 0.5$ $\gamma = 0.8$

Figure 5.7 shows that the agent was able to store enough WQSA water to just barely provide drought protection during the early 1990's, without carrying over any additional storage. The water temperatures predicted during the later part of the drought in this run are shown in Figure 5.8.

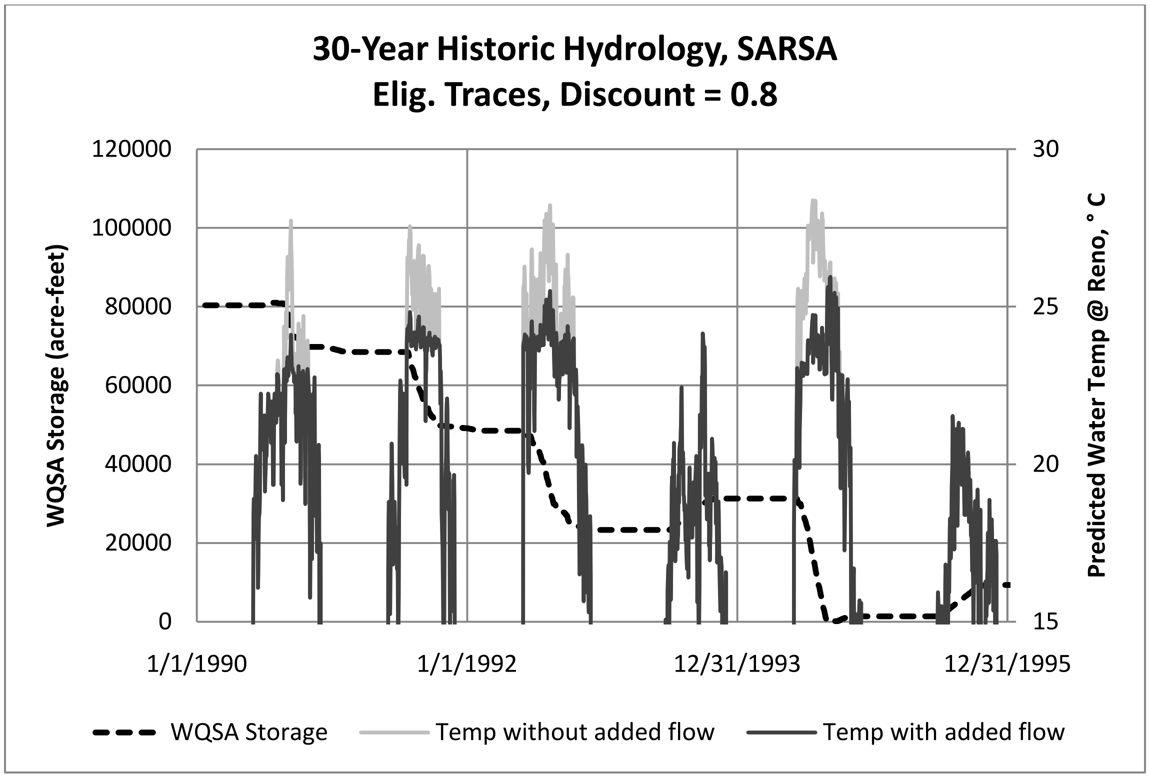


Figure 5.8 – Water temperature predictions, 30-year historic hydrology, SARSA, eligibility traces, λ = 0.5 γ = 0.8

Figure 5.8 shows the predicted water temperatures if no agent action were taken, and the predicted temperatures based on the agent's action selection. It illustrates that the agent was successfully able to mitigate most of the highest potential temperature issues in the river during the drought of the early 1990's, with essentially no water carried over at the end of the drought event.

Tests were conducted using the synthetic hydrology described in Section 4.5.1.2. These tests showed that the agent was able to adapt its strategies to different environmental conditions. The agent's performance using the synthetic hydrology is shown in Figure 5.9.

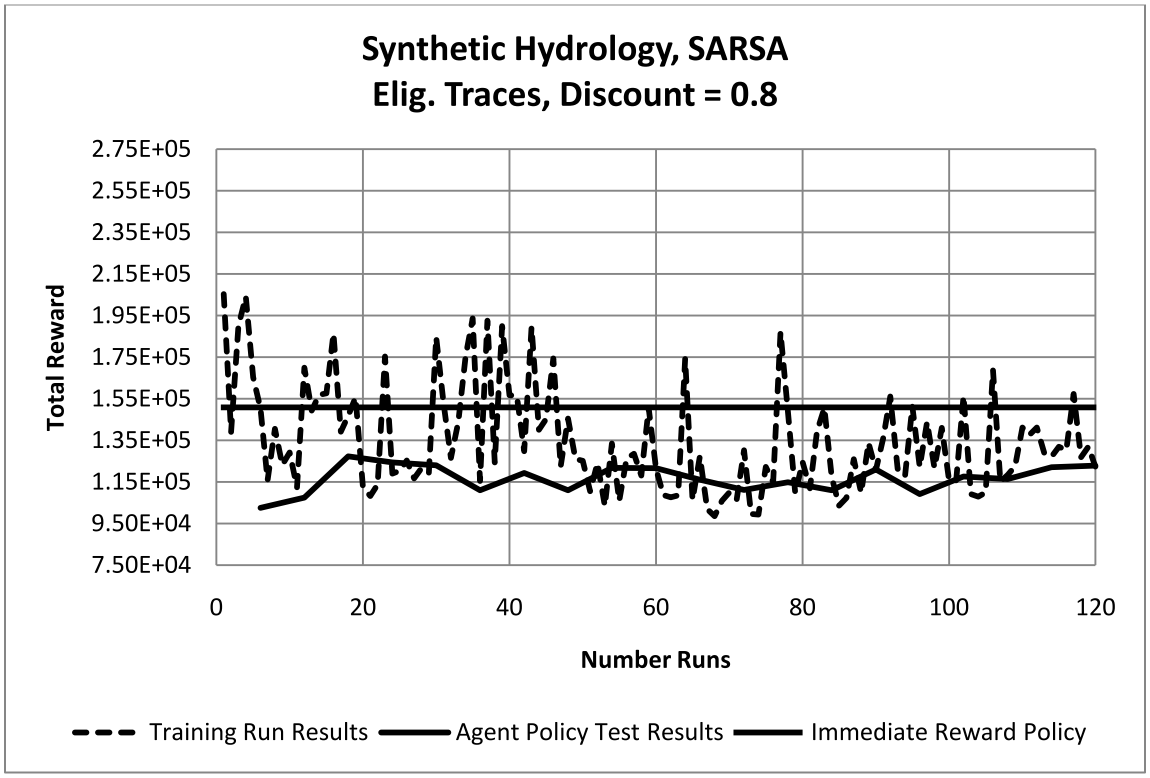


Figure 5.9 - Agent performance on synthetic hydrology, using SARSA with eligibility traces, λ = 0.5, γ = 0.8. Note that baseline reward result for this case is approx. 572,000

Figure 5.9 illustrates that the agent is able to produce improved policies when faced with different hydrologic scenarios. Similar to the case of the 30-year historic hydrology shown in Figure 5.7, the agent also seeks to maximize its performance by storing just enough water to get through drought periods in the synthetic dataset. An example of this is shown in Figure 5.10, which represents the agent's storage strategy in one of the better performing runs using the synthetic hydrology.

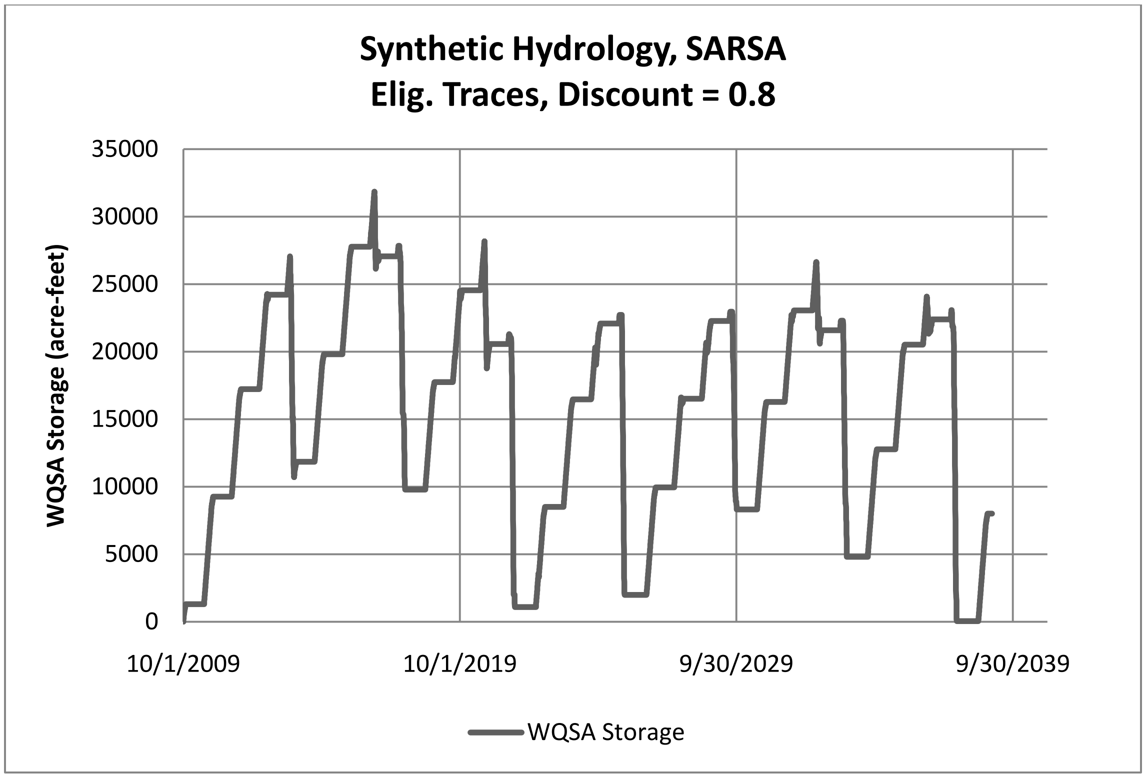


Figure 5.10 - Total WQSA storage, synthetic hydrology, SARSA, eligibility traces, $\lambda = 0.5$ $\gamma = 0.8$

Figure 5.10 illustrates the agent's ability to maximize the use of the WQSA water, without storing more than is necessary to address periodic drought conditions. The figure shows that the agent seeks and discovers improved longer-term strategies, even when fairly significant drought periods come at more frequent intervals than with the 30-year historic hydrology. This serves to show that the agent has the ability to adapt to changing environments, and can evaluate potential future conditions using long-term projected hydrology.

5.2 – Evaluation of Function Approximation

The tests and results shown thus far in this chapter represent the agent's abilities when using the tabular implementation described in Section 3.2.5.1. The use of neural networks as a function approximation tool for the action-value and agent policy functions was also evaluated and tested. Tests and evaluations were conducted for the short-term immediate reward case, as well as the long-term case, both using the 30-year historic hydrology.

5.2.1 – Function Approximation for Short-Term Immediate Reward Agent

An evaluation was conducted on the neural network structure and training parameters to determine the best configuration for use in function approximation for the short-term immediate reward agent. As a part of this evaluation, the neural network function approximation was tested against its ability to replicate the analytically derived action-value and policy functions as discussed in Section 5.1. The sum of the squared error in the overall function approximation for the complete state and state-action spaces was used as the performance measure for the approximation. The results of neural network training through the first 5,000 iterations of the genetic algorithm are shown in Figures 5.11 through 5.22. The various tests conducted, and their associated graphical results, are summarized as follows:

- Neural network structure: number of hidden units in one hidden layer
    - Figure 5.11 – Action-value neural network
    - Figure 5.12 – Policy neural network
- Neural network structure: number of hidden layers
    - Figure 5.13 – Action-value neural network
    - Figure 5.14 – Policy neural network
- Genetic algorithm training: number of solutions in the genetic population
    - Figure 5.15 – Action-value neural network
    - Figure 5.16 – Policy neural network
- Neural network configuration: minimum and maximum bounds on connection weights
    - Figure 5.17 – Action-value neural network
    - Figure 5.18 – Policy neural network
- Neural network configuration: sigmoid scale factors
    - Figure 5.19 – Action-value neural network
    - Figure 5.20 – Policy neural network
- Genetic algorithm training: mutation rate
    - Figure 5.21 – Action-value neural network
    - Figure 5.22 – Policy neural network

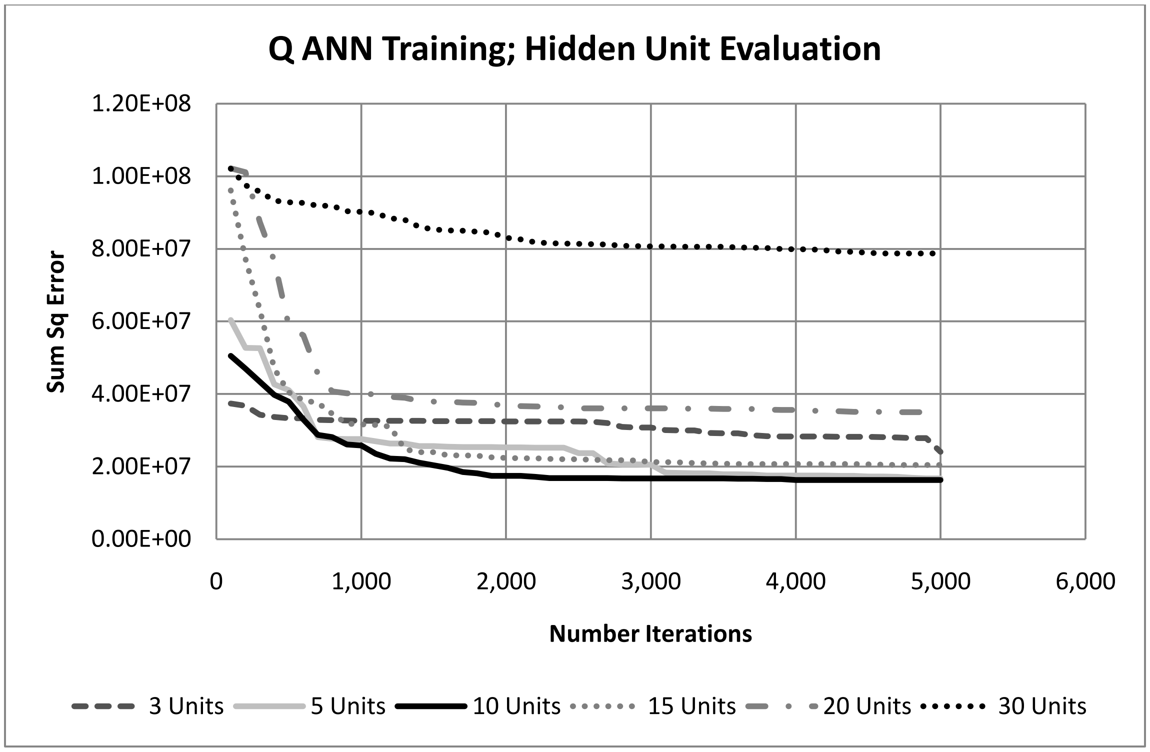


Figure 5.11 – Action-value neural network training; 1 hidden layer, min/max weights = +-30, genetic population = 20, mutation rate = 10%

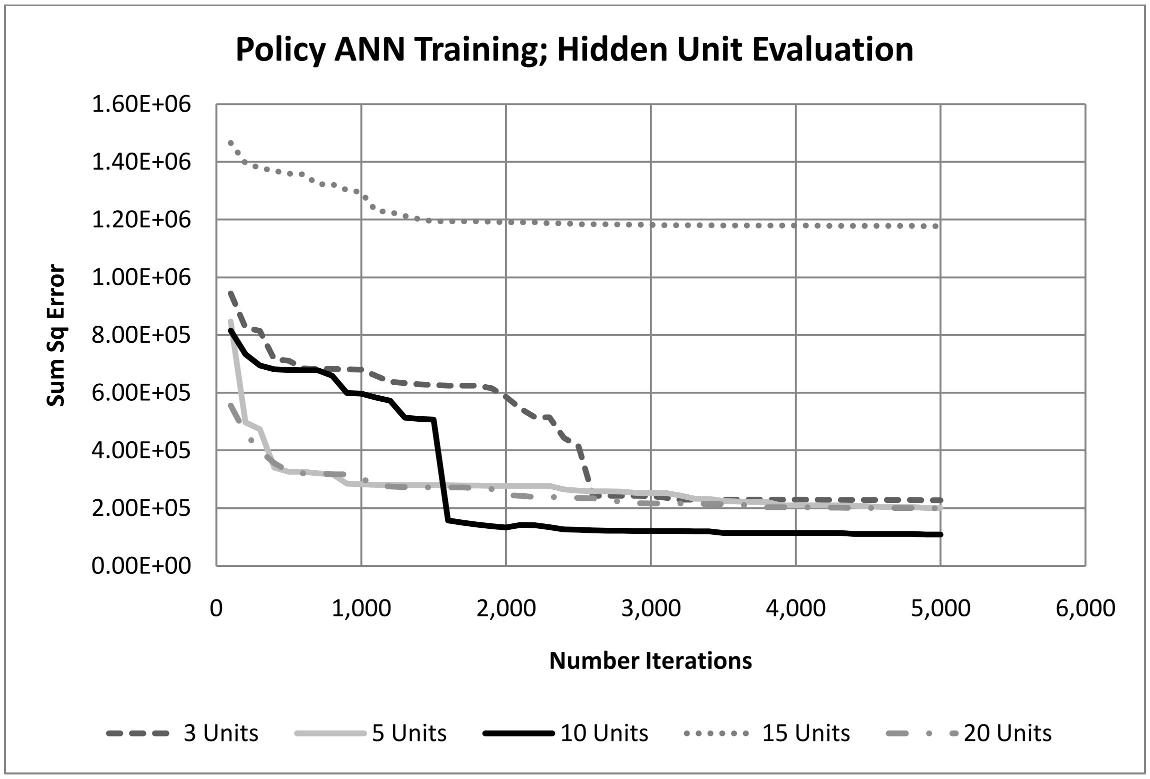


Figure 5.12 – Policy neural network training using different numbers of hidden units; 1 hidden layer, min/max weights = +-30, genetic population = 20, mutation rate = 10%

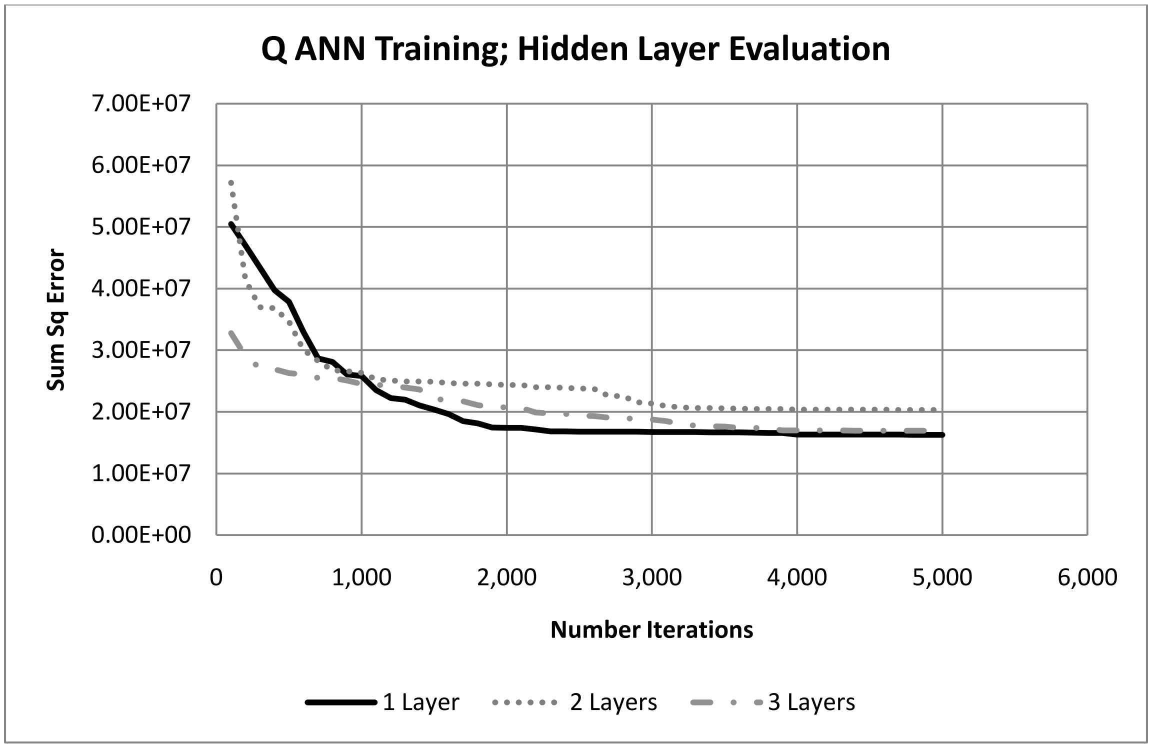


Figure 5.13 – Action-value neural network training using different numbers of hidden layers; 10 hidden units, min/max weights = +-30, genetic population = 20, mutation rate = 10%

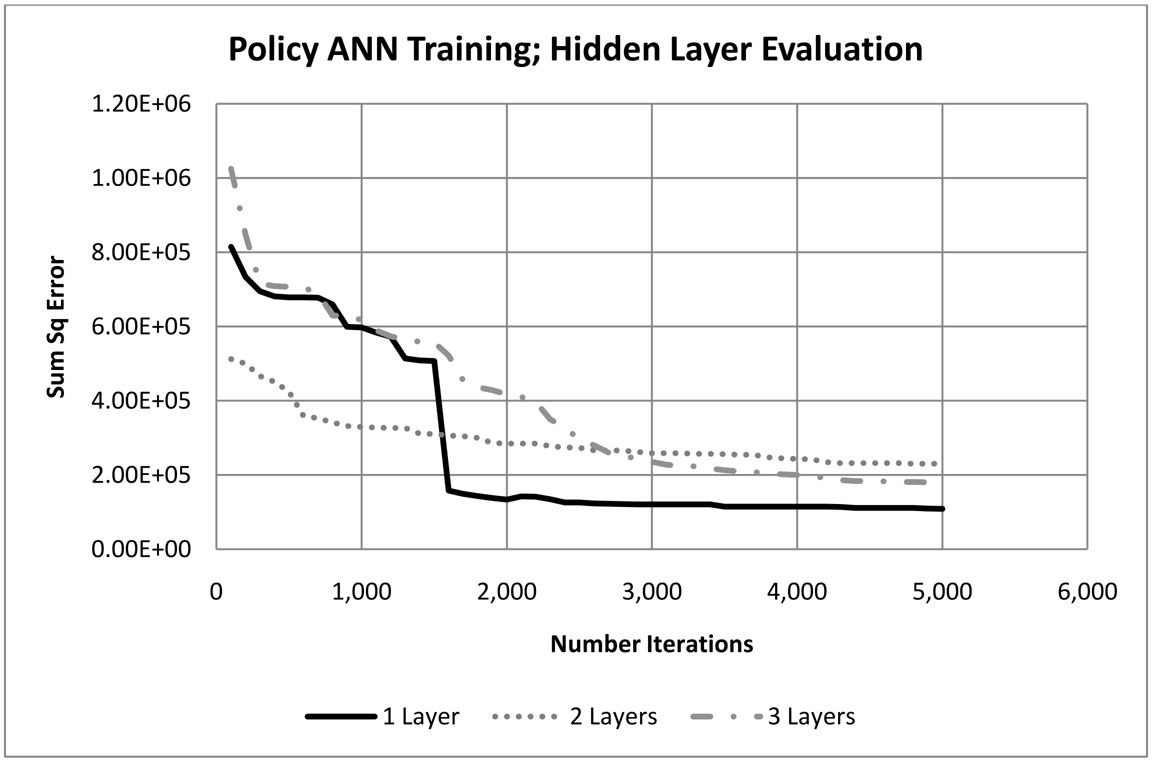


Figure 5.14 – Policy neural network training using different numbers of hidden layers; 10 hidden units, min/max weights = +-30, genetic population = 20, mutation rate = 10%

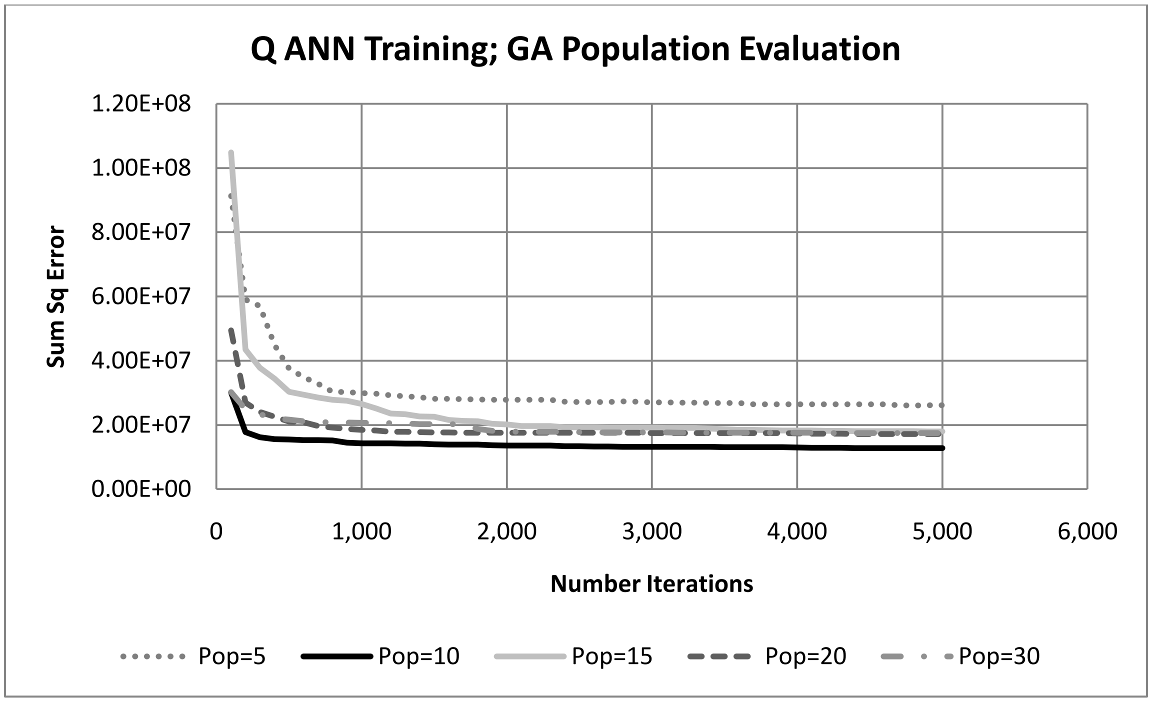


Figure 5.15 – Action-value neural network training using different numbers of genetic populations; 1 hidden layer, 10 hidden units, min/max weights = +-30, mutation rate = 30%

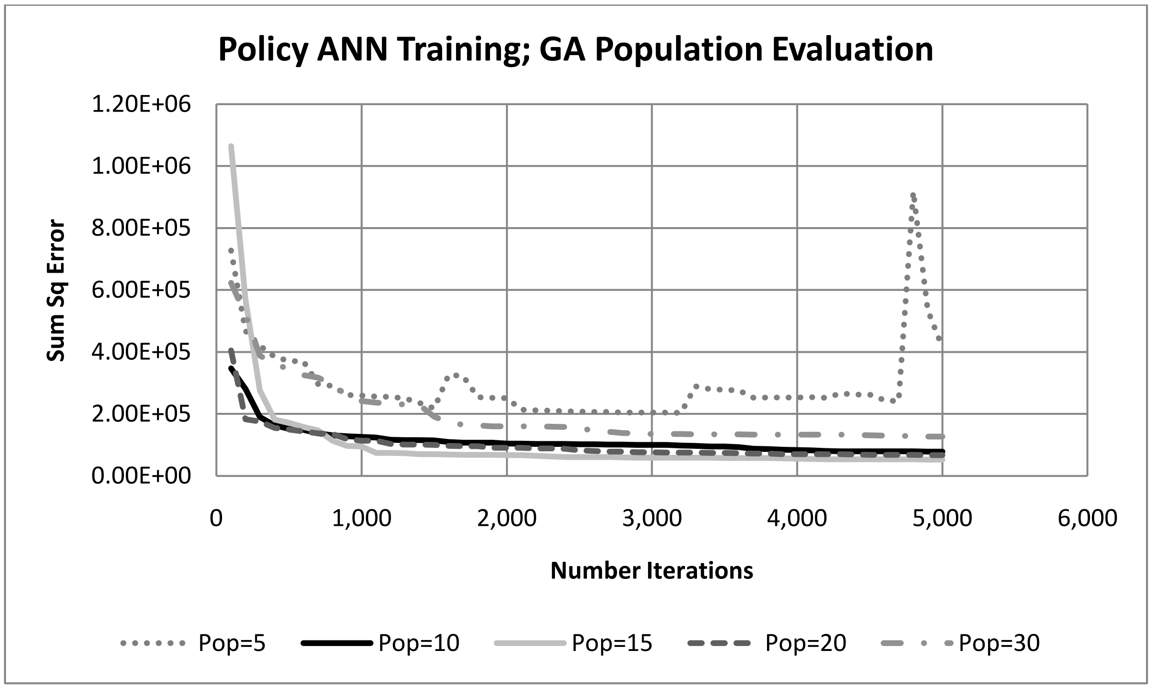


Figure 5.16 – Policy neural network training using different numbers of genetic populations; 1 hidden layer, 10 hidden units, min/max weights = +-30, mutation rate = 40%

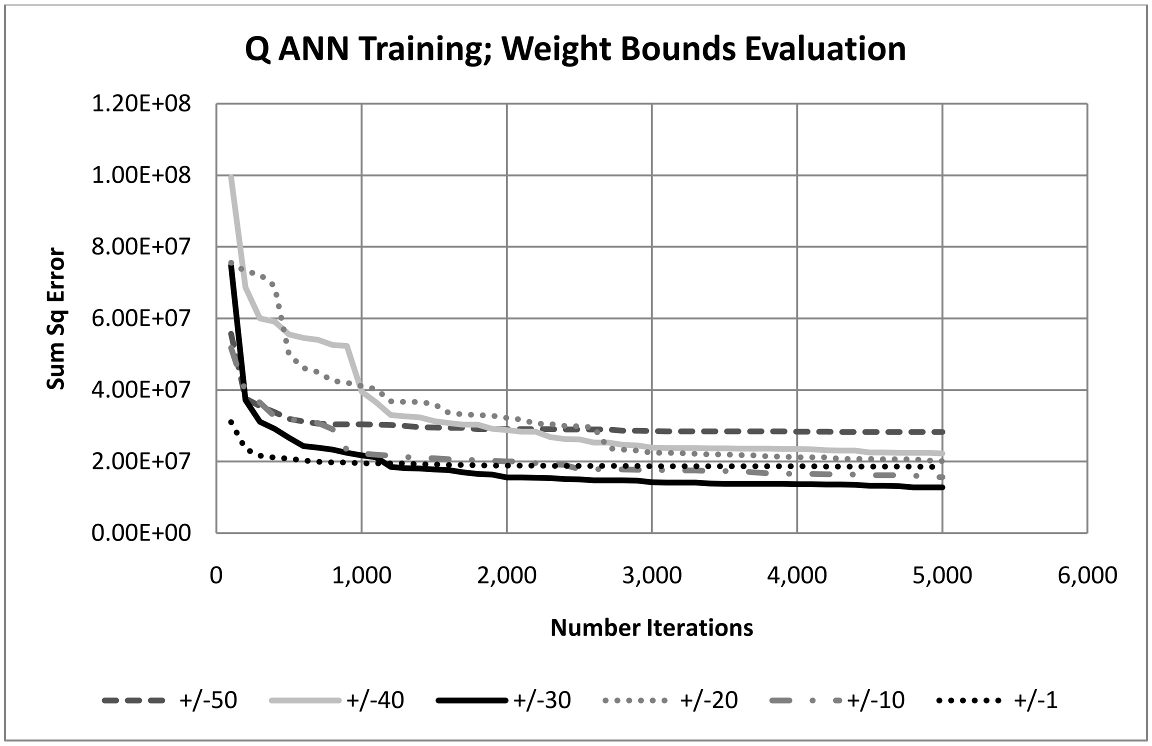


Figure 5.17 – Action-value neural network training using different bounds on the connection weights; 1 hidden layer, 10 hidden units, genetic population = 10, mutation rate = 30%

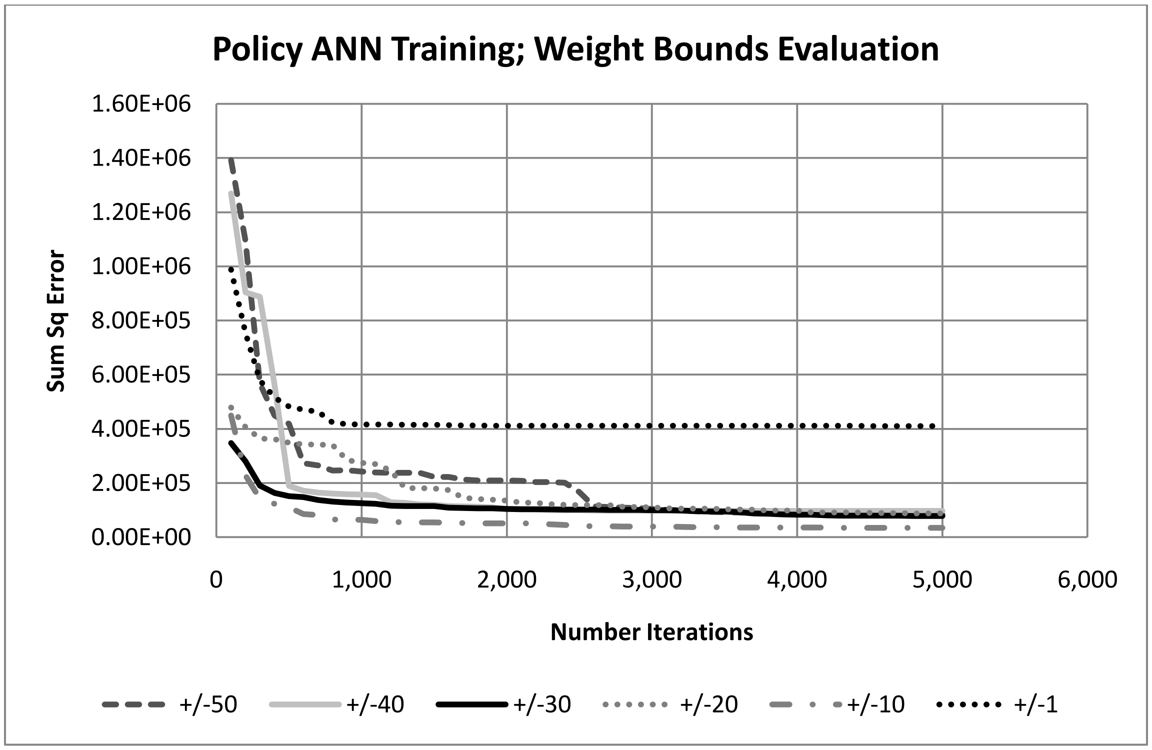


Figure 5.18 – Policy neural network training using different bounds on the connection weights; 1 hidden layer, 10 hidden units, genetic population = 10, mutation rate = 40%

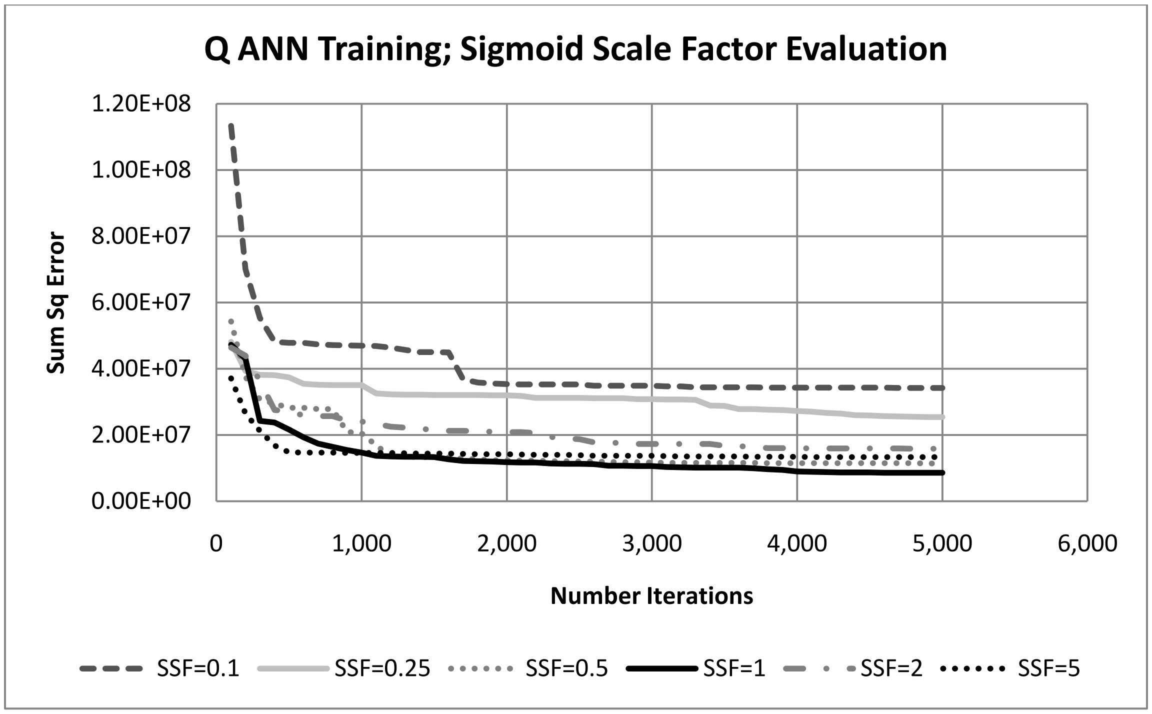


Figure 5.19 – Action-value neural network training using different bounds on the connection weights; 1 hidden layer, 10 hidden units, min/max weights = +/-30, genetic population = 10, mutation rate = 30%

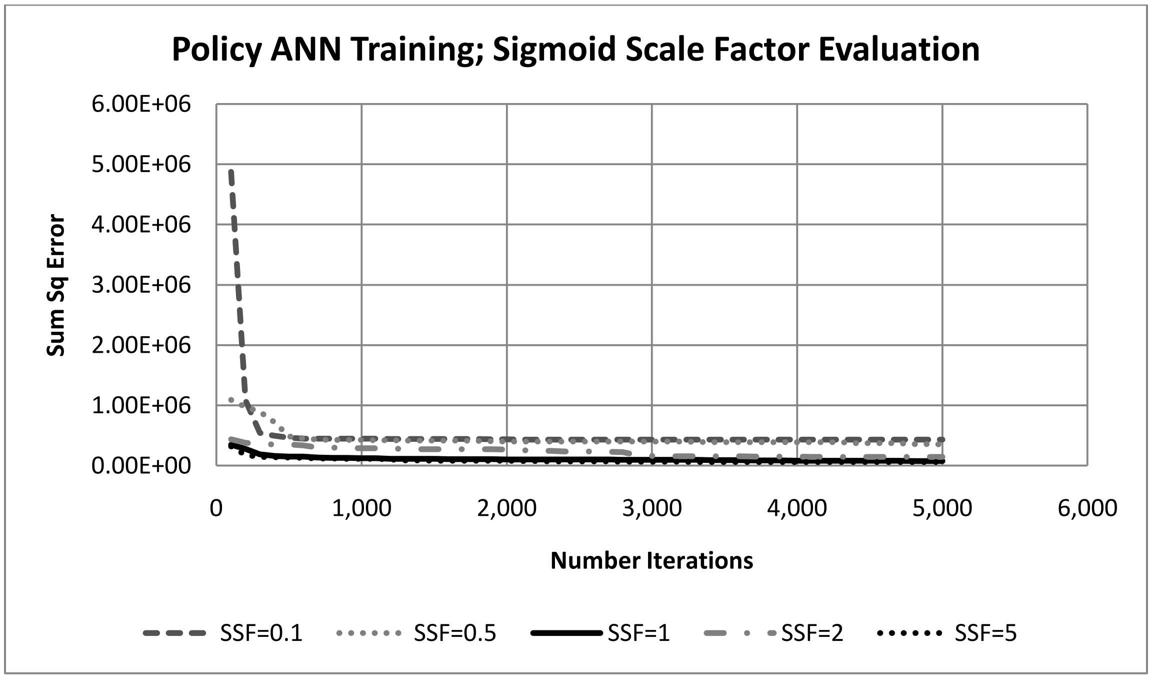


Figure 5.20 – Policy neural network training using different bounds on the connection weights; 1 hidden layer, 10 hidden units, min/max weights = +/-30, genetic population = 10, mutation rate = 40%

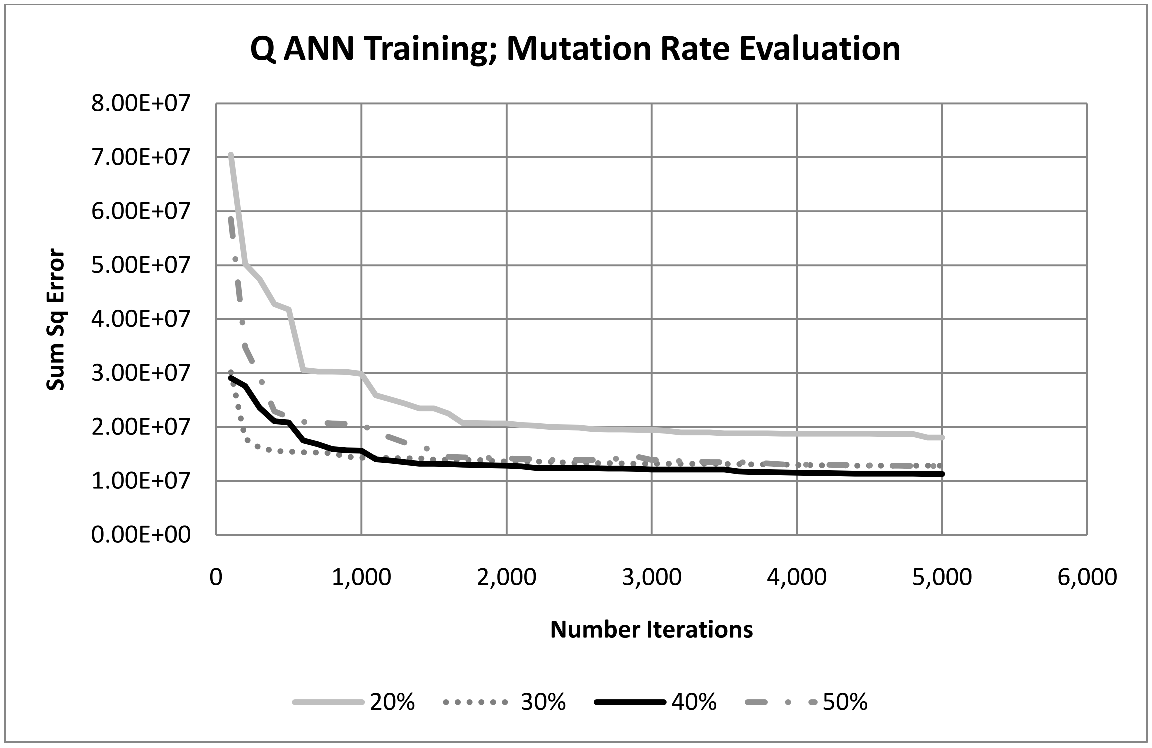


Figure 5.21 – Action-value neural network training using different genetic algorithm mutation rates; 1 hidden layer, 10 hidden units, min/max weights = +/-30, genetic population = 10

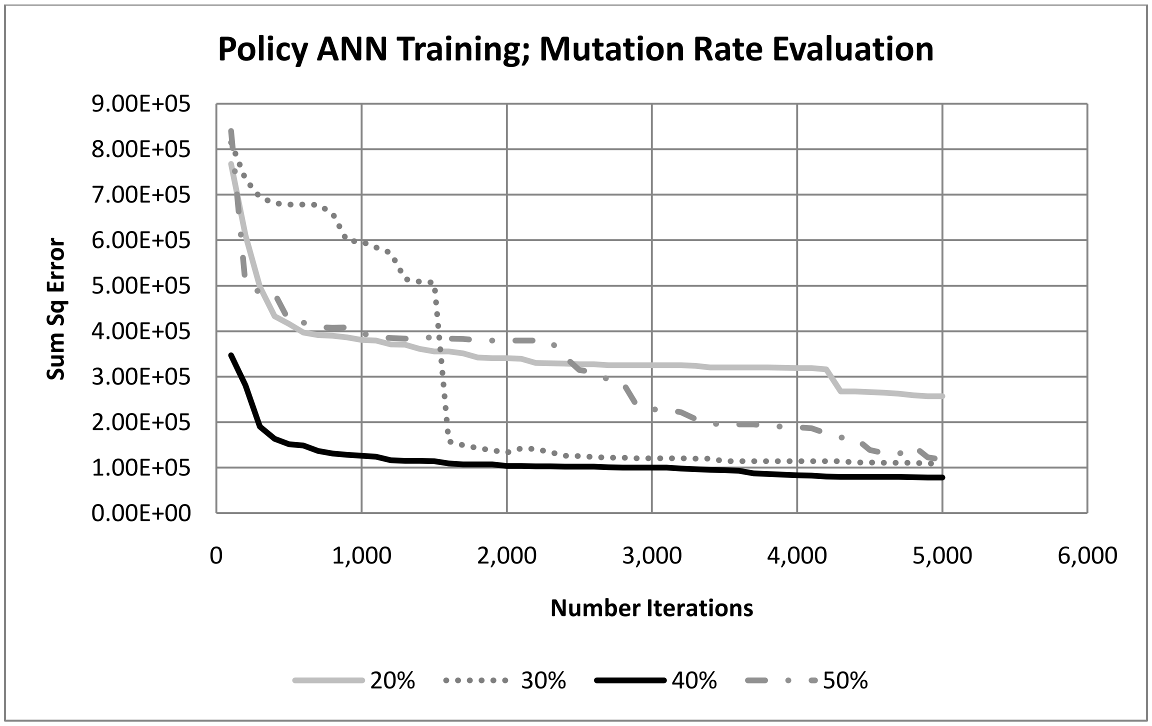


Figure 5.22 – Policy neural network training using different genetic algorithm mutation rates; 1 hidden layer, 10 hidden units, min/max weights = +/-30, genetic population = 10

Based on the results shown in the figures above, it was generally found that the best structure and configuration for the neural networks is summarized as:

- Number hidden layers: 1
- Number hidden units: 10
- Minimum and maximum connection weights: +/- 30
- Sigmoid scale factor: 1
- Genetic solution population: 10-15
- Genetic mutation rate: 30-40%

Testing was conducted on the short-term agent using those basic structure and configuration settings for the action-value and policy neural networks. It was generally found that in actual training, the agent encountered difficulties replicating the action-value function, and thus the overall policy function produced by the tabular implementation. One of the primary issues causing the difficulty involved the less precise nature of the neural network approximation as compared to the tabular representation of the action-value function. An evaluation was conducted on the ability of the policy update program to generate an agent policy similar to that of the tabular implementation in an idealized setting, where the analytically derived tabular action-value function was provided to the action-value neural network as training data. Based on the neural network approximation of the action-value function, the associated policy was significantly different than that created by the tabular implementation, with the agent's release actions generally only occurring when water temperatures were predicted to be above 24.5° C. Based on the results of the testing on the short-term agent using neural network approximation, additional research into the nature of the issues encountered and potential solutions to those issues is recommended as future work to this study, as discussed in Section 6.4 in the proceeding chapter.

5.2.2 – Function Approximation for Long-Term Agent

Testing and evaluation was conducted using neural network function approximation with the long-term agent using the 30-year historic hydrologic dataset. The same neural network structure and configuration was used for the long-term agent as was used with the short-term agent. This testing provided improved results over those produced by the short-term agent using function approximation, attributable to a simpler action-value function. The long-term agent was able to produce strategies that outperformed the immediate reward case, similar to the long-term agent using tabular implementation. The agent's performance using neural network function approximation is shown in Figure 5.13 below.

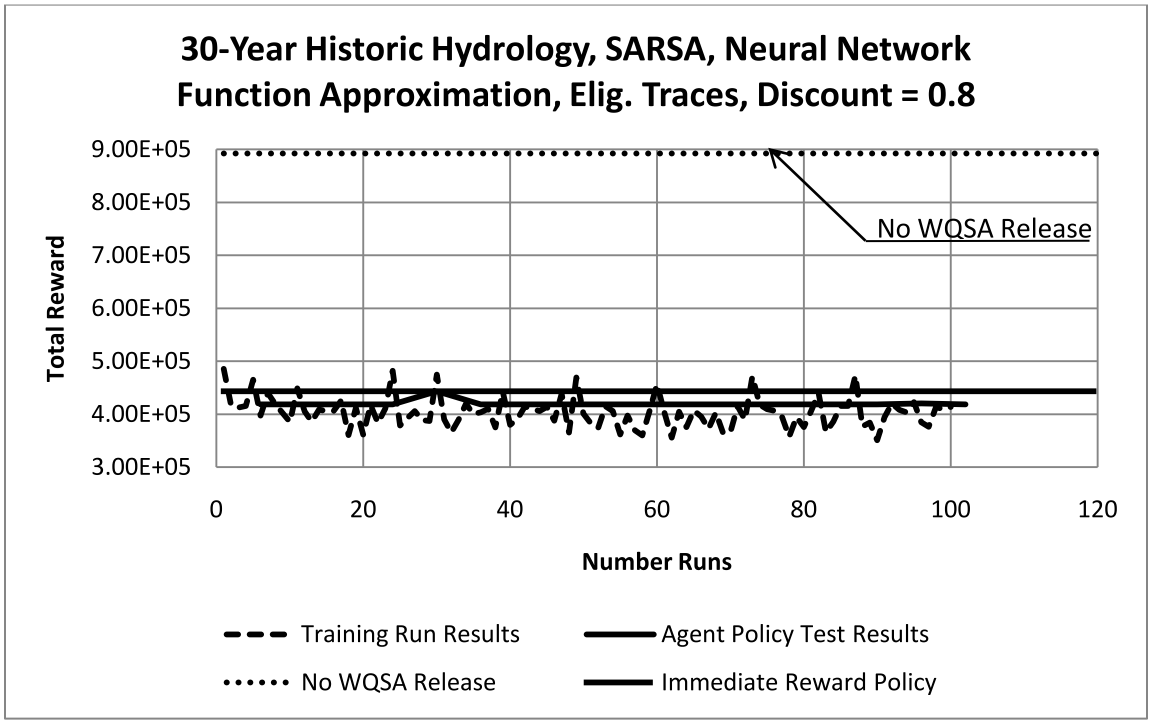


Figure 5.23 – Agent performance on 30-year historic hydrology, using SARSA with eligibility traces and neural network function approximation, λ = 0.5, γ = 0.8

A comparison of the performance of the agent using the neural network implementation against the tabular implementation is shown in Table 5.4. Table 5.4 shows agent performance during actual testing, with the agent using exploratory actions. The results of the evaluation of the agent policy throughout the training cycle based on the agent policy function without exploratory actions are not shown, but exhibited similar behavior. Table 5.4 shows that the agent performance is generally similar using either the tabular or neural network implementation, with the exception of the overall average of the training results using the neural network implementation, which was better than the average training result for the tabular approach. This is largely due to the fact that the agent using neural network implementation does not perform as poorly during the initial learning phase.

Table 5.4 – Comparison of the training results of agents using tabular and neural network implementations, SARSA, no eligibility traces, $\gamma = 0.8$

	Avg. Training Run		Best Training Run	
Water Quality Variable	Tabular Agent	Neural Network Agent	Tabular Agent	Neural Network Agent
Chronic Violations	15	14	17	15
Acute Violations	29	29	55	39
Absolute Violations	169	152	100	100
22-22.5° C (# days)	188	165	154	144
22.5-23° C (# days)	116	136	189	202
23-23.5° C (# days)	66	78	115	143
23.5-24° C (# days)	68	61	133	43
24-24.5° C (# days)	40	34	48	15
24.5-25° C (# days)	25	21	22	14
25-25.5° C (# days)	25	23	13	15
25.5-26° C (# days)	28	26	8	17
26-27° C (# days)	38	35	9	32
>27° C (# days)	14	13	0	7
Total Reward	426,352	403,199	346,582	350,945

5.3 – Evaluations of SARSA, Q-Learning, Eligibility Traces, and Discount Rates

The use of both SARSA and Q-Learning were evaluated in tests of the long-term agent. The evaluation involved the comparison of the two technologies using the 30-year historic hydrology. In addition, the use of eligibility traces was evaluated for each type of reinforcement learning technology on the 30-year historic hydrology. The performance of the agent using Q-Learning and SARSA without eligibility traces is shown in Table 5.7. The performance of the agent using Q-Learning and SARSA with eligibility traces is shown in Table 5.8. The results are reconfigured in Tables 5.9 and 5.10 to provide side-by-side comparison of the use of eligibility traces for Q-Learning and SARSA respectively. All results shown are for agent performance during training. Results for the evaluation of agent policy functions throughout the training cycle are not shown, but illustrated similar behavior.

Table 5.7 – Comparison of Q-Learning and SARSA, no eligibility traces, $\gamma = 0.8$

	Avg. Training Run		Best Training Run	
Water Quality Variable	Q-Learning, No E.T.	SARSA, No E.T.	Q-Learning, No E.T.	SARSA, No E.T.
Chronic Violations	13	15	17	17
Acute Violations	30	29	45	55
Absolute Violations	170	169	97	100
22-22.5° C (# days)	177	188	143	154
22.5-23° C (# days)	105	116	187	189
23-23.5° C (# days)	65	66	136	115
23.5-24° C (# days)	73	68	66	133
24-24.5° C (# days)	41	40	18	48
24.5-25° C (# days)	26	25	12	22
25-25.5° C (# days)	24	25	14	13
25.5-26° C (# days)	27	28	20	8
26-27° C (# days)	38	38	26	9
>27° C (# days)	13	14	7	0
Total Reward	425,923	426,352	350,561	346,582

Table 5.8 – Comparison of Q-Learning and SARSA, with eligibility traces, $\gamma = 0.8$ $\lambda = 0.5$

	Avg. Training Run		Best Training Run	
Water Quality Variable	Q-Learning, E.T.	SARSA, E.T.	Q-Learning, E.T.	SARSA, E.T.
Chronic Violations	14	14	14	14
Acute Violations	30	32	52	47
Absolute Violations	170	169	105	116
22-22.5° C (# days)	179	186	137	131
22.5-23° C (# days)	110	113	176	163
23-23.5° C (# days)	63	66	113	87
23.5-24° C (# days)	74	76	124	155
24-24.5° C (# days)	42	42	47	69
24.5-25° C (# days)	26	27	21	33
25-25.5° C (# days)	24	24	13	8
25.5-26° C (# days)	27	26	11	6
26-27° C (# days)	38	37	10	0
>27° C (# days)	13	12	3	0
Total Reward	425,614	425,375	349,060	342,071

Table 5.9 – Comparison of the use of eligibility traces with Q-Learning, $\gamma = 0.8$ $\lambda = 0.5$

	Avg. Training Run		Best Training Run	
Water Quality Variable	Q-Learning, No E.T.	Q-Learning, E.T.	Q-Learning, No E.T.	Q-Learning, E.T.
Chronic Violations	13	14	17	14
Acute Violations	30	30	45	52
Absolute Violations	170	170	97	105
22-22.5° C (# days)	177	179	143	137
22.5-23° C (# days)	105	110	187	176
23-23.5° C (# days)	65	63	136	113
23.5-24° C (# days)	73	74	66	124
24-24.5° C (# days)	41	42	18	47
24.5-25° C (# days)	26	26	12	21
25-25.5° C (# days)	24	24	14	13
25.5-26° C (# days)	27	27	20	11
26-27° C (# days)	38	38	26	10
>27° C (# days)	13	13	7	3
Total Reward	425,923	425,614	350,561	349,060

Table 5.10 – Comparison of the use of eligibility traces with SARSA, $\gamma = 0.8$ $\lambda = 0.5$

	Avg. Training Run		Best Training Run	
Water Quality Variable	SARSA, No E.T.	SARSA, E.T.	SARSA, No E.T.	SARSA, E.T.
Chronic Violations	15	14	17	14
Acute Violations	29	32	55	47
Absolute Violations	169	169	100	116
22-22.5° C (# days)	188	186	154	131
22.5-23° C (# days)	116	113	189	163
23-23.5° C (# days)	66	66	115	87
23.5-24° C (# days)	68	76	133	155
24-24.5° C (# days)	40	42	48	69
24.5-25° C (# days)	25	27	22	33
25-25.5° C (# days)	25	24	13	8
25.5-26° C (# days)	28	26	8	6
26-27° C (# days)	38	37	9	0
>27° C (# days)	14	12	0	0
Total Reward	426,352	425,375	346,582	342,071

Tables 5.7 and 5.8 show that average agent performance during training was highly similar under both SARSA and Q-Learning for the tests conducted. The best policy discovered by the agent, both with and without eligibility traces, was through the use of SARSA. The best overall policy was found using SARSA with eligibility traces. Tables 5.9 and 5.10 show that the use of eligibility traces had little effect on agent performance under Q-Learning for the tests conducted. The use of eligibility traces improved the performance of the agent when utilizing SARSA.

An evaluation was also conducted of the reinforcement learning discount rate, as used with Q-Learning on the long-term agent using the 30-year hydrology. The results of the analysis are shown in Tables 5.11 and 5.12. Table 5.11 represents the average results across all training model simulations, and Table 5.12 shows the results of the best single model simulation. All results shown are for agent performance during training. Results for the evaluation of agent policy functions throughout the training cycle are not shown, but generally followed similar behavior.

Table 5.11 – Comparison of average training results with different discount rates, Q-Learning, no eligibility traces

	Average Training Run					
Water Quality Variable	$\gamma = 0.0$	$\gamma = 0.1$	$\gamma = 0.2$	$\gamma = 0.5$	$\gamma = 0.8$	$\gamma = 0.95$
Chronic Violations	8	8	8	10	17	14
Acute Violations	17	18	19	24	45	31
Absolute Violations	196	192	189	180	97	174
22-22.5° C (# days)	153	154	158	169	143	183
22.5-23° C (# days)	51	58	61	81	187	113
23-23.5° C (# days)	39	42	43	53	136	65
23.5-24° C (# days)	41	46	47	58	66	76
24-24.5° C (# days)	33	34	34	38	18	42
24.5-25° C (# days)	27	27	26	26	12	27
25-25.5° C (# days)	30	29	28	27	14	25
25.5-26° C (# days)	36	36	34	30	20	28
26-27° C (# days)	49	47	47	42	26	38
>27° C (# days)	21	20	20	16	7	14
Total Reward	442,035	440,031	435,854	428,147	425,614	436,371

Table 5.12 – Comparison of best single run training results with different discount rates, Q-Learning, no eligibility traces

	Best Training Run					
Water Quality Variable	γ = 0.0	γ = 0.1	γ = 0.2	γ = 0.5	γ = 0.8	γ = 0.95
Chronic Violations	14	13	11	14	17	11
Acute Violations	39	41	54	52	45	20
Absolute Violations	137	130	133	114	97	174
22-22.5° C (# days)	118	129	111	116	143	209
22.5-23° C (# days)	133	115	128	162	187	55
23-23.5° C (# days)	88	84	102	118	136	37
23.5-24° C (# days)	91	134	135	123	66	47
24-24.5° C (# days)	44	54	63	50	18	26
24.5-25° C (# days)	29	33	44	29	12	19
25-25.5° C (# days)	20	15	18	17	14	27
25.5-26° C (# days)	20	14	8	10	20	36
26-27° C (# days)	19	13	0	8	26	45
>27° C (# days)	5	1	0	0	7	21
Total Reward	365,969	360,337	353,161	350,052	350,561	420,065

Tables 5.11 and 5.12 show that agent performance, as measured by the total reward for a model simulation, is generally better with a discount rate in the range of γ = 0.5 and γ = 0.8. A discount rate of γ = 0.8 performs better on the average during training, but the agent operating with a discount rate of γ = 0.5 performed slightly better on the best simulation run during testing.

6 – SUMMARY AND CONCLUSIONS

6.1 – Summary

Water quality issues often exist in complex and highly controlled water resources systems. Water quality downstream of reservoirs can be a particular concern and challenge for water managers, and increasing focus has been placed on the improvement of downstream water quality through alternative operating strategies for reservoir systems. Past optimization efforts have largely focused on the use of more traditional methods and technologies for the development of improved operational strategies. These methods are often encumbered by high computational intensity, and the need for development of complex statistical models of the dynamics for a particular river and reservoir environment. More recently, reinforcement learning methodologies have emerged in the field of water resources for the development of reservoir operations strategies. One significant advantage of reinforcement learning is the ability to learn the dynamics of a particular environment through simulated experience, rather than requiring up-front knowledge or development of complex transition models of the environment.

A system was developed based on reinforcement learning methodologies for the purpose of developing strategies for the improvement of water quality downstream of reservoirs using water stored for that specific goal. The system was applied to a case study on the Truckee River in California and Nevada. The system was used to produce basic release policies to improve downstream temperatures to make the river more habitable by threatened and endangered fish populations. It was further utilized to develop long-term strategies for the use of the stored water that mitigate water quality issues over a number of years, focusing largely on the mitigation of water quality issues during drought periods. Several different variations and options for the application of reinforcement learning were evaluated. Efficiency and generalization techniques to reduce computational intensity were developed and tested.

6.2 – Conclusions

Based on the system developed and the results obtained from the Truckee River case study, the primary objectives of the study were met. The objective of creating a generalized reinforcement learning-based mechanism for the development of water release strategies for the improvement of downstream water quality was met through the development of the system detailed in Section 3.2. The usefulness of the system was illustrated through its application to the Truckee River case study, as discussed in Chapter 4. The system's capabilities shown in Chapter 5 met the other study objectives of having flexibility to deal with changing conditions, and working with preexisting hydrologic and water quality models.

6.2.1 – Conclusions on the Truckee River Case Study

The results in Chapter 5 show that in order to be useful as part of a long-term water quality improvement strategy, the water acquired under the WQSA will need to be able to be stored through some other mechanism than a strict exchange for fish recovery water held back as a result of the WQSA instream flows. A different mechanism may become available under future conditions if TROA is implemented, but until that time, it may be necessary for the WQSA parties to work with the team managing the fish recovery water to devise an alternative mechanism that allows the water to be more fully stored as was the case with the more successful long-term test runs illustrated in the results of this study. In addition, the water will need to have more protection from displacement and spill in the reservoirs, as shown by the results of this study. Because the water for fishery purposes is managed by several of the same parties that will be managing the WQSA water, and the purposes of that water are somewhat connected to those of the WQSA water, it is theoretically possible that some of the concepts of WQSA storage and protection utilized in the more successful long-term strategies shown in this study might be feasible in actual practice, at least in some form. This study shows that with those concepts and flexibilities in place, it may be possible to achieve highly improved water quality conditions during times of drought, providing significant improvements over current operating strategies. The study also shows that the ability to mitigate conditions during a drought is heavily dependent not only on the ability for the water to avoid displacement or spill, but also on the amount of water that is protected.

This study concludes that, in the Truckee River case, not enough water is available to always address the immediate water quality needs of the river. For this reason, longer-term strategies require that some of the immediate downstream needs may need to be foregone in order to store water to deal with longer-term drought requirements. When that is permitted to take place, reinforcement learning was shown to be a useful tool for generating effective long-term strategies. The study shows that the total amount of WQSA water available in storage and the projected peak storage in Lake Tahoe are useful indicators in deciding how to best manage the WQSA water from year to year as part of a long-term strategy.

6.2.2 – Conclusions on Generalized Reinforcement Learning Strategies

The results in Chapter 5 show that the reinforcement learning agent system can be used to develop both short-term immediate reward strategies for the improvement of water quality downstream of reservoirs, as well as long-term strategies that provide added protection from more severe water quality issues that occur during a drought. This study shows a beneficial combination of the two ways of using the system, by both generating one or more short-term strategies, and then using those for the development of an overall long-term strategy. In this manner, the short-term and long-term policies generated by the reinforcement learning system can be integrated to achieve the best overall result.

The use of both historic and synthetic hydrologic datasets in this study show that the reinforcement learning approach and the agent design presented in Chapter 3 have the ability to easily adapt to changing conditions. The adjustment of reward functions between the short-term and long-term agent illustrates how the agent's behavior can be adjusted depending on the overall desired goals for the agent.

Function approximation through the use of neural networks was shown to be a viable alternative to the tabular approach in the case of the long-term agent. However, issues that arose on the use of neural networks with the short-term agent reveal additional research and development that will need to take place before the agent design presented will be able to fully utilize the neural network approximation for more complex action-value functions.

For the tests conducted, it was shown that both SARSA and Q-Learning can be almost equally effective at producing long-term improved strategies. Eligibility traces were shown to be more useful with SARSA, and in all cases, middle to higher range discount rates were found to be more effective.

This study provided a successful example of the use of reinforcement learning in the field of water resources, specifically addressing the issue of water quality. Due to the generalized nature of the reinforcement learning mechanism, and its ability to interactively learn the underlying stochastic nature of a river and reservoir system without requiring a complex probabilistic model, the system presents a viable alternative to more traditional optimization methods for the development of improved reservoir operating strategies for water quality purposes.

6.3 – Implementation Findings

During the course of the development of the water quality reinforcement learning agent, as well as the testing and evaluation of the agent on the Truckee River case study, a number of practical implementation issues were identified and addressed. A discussion of those issues and the solution approaches that were used is contained in Appendix E. It is expected that the information presented in Appendix E might aid researchers as well as practitioners encountering similar issues in future studies and applications.

6.4 – Future Work

6.4.1 – Future Work on Truckee River and WQSA Issues

A large number of options for reinforcement learning techniques were introduced to the reinforcement learning system described in Section 3.2.5. In addition, a variety of options for the use of neural networks as function approximations were introduced. The number of possible combinations and permutations of the different options, including their different possible settings and ranges of values, are immense. Future work should include an evaluation of the combinations of different options, and an extensive sensitivity analysis of the settings for each of the variables and parameters available to determine the optimal combinations and settings.

Due to the fact that complex water quality modeling was outside the scope of this study, coupled with the difficulties encountered with the linkage of the TrHSPF water quality model to the hydrologic and reservoir operations model, this study focused on the single water quality constituent of temperature. Future studies should expand the focus of the development of policies for the use of the WQSA water to include consideration of other water quality variables as well. These should include, at a minimum, dissolved oxygen and total dissolved solids.

Future work should include the evaluation of alternative state representations provided to the reinforcement learning agent. It is possible that improved agent performance may stem from providing the agent with different variables and indicators of the state of the Truckee River and reservoir system. In addition, different reward structures should be researched and tested. The current study utilized reward structures based around daily water quality improvement, and overall annual performance. It is possible that other reward mechanisms might motivate the agent towards improved performance. One such reward mechanism might include additional state variables indicating an imminent violation in the absence of any action by the water quality agent. The agent would receive a "0" variable value if no violations are predicted, and a "1" value if the temperature is either considered too high, or has been high for enough days to trigger a violation. The reward mechanism would then only provide a reward signal if an actual violation had occurred. For this "violation avoidance" case, Equations 4.1 and 4.3 in Section 4.5.3 would be replaced with something similar to Equations 6.1 and 6.2 presented below:

$$r_x(s,a) = \begin{cases} [b_x * d_x]^{c_x} & if\ t_{Reno} > t_{absolute} \\ [b_x * e_x]^{c_x} & if\ \#Days_{acute} \leq DayLimit_{acute} \\ [b_x * f_x]^{c_x} & if\ \#Days_{chronic} \leq DayLimit_{chronic} \end{cases} \qquad \text{(Eqn 6.1)}$$

where:

$d_x = Absolute\ temperature\ violation\ penalty\ constant$

$e_x = Acute\ temperature\ and\ timespan\ violation\ penalty\ constant$

$f_x = Chronic\ temperature\ and\ timespan\ violation\ penalty\ constant$

$t_{absolute} = Absolute\ maximum\ temperature\ (See\ Section\ 4.5.2.3)$

$\#Days_{acute} = \#\ days\ temp.exceeded\ t_{chronic}, but\ not\ t_{absolute}$

$\#Days_{chronic} = \#\ days\ temp.exceeded\ t_{preferred}, but\ not\ t_{chronic}$

$DayLimit_{acute} = \#\ survivable\ days\ in\ acute\ temperature\ range$

$DayLimit_{chronic} = \#\ survivable\ days\ in\ chronic\ temperature\ range$

The remaining coefficients in the equation are the same as those shown for Equation 4.1

$$r_y(s,a) = \begin{cases} \left[b_y * \left(t_{target} - t_{Reno}\right)\right]^{c_y} & if\ t_{Reno} \leq\ t_{target}\ and\ a_{t-1} > 0 \\ \left[b_y * d_y\right]^{c_y} & if\ no\ violation\ and\ a_{t-1} > 0 \\ 0 & otherwise \end{cases} \qquad \text{(Eqn 6.2)}$$

where:

$d_y = Unnecessary\ release\ penalty\ constant$

The current study was based on the operating policies for the Truckee River basin as they exist today. The likely future policy under the TROA will change the system dynamics, and require alternative operating strategies. In addition, the current study illustrated that in order for the WQSA water to be useful in a longer-term strategy, a certain amount of the storage will need to be more secure from spill. Future work should focus on evaluating strategies for the use of the WQSA water under the TROA policy, as well as strategies for providing more secure storage through trades and exchanges within the context of the TROA.

The reinforcement learning system developed in Section 3.2.5 is generalized to the point that it can be applied to any optimization task, and is not limited to use for water quality purposes. Under the future TROA operating policy, there will be a wide variety of different accounts of water being managed by many entities. Future work should focus on utilizing the system developed to generate operational strategies for those other types of water to meet their intended purposes. In addition, since operation under TROA will represent a significant multi-purpose and multi-agent operation, research should focus on utilizing the system in a multi-agent capacity. This will shed new light on the ability to conduct multi-purpose optimization on a river system through the use of a multi-agent reinforcement learning structure. This might be similar to the approach noted in Mariano-Romero et al. (Mariano-Romero 2007).

The hydrologic data provided to the agent in this study was primarily a 30-year dataset, as well as a limited number of other synthetic datasets. Future research should incorporate different hydrologic scenarios, including those that simulate climate change. In addition, the use of stochastically generated hydrology in the training of the agent should be evaluated, to potentially discover the best set of training hydrology that promotes optimal agent behavior. Such an evaluation might be similar to that developed by Bouchart et al. (Savic and Walters 1999).

6.4.2 – Future Work on Reinforcement Learning System

The water quality reinforcement learning agent system described in Section 3.2.5 was implemented in a highly generalized fashion, to the point that it is almost immediately applicable to any other MDP where a model of the environment is available. The system is not limited to water quality applications, or even water resources applications in general. The only component of the system that requires adjustment to apply the system to any other decision process is the “interface program” described in Section 3.2.5. That program would need to be adjusted to allow the system to interface with a different environment simulation model for a given decision process. Future work on the system should initially include the creation of interface programs to allow the system to connect to a wider variety of hydrologic and reservoir operations models, such as MODSIM, HEC-RESSIM, WRIMS (CalSim) and other widely used generalized modeling systems. The system should be applied to different issues on other river systems. Future work should also include extending the system to applications in other water resources related fields of study, and to applications outside the field of water resources.

Generalization through function approximation presents certain advantages as discussed in Section 3.1.3. However, issues were encountered during this study with the use of neural networks for function approximation. Continued development is needed to deal with issues involving the use of the neural network function approximation within the GPI structure developed for the agent in Section 3.2.5, particularly with respect to improving agent behavior when working with less precise estimates of the action-value function. This future development should initially involve improvements to the neural network approximations, and might focus particularly on the issue of scaling the input and output training data when that training data is constantly in a state of flux while being updated by the agent. Additional research might also include addressing issues with the interactions between the function approximation and the use of gradually reducing step sizes and eligibility traces, which were observed in this study to cause degradation in the quality of the neural network training data.

For the improvement of the neural network function approximations, future development on the system described in Section 3.2.5 should include implementation of cross-validation capability for the neural network training functions of the system. In addition, traditional error back-propagation training methods should be implemented as an alternative to the genetic algorithm training mechanism currently employed. Tests should be conducted to identify the best approach to balancing the number of members in the genetic algorithm solution population with the speed and efficiency of the genetic algorithm network training. Other neural network training and configuration options might be added as options to the system, such as the ability to use recurrent network structures. The ability to run sensitivity analyses on just the neural network training component of the system, rather than the entire reinforcement learning agent system, should be added.

For the agent design discussed in Section 3.2.5, efficiency mechanisms were used to improve the performance of the tabular implementation. These included a binary search algorithm for finding table values in files, as well as the utilization of single files for each state rather than a complete database for the tracking of action exploration and average immediate rewards. Some testing was done to identify efficiency gains that would occur through the use of binary files rather than text files. These gains were found to be marginal, but future work should include the full addition of that option to the system to take advantage of even small efficiency improvements. In addition, future work should include the use of improved database systems for efficiency improvements. These systems could be as simple as single files for each state containing the action-value and policy functions rather than complete database files, or as complex as higher-level database storage systems. For the case where single text or binary file systems are used, the possibility of other improved search algorithms should be explored.

Future work on the system should also include expansion of the distributed computing capabilities already implemented. These would include the capability to initiate more environment simulation models for systems with additional memory and CPU capabilities, the ability to initiate simulation model runs on remote computers, and the ability to incrementally add simulation model runs to a run sequence currently underway. Multithreading capability should be added to the process controller, and the other programs should be built-in to the process controller to allow a user to either run the program as a standalone executable, or directly from within the process controller. The ability to examine more complex issues through more massively distributed/parallel computing should be explored using the system.

The policy update program described in Section 3.2.5 currently allows the user to either conduct policy updates at a given interval of policy evaluation steps, or when the action-value function stabilizes. Future studies should include construction of improved criteria for the system to determine when the policy update should be conducted.

Future work should utilize the system to conduct more extensive comparisons of the various technological options provided to the user by the system. These comparisons would more fully illustrate the differences between the learning capabilities of the various options. Comparisons would include SARSA against Q-Learning, the use of eligibility traces, the specific use of accumulating traces against replacing traces, and a full comparison of the effectiveness of the various exploration and action-selection alternatives provided by the system.

REFERENCES

(1915). United States of America v. Truckee River General Electric Company, District Court of the United States, Northern District of California, Second Division.

(1944). United States v. Orr Water Ditch Company, et al., United States District Court, District of Nevada.

Abolpour, B. (2007). "Water allocation improvement in river basin using Adaptive Neural Fuzzy Reinforcement Learning approach." APPLIED SOFT COMPUTING 7(1): 265-285.

Andrews, A. K., R. A. Ellison, et al. (1979). Application of the Adaptive Environmental Assessment Methodology to the Truckee-Carson River Quality Assessment. Fort Collins, CO, U.S. Fish and Wildlife Service.

ASCE (2000). "Artificial neural networks in hydrology. I: Preliminary concepts." Journal of Hydrologic Engineering [J. Hydrol. Eng.]. Vol. 5 5(2): 115-123.

ASCE (2000). "Artificial neural networks in hydrology. II: Hydrologic applications." Journal of Hydrologic Engineering [J. Hydrol. Eng.]. Vol. 5 5(2): 124-137.

Association, A. W. W. (1987). Truckee River Water Quality Monitoring. The 1987 AWWA Water Quality Technology Conference. Baltimore, MD.

Berris, S. N. (1996). Daily Flow-Routing Simulations for the Truckee River, California and Nevada; Water-Resources Investigations Report 96-4097. Carson City, NV, U.S. Geological Survey.

Berris, S. N., G. W. Hess, et al. (2001). River and Reservoir Operations Model, Truckee River Basin, California and Nevada, 1998; Water-Resources Investigations Report 01-4017. Carson City, NV.

Berris, S. N., G. W. Hess, et al. (1996). Simulation of Selected Reservoir Operations in the Upper Truckee River Basin, California; Fact Sheet 082-96. Carson City, NV, U.S. Geological Survey.

Bertsekas, D. P. (1996). Neuro-dynamic programming. Belmont, Mass., Athena Scientific.

Bhattacharya, B. (2003). "Neural networks and reinforcement learning in control of water systems." JOURNAL OF WATER RESOURCES PLANNING AND MANAGEMENT-ASCE 129(6): 458-465.

Bhimani, S., S. McDonald, et al. (2002). Beyond TMDLs on the Truckee River: Implementation of a Water Quality Trading Program Through a Phased Permitting Approach. National TMDL Science and Policy 2002 Specialty Conference, Water Environment Federation.

Bishop, C. M. (2006). Pattern recognition and machine learning. New York, Springer.

Bohman, L. R. (1996). The Truckee-Carson Program; USGS Surface-water quality and flow Modeling Interest Group Feature Article. Accessed 2005 - smig.usgs.gov.

Bohman, L. R., S. N. Berris, et al. (1995). Interactive Computer Program to Simulate and Analyze Streamflow, Truckee and Carson River Basins, Nevada and California; Fact Sheet 165-95. Carson City, NV, U.S. Geological Survey.

Bouchart, F. J. C. (1998). "A reinforcement learning model for the operation of conjunctive use schemes." HYDRAULIC ENGINEERING SOFTWARE VII 4: 319-329.

Boyer, J. T. (2006). Interagency Coordination for Truckee and Carson River Operations. Proceedings of the 2006 Operations Management Conference, ASCE. Sacramento, CA.

Boyer, J. T. (2006). Water Accounting in the Truckee Basin RiverWare Model. Proceedings of the 2006 Federal Interagency Hydrologic Modeling Conference, Reno, NV, Federal Interagency Committee on Hydrology.

Boyer, J. T. (2006). "Water Supply Accounting and Categorization in the Truckee Basin Using the RiverWare Model." Journal of the Nevada Water Resources Association 3(1).

Brock, J. T. (2000). Data Availability for Modeling Water Quality in the Truckee River Basin; Draft Technical Memorandum prepared for Carollo Engineers. Boise, ID.

Brown, W. M., J. O. Nowlin, et al. (1986). River-Quality Assessment of the Truckee and Carson River System, California and Nevada -- Hydrologic Charactaristics. Open-File Report 84-576. Sacramento, CA, U.S. Geological Survey.

Buckland, M. (2010). Genetic Algorithms in Plain English. 2010.

Buckland, M. (2010). Neural Networks in Plain English.

Castelletti, A. (2002). "Reinforcement learning in the operational management of a water system." MODELING AND CONTROL IN ENVIRONMENTAL ISSUES 2001: 303-308.

Castelletti, A., G. Garbarini, et al. (2009). Reinforcement Learning Control of Selective Withdrawl Reservoirs Accounting for Both Quality and Quantity Targets. 18th World IMACS/MODSIM Congress. Cairns, Australia.

Caupp, C. L., J. T. Brock, et al. (1991). Application of the Dynamic Stream Simulation and Assessment Model (DSSAM III) to the Truckee River Below Reno, Nevada: Model Formulation and Program Description. Submitted to Nevada Division of Environmental Protection, Carson City and Washoe County Department of Comprehensive Planning, Reno. Boise, ID, Rapid Creek Water Works: 129.

Chaves, P. (2004). "Operation of storage reservoir for water quality by using optimization and artificial intelligence techniques." MATHEMATICS AND COMPUTERS IN SIMULATION 67(4-5): 419-432.

Chaves, P. and T. Kojiri (2007). "Deriving reservoir operational strategies considering water quantity and quality objectives by stochastic fuzzy neural networks." Advances in Water Resources [Adv. Water Resour.]. Vol. 30 30(5): 1329-1341.

Chaves, P. and T. Kojiri (2007). "Stochastic Fuzzy Neural Network: Case Study of Optimal Reservoir Operation." Journal of Water Resources Planning and Management [J. Water Resour. Plann. Manage.]. Vol. 133 133(6): 509-518.

Chen, C. (2002). Development of TMDL Implementation Plan with Consensus Module of WARMF. National TMDL Science and Policy 2002 Specialty Conference, Water Environment Federation.

Chen, H. W. (1998). "Water pollution control in the river basin by fuzzy genetic algorithm-based multiobjective programming modeling." WATER SCIENCE AND TECHNOLOGY 37(8): 55-63.

Chiatovich, T. D. and J. W. Fordham (1979). "A Reservoir Operating Model with Inorganic Quality Constraints for the Truckee River." Water Resources Bulletin 15(2): 301-315.

Cobb, E. D., A. F. Olson, et al. (1990). Review of Selected Water-Management Models and Results of Simulations for the Truckee-Carson Rivers System, California and Nevada; Open-File Report 90-393. Reson, VA, U.S. Geological Survey.

Colby, B. G., M. A. McGinnis, et al. (1991). "Mitigating Environmental Externalities Through Voluntary and Involuntary Water Reallocation: Nevada's Truckee-Carson River Basin." Natural Resources Journal 31(Fall, 1991): 757-783.

Coors, S. (2006). An Overview of the Truckee-Carson RiverWare Modeling System. Proceedings of the 2006 Operations Management Conference, ASCE. Sacramento, CA.

Coors, S. (2006). "Simulating Operations in the Truckee-Carson RiverWare Modeling System." Journal of the Nevada Water Resources Association 3(1).

Coors, S. (2006). Truckee-Carson Basin RiverWare Operations Model. Proceedings of the 2006 Federal Interagency Hydrologic Modeling Conference, Reno, NV.

Coors, S. (2010). Planning Hydrology Worksheet, Precision Water Resources Engineering, LLC.

Darsono, S. (2007). "Neural-optimal control algorithm for real-time regulation of in-line storage in combined sewer systems." ENVIRONMENTAL MODELLING & SOFTWARE 22(9): 1349-1361.

Engineers, C. (2003). TMDLs and Watershed Modeling: Pollutant Trading on the Truckee River Basin. www.carollo.com. C. Engineers.

Facility, T. M. W. R. (2004). Truckee Meadows Water Reclamation Facility River Monitoring Data, TMWRF. 2004.

Federal Water Master's Office, U. D. C. (2010). Instream Flow Deliveries over Derby Dam - Permit Summaries. Reno, NV: 1.

Fordham, J. W. (1972). Simulation Theory Applied to Water Resources Management Phase III; Development of Optimal Operating Rules. Reno, NV, Desert Research Institute.

Fritchel, P. F. (2009). Documentation for Hydrologic Data. Carson City, NV, Bureau of Reclamation.

Galat, D. L. (1990). "Seasonal and Long-Term Trends in Truckee River Nutrient Concentrations and Loadings to Pyramid Lake, Nevada: A Terminal Saline Lake." Water Research 24(8): 1031-1040.

Great Basin Land and Water, N.-p. W. R. P. O. (2011). Records of Water Rights Purchased under the WQSA (personal communication). Carson City, NV.

Hastie, T., R. Tibshirani, et al. (2001). The elements of statistical learning data mining, inference, and prediction : with 200 full-color illustrations. New York, Springer.

Hoffman, R. J. (1990). Phosphorus in the Truckee River Between Vista and Patrick, Storey and Washoe Counties, Nevada, August 1984. Water-Resources Investigation Report 89-4175. Carson City, NV, U.S. Geological Survey.

Hoffman, R. J. and G. G. Scoppettone (1989). Effect of Water Quality on Survival of Lahontan Cutthroat Trout Eggs in the Truckee River, West-Central Nevada and Eastern California. Water Supply Paper 2319. Carson City, NV, U.S. Geological Survey and U.S. Fish and Wildlife Service.

Israel, M. S. (1996). Modeling Water Resources Management Institutions: An Application to the Truckee-Carson River System. Civil and Environmental Engineering. Davis, CA, University of California, Davis.

Israel, M. S. and J. R. Lund (1999). "Priority Preserving Unit Penalties in Network Flow Monitoring." Journal of Water Resources Planning and Management 125(4): 205-214.

Jeton, A. E. (2000). Precipitation-Runoff Simulations for the Upper Part of the Truckee River Basin, California and Nevada; Water-Resources Investigations Report 99-4282. Carson City, NV, U.S. Geological Survey.

Kerachian, R. and M. Karamouz (2007). "A stochastic conflict resolution model for water quality management in reservoir-river systems." Advances in Water Resources 30(4): 866-882.

Labadie, J. W. (1998). "Reservoir Systems Optimization Models." Water Resources Update Journal(107).

Labadie, J. W. (2004). "Optimal Operation of Multireservoir Systems: State-of-the-Art Review." Journal of Water Resources Planning and Management 130(2): 93-111.

Labadie, J. W. (2005). "Closure to "Optimal operation of multireservoir systems: State-of-the-art review" by John W. Labadie." JOURNAL OF WATER RESOURCES PLANNING AND MANAGEMENT-ASCE 131(5): 407-408.

Labadie, J. W. and M. L. Baldo (2001). "Priority Preserving Unit Penalties in Network Flow Modeling; Discussion." Journal of Water Resources Planning and Management 125(5): 67-68.

Lee, J. H. and J. W. Labadie (2007). "Stochastic optimization of multireservoir systems via reinforcement learning." WATER RESOURCES RESEARCH 43(11).

Lobbrecht, A. H. and D. P. Solomatine (2002). "Machine learning in real-time control of water systems." Urban Water [Urban Water]. Vol. 4 4(3): 283-289.

Lovell, S. (2000). "Using water markets to improve environmental quality: Two innovative programs in Nevada." Journal of Soil and Water Conservation 55(1): 19-26.

Lumb, A. M., J. L. Kittle, et al. (1990). Users manual for ANNIE, a computer program for interactive hydrologic analyses and data management. Reston, Va., Dept. of the Interior, U.S. Geological Survey
Books and Open-File Reports Section [distributor].

Mahootchi, M. (2007). "Opposition-based reinforcement learning in the management of water resources." 2007 IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning: 217-224.

Mahootchi, M., H. R. Tizhoosh, et al. (2007). Reservoir Operation Optimization by Reinforcement Learning. Contemporary Modeling of Urban Water Systems, Monograph 15. W. James. Guelph, ON Canada, CHI.

Maier, H. R. (1996). "The use of artificial neural networks for the prediction of water quality parameters." WATER RESOURCES RESEARCH 32(4): 1013-1022.

Maier, H. R. (2000). "Neural networks for the prediction and forecasting of water resources variables: a review of modelling issues and applications." ENVIRONMENTAL MODELLING & SOFTWARE 15(1): 101-124.

Mann, M. (2006). Hydrologic Forecasting in the Truckee-Carson RiverWare System. Proceedings of the 2006 Federal Interagency Hydrologic Modeling Conference, Reno, NV.

Mann, M. (2006). "Hydrologic Forecasting in the Truckee-Carson RiverWare System." Journal of the Nevada Water Resources Association 3(1).

Mariano-Romero, C. E. (2007). "Multi-objective optimization of water-using systems." EUROPEAN JOURNAL OF OPERATIONAL RESEARCH 181(3): 1691-1707.

McDonald, S., S. Bhimani, et al. (2000). A Case Study of Pollutant Load Trading on the Truckee River. Proceedings of Watershed Management 2000, Ft. Collins, CO, ASCE/EWRI.

McDonald, S., S. Bhimani, et al. (2000). Decision Support Process for Truckee River Watershed Management. Proceedings of Watershed Management 2000, Ft. Collins, CO.

Mohammadian, M., R. A. Sarker, et al. (2003). Computational intelligence in control. Hershey, Idea Group Pub.

Nelson, T. (2006). Comparison of RiverWare Model Results to Actual Flows, Department of Water Resources, State of California.

Neumann, D. (2001). An Operations Model for Temperature Management of the Truckee River Above Reno, Nevada (M.S. Thesis). Civil Engineering. Boulder, University of Colorado: 121.

Neumann, D., B. Rajagopalan, et al. (2003). "Regression Model for Daily Maximum Stream Temperature." Journal of Environmental Engineering 129(7): 667-674.

Neumann, D. W. (2006). "A decision support system to manage summer stream temperatures." JOURNAL OF THE AMERICAN WATER RESOURCES ASSOCIATION 42(5): 1275-1284.

Nevada Division of Water Planning, D. o. C. a. N. R. (1997). Truckee River Chronology. Carson City, NV, Nevada Div. of Water Planning.

Nevada, S. o. (2010). Nevada Administrative Code. NAC 445A.184-190. S. o. Nevada. NAC 445A.184-190.

Nowlin, J. O. (1987). Documentation for a Digital Computer Model of Nutrient and Dissolved-Oxygen Transport in the Truckee River and Truckee Canal Downstream from Reno, NV. Open-File Report 87-554. Carson City, NV, U.S. Geological Survey.

Otero, J. M. (1995). "Optimization of managed runoff to the St Lucie estuary." WATER RESOURCES ENGINEERING, VOLS 1 AND 2: 1506-1510.

Pohll, G., D. McGraw, et al. (2001). Evaluation of Groundwater and Solute Transport in the Fernley-Wadsworth Area. Publication No. 41173. Reno, NV, Desert Research Institute.

Pratt, J. (1997). Truckee-Carson River Basin Study Final Report. Report to the Western Water Policy Review Advisory Commission. Seattle, WA, Clearwater Consulting Corporation.

Protection, N. D. o. E. (1994). Truckee River Final Total Maximum Daily Loads and Waste Load Allocations. Carson City, NV, Nevada Div. of Env. Protection, Bureau of Water Quality Planning.

Raman, H. (1996). "Deriving a general operating policy for reservoirs using neural network." JOURNAL OF WATER RESOURCES PLANNING AND MANAGEMENT-ASCE 122(5): 342-347.

Rani, D. and M. Moreira (2010). "Simulation–Optimization Modeling: A Survey and Potential Application in Reservoir Systems Operation." Water Resources Management 24(6): 1107-1138.

Reclamation, B. o. (1997). Operating Criteria and Procedures for the Newlands Project, Nevada. B. o. Reclamation. 43 CFR 418.

Reclamation, B. o. (2010). Projects and Facilities Database.

Reno, C. o., C. o. Sparks, et al. (2003). Regional Wastewater Facilities Master Plan. Reno, NV.

Reno, C. o., C. o. Sparks, et al. (1996). Truckee River Water Quality Settlement Agreement.

Resources, C. D. o. W. (1991). Truckee River Atlas.

Rieker, J. D. (2005). Documentation for Hydrologic Data. Carson City, NV, Bureau of Reclamation.

Rieker, J. D. (2006). Decision Support for Water Quality Releases on the Truckee River. Proceedings of the 2006 Federal Interagency Hydrologic Modeling Conference, Reno, NV.

Rieker, J. D. (2006). Stakeholder Interaction Methods to Present Results of the Truckee-Carson Models. Proceedings of the 2006 Operations Management Conference. Sacramento, CA.

Rieker, J. D., S. Coors, et al. (2005). Modeling in Support of Water Operations in the Truckee River Basin. Proceedings of Watershed Management 2005, Williamsburg, VA, EWRI and ASCE.

Risley, J. C., E. A. Roehl, et al. (2003). "Estimating Water Temperature in Small Streams in Western Oregon Using Neural Network Models." Water Resources Investigations Report. United States Geological Survey [Water Resour. Invest. Res. U.S. Geol. Surv.]. no. 4218(4218).

Roehl, E. A., P. A. Conrads, et al. (2000). Real-Time Control of the Salt Front in a Complex, Tidally Affected River Basin. ANNIE 2000. St. Louis, MO.

Rounds, S. (2002). Development of a Neural Network Model for Dissolved Oxygen in the Tualatin River, Oregon. Second Federal Interagency Hydrologic Modeling Conference. Las Vegas, NV.

Rowell, J. H. (1975). Truckee River Temperature Prediction Study. Sacramento, CA, Bureau of Reclamation.

Russell, S. J. and P. Norvig (2003). Artificial intelligence a modern approach. Upper Saddle River, N.J., Prentice Hall/Pearson Education.

Savic, D.-. and G.-. Walters (1999). Water industry systems modelling and optimization applications. Baldock, Hertfordshire, England ;, Research Studies Press.

SCOPPETTONE, G. G., M. COLEMAN, et al. (1983). "Reproduction by the Endangered Cui-ui in the Lower Truckee River." Transactions of the American Fisheries Society 112(6): 788-793.

Scott, T. (2006). Review of Truckee River Operating Agreement Negotiations and Related Modeling Efforts. Proceedings of the 2006 Operations Management Conference, ASCE. Sacramento, CA.

Sharp, J. V., R. L. Bateman, et al. (1970). Digital Simulation Model of Inorganic Water Quality of Tahoe-Truckee System, Nevada-California. Reno, NV, Desert Research Institute.

Shirangi, E., R. Kerachian, et al. (2008). "A simplified model for reservoir operation considering the water quality issues: Application of the Young conflict resolution theory." Environmental Monitoring and Assessment 146(1): 77-89.

Solomatine, D. P. (1996). "Neural network approximation of a hydrodynamic model in optimizing reservoir operation." HYDROINFORMATICS '96, VOLS 1 AND 2: 201-206.

States, U., S. o. California, et al. (2008). Truckee River Operating Agreement. California/Nevada.

States, U., T. C. I. District, et al. (1935). Truckee River Agreement. California/Nevada.

Suen, J. P. (2003). "Evaluation of neural networks for modeling nitrate concentrations in rivers." JOURNAL OF WATER RESOURCES PLANNING AND MANAGEMENT-ASCE 129(6): 505-510.

Sutton, R. S. and A. G. Barto (1998). Reinforcement learning an introduction. Cambridge, Mass., MIT Press.

Szepesvári, C. (2010). Algorithms for reinforcement learning. San Rafael, Calif. (1537 Fourth Street, San Rafael, CA 94901 USA), Morgan & Claypool.

Taylor, R. L. (1998). Simulation of hourly stream temperature and daily dissolved solids for the Truckee River, California and Nevada, U.S. Geological Survey Water-Resources Investigations Report 98-4064: 70.

Tizhoosh, H. R.-. and M. Ventresca (2008). Oppositional concepts in computational intelligence. Berlin, Springer-Verlag.

U.S. Army Corps of Engineers (1995). Lower Truckee River, Nevada; Reconnaissance Report.

U.S. Environmental Protection Agency Office of Water (1994). TMDL Case Study: Truckee River, Nevada.

U.S. Geological Survey (1996). Environmental and Hydrologic Settings of the Las Vegas Valley Area and the Carson and Truckee River Basins, Nevada and California, U.S. Geological Survey.

U.S. Geological Survey (1997). Water-Quality Assessment of the Las Vegas Valley Area and the Carson and Truckee River Basins, Nevada and California - Nutrients, Pesticides, and Suspended Sediment, October 1969-April 1990; Water-Resources Investigations Report 97-4106, U.S. Geological Survey.

United States Department of the Interior, B. o. I. A. (2002). Final Environmental Impact Statement - Truckee River Water Quality Settlement Agreement - Federal Water Rights Acquisition Program. Phoenix, AZ.

USDOI (2002). Record of decision Truckee River Water Quality Settlement Agreement - Federal Water Rights Acquisition program (California and Nevada). [Phoenix, AZ], U.S. Dept. of the Interior, Bureau of Indian Affairs, Western Regional Office.

USDOI (2004). Truckee River operating agreement, California and Nevada revised draft environmental impact statement/environmental impact report. [Washington, D.C.], U.S. Dept. of the Interior
State of California, Dept. of Water Resources.

USDOI (2008). Final environmental impact statement, environmental impact report : Truckee River operating agreement. [Washington, D.C.], U.S. Dept. of the Interior
State of California, Dept. of Water Resources.

USGS (2010). USGS National Water Information System: Web Interface.

Wagner, P. and M. Lebo (1996). "Managing the resources of Pyramid Lake, Nevada, amidst competing interests." Journal of Soil and Water Conservation.

Wan, Y., J. W. Labadie, et al. (2006). "Optimization of Frequency Distribution of Storm-Water Discharges for Coastal Ecosystem Restoration." Journal of Water Resources Planning and Management 132(5): 320-329.

Warwick, J. J., D. Cockrum, et al. (1999). "Modeling the Impact of Subsurface Nutrient Flux on Water Quality in the Lower Truckee River, Nevada." Journal of American Water Resources Association 35(4): 837-851.

Wen, C. G. (1998). "A neural network approach to multiobjective optimization for water quality management in a river basin." WATER RESOURCES RESEARCH 34(3): 427-436.

Wilson, G. (1996). "Reinforcement learning: A new technique for the real-time optimal control of hydraulic networks." HYDROINFORMATICS '96, VOLS 1 AND 2: 893-900.

Wurbs, R. A. (1993). "RESERVOIR-SYSTEM SIMULATION AND OPTIMIZATION MODELS." JOURNAL OF WATER RESOURCES PLANNING AND MANAGEMENT-ASCE 119(4): 455-472.

Wurbs, R. A. (2005). Comparative Evaluation of Generalized River/Reservoir Systems Models. College Station, Texas, Texas Water Resources Institute.

Yeh, W.-G. (1985). "Reservoir Management and Operations Models: A State-of-the-Art Review." Water Resources Research WRERAQ Vol. 21 21.

Yi, C.-h. (2005). Basin-wide multi-reservoir operation using reinforcement learning.

Zagona, E. A. (2001). "Riverware: A generalized tool for complex reservoir system modeling." JOURNAL OF THE AMERICAN WATER RESOURCES ASSOCIATION 37(4): 913-929.

APPENDIX A – TRUCKEE RIVER GOVERNING DOCUMENTS

Water storage and flow in the Truckee River have historically and are currently governed by a number of court decrees, court decisions, agreements, laws, and regulations. These include the Truckee River General Electric Decree, Truckee River Agreement, Orr Ditch Decree, Tahoe-Prosser Exchange Agreement, Carson-Truckee Water Conservancy District v. Watt et al., Memorandum of Agreement-Truckee River Water Management, Pyramid Lake Tribe of Indians v. Morton, Newlands Project Operating Criteria and Procedures (OCAP), and Interim Storage Agreement. In addition, most of the reservoirs contain flood operating guidelines set forth by the United States Army Corps of Engineers or the California Department of Water Resources' Division of Safety of Dams. Because the above referenced documents directly refer to exact quantities and flow rates of water in imperial units, that system of units is utilized as the primary system of units in this appendix rather than metric units.

<u>Truckee River General Electric Decree</u>

The Truckee River General Electric Decree is a final judgment and decree issued in *United States of America vs. The Truckee River General Electric Company* in 1915. The case involved the condemnation of the lands, dam, and controlling works at Lake

Tahoe. The decree permitted the United States to control the dam at Lake Tahoe, but required that the flow in the Truckee River be maintained at a rate of 500 cubic feet per second (cfs) from March 1 through September 30 of each year. From October 1 through the last day in February, the flow requirement was reduced to 400 cfs. These rates of flow became known as the "Floriston Rates", and were measured by a streamgage at Iceland, California, near the present day USGS streamgage at Farad California (USGS gage number 10346000). The rates of flow were based on the demands of a paper mill that no longer exists. The decree provided for special flow requirements during winter months to remove ice interfering with the power plants on the river, and reduced flow requirements when specifically requested by the Truckee River General Electric Company. The decree allowed the United States to use any water stored more than four feet above the natural rim of Lake Tahoe for its own purposes, and provided for limited use of the water below that storage level by the United States under certain conditions.

Orr Ditch Decree and Truckee River Agreement

The Orr Ditch Decree is an order, adjudgment, and decree entered in *United States of America vs. Orr Ditch Water Company, et al*. The decree was sought by the United States to adjudicate the Truckee River water rights in the state of Nevada. The decree adjudicated Truckee River water rights and adopted the Truckee River Agreement as the governing procedure for operation of the Truckee River and its reservoirs. The decree appointed a Water Master to carry out and enforce its provisions. The decree provided for use of water during any time of the year, subject to the limitation that the total amount used could not exceed applicable quantities allowed to the land. It also limited the amount of water to be used in any single month. The decree established two of the most senior rights on the Truckee River to belong to the Pyramid Lake Paiute Tribe, known as Claims 1 and 2, with a priority date of December 8, 1859. It established one of the more junior rights to belong to the United States for diversion of water to the Truckee Canal, known as Claim 3, with a priority date of July 2, 1902. The United States' right under Claim 3 provided for diversion of up to 1,500 cfs into the Truckee Canal for irrigation of 232,800 acres of land and other purposes (the maximum acreage of the project was generally modified by the United States through contract with the Truckee-Carson Irrigation District in 1926 to 87,500 acres). The decree stated that the amount of diversion was subject to the control, disposal, and regulation of the United States. Irrigation was limited to 3.5 acre-feet per acre on bottom land and 4.5 acre-feet per acre of bench land for the Newlands Project.

The Truckee River Agreement is an agreement signed in 1935 between the United States, Truckee-Carson Irrigation District, Washoe County Water Conservation District, Sierra Pacific Power Company (successor to the Truckee River General Electric Company), and other users of the river known as the "parties of the fifth part." The agreement was approved and adopted by the Orr Ditch Court in the Orr Ditch Decree. The agreement detailed the operating rules for the Truckee River and its reservoirs. Central to the agreement was the concept of the "Floriston Rates" as developed in the Truckee River General Electric Decree. Further reductions to the Floriston Rates developed in the Truckee River General Electric Decree were permitted, allowing flow to be reduced to 350 cfs between November 1 and March 31 when Lake Tahoe's elevation is more than 2.25 feet above the natural rim but below 3 feet above the natural rim, and 300 cfs if Lake Tahoe's elevation is below 2.25 feet above the natural rim. The elevation of the natural rim of the lake was determined in the agreement to exist at 6,223.0 feet above sea level. The agreement also provided for the construction of Boca Dam on the Little Truckee River. The agreement set forth the rules and priorities for storage of water in Boca Reservoir. The agreement also recognized and outlined the rules for privately owned stored water in the Truckee River system.

Tahoe-Prosser Exchange Agreement

The Tahoe-Prosser Exchange Agreement is an agreement signed in 1959 by the United States, Truckee-Carson Irrigation District, Washoe County Water Conservation District, and Sierra Pacific Power Company. The agreement provided for the construction of Prosser Creek Dam and Reservoir. The agreement also described the operating procedures for the dam and reservoir. The primary operation set forth by the agreement is an exchange of water between Lake Tahoe and Prosser Creek Reservoir, whereby water may be released from Lake Tahoe when it would not otherwise do so under the provisions of the Truckee River Agreement, in order to ensure minimum flows downstream of Lake Tahoe Dam of 50 or 70 cfs depending on the time of year. An equivalent amount of water is then stored in Prosser Creek Reservoir and becomes "Tahoe Exchange Water" for later use under the Truckee River Agreement provisions relating to Lake Tahoe as if it were water stored in Lake Tahoe. In the event that water is not available to be held back in Prosser Reservoir as the exchange is occurring, water previously stored in Prosser Reservoir is converted to "Tahoe Exchange Water" to later be used as if it were Lake Tahoe water.

Carson-Truckee Water Conservancy District v. Watt et al. and Memorandum of Agreement-Truckee River Water Management

The basic operation of Stampede Dam and Reservoir set forth by the Department of the Interior was affirmed by the judgment and opinion issued in 1983 in *Carson-Truckee Water Conservancy District v. Watt et al.* The 1983 judgment and opinion stated that the Department of the Interior must use all waters stored in Stampede Reservoir for the benefit of the Pyramid Lake Fishery until both the cui-ui and Lahontan cutthroat trout had been removed from threatened or endangered status or other sources of water are made available to conserve the species. Based on that judgment, current operation of Stampede Reservoir is guided by the Short-Term Action Plan for Lahontan Cutthroat Trout developed by the Truckee River Basin Recovery Implementation Team for the United States Fish and Wildlife Service in 2003. That document developed a set of operational plans and decision processes to guide reservoir operations based on various monthly flow targets for the Lower Truckee River to be used depending on the storage level in Stampede Reservoir and the projected spring runoff into the reservoir. The Memorandum of Agreement-Truckee River Water Management is an agreement signed by the United States Fish and Wildlife Service, Pyramid Lake Paiute Tribe, Bureau of Reclamation, and Bureau of Indian Affairs in 1999 that delineated the roles and responsibilities for development of decisions and operating plans for management of

Truckee River water for the protection and conservation of the two listed species. Under this agreement, the Pyramid Lake Paiute Tribe takes the lead role in those decisions and plans, with the other signatory agencies providing technical support. The primary responsibilities for the team include management of the waters of Stampede Reservoir and the waters of Prosser Creek Reservoir not categorized as "Tahoe Exchange Water."

Interim Storage Agreement

The Interim Storage Agreement is an agreement signed by the United States, Pyramid Lake Paiute Tribe, Sierra Pacific Power Company, and Washoe County Water Conservation District in 1994. The agreement permits the storage of privately owned water in Stampede and Boca Reservoirs by the Sierra Pacific Power Company (predecessor to the Truckee Meadows Water Authority). The agreement sets forth rules for establishment, storage, and exchange of privately owned (non-project) water within the reservoirs, and sets an upper limit of 5,000 acre-feet for the water to be carried over each year as of September 1.

Flood Control

Flood control limitations have been placed on Martis Creek Reservoir, Prosser Creek Reservoir, Stampede Reservoir, and Boca Reservoir by the United States Army Corps of Engineers. These limitations generally consist of a date in the fall by which the reservoir must be drawn down to a certain water surface elevation. They also restrict the timing when the reservoir may be brought back to its full capacity in the spring based on the status of the winter snowpack. The season for the maximum flood control limits on these dams begins on November 1 and ends on April 10 of each year, after which the reservoirs may gradually increase towards their full capacity depending on the snowpack. The winter flood control limitation for Martis Creek Reservoir is 800 acre-feet, Prosser Creek Reservoir is 9,800 acre-feet, Stampede Reservoir is 204,500 acre-feet, and Boca Reservoir is 32,900 acre-feet.

Certain flood control and dam safety limitations are followed on Donner and Independence Lakes. For Donner Lake the discharge gates on the dam are held open from November 15 through April 15. For Independence Lake the flashboards are removed from two bays in the spillway structure from November 1 through May 15, generally resulting in a maximum storage between 13,000 and 15,000 acre-feet.

Lake Tahoe is not subject to flood control limitations other than those placed by the Truckee River Agreement. Provisions in the Truckee River Agreement require that the dam be operated to the maximum extent practicable to prevent the lake from exceeding an elevation of 6,229.1 feet. A set of procedures is outlined in the Truckee River Agreement in order to facilitate this goal.

OCAP and *Pyramid Lake Paiute Tribe of Indians v. Morton*

The Operating Criteria and Procedures for the Newlands Reclamation Project, Nevada (OCAP) is a federal regulation found in the Code of Federal Regulations Volume 43, Part 418. The OCAP were originally promulgated as a federal rule in 1967 to address the water supply issues of the Truckee River below Derby Dam as they related to the operation of the Truckee Canal to divert water to the Newlands Project. The 1967 OCAP placed an upper limit on the amount of water used by the Newlands Project, and ended previous use of water for single-purpose power generation by limiting power generation from Lahontan Reservoir and the V-Canal in the Carson Division to only that generation which is incidental to consumptive uses, spills, or precautionary reservoir drawdowns. In 1973, United States District Judge Gerhard Gesell issued a judgment and order in *Pyramid Lake Paiute Tribe of Indians v. Morton* which included a new set of OCAP. The associated opinion issued by the judge provided clear direction to the Secretary of the Interior that "all water not obligated by court decree or contract with the [Truckee-Carson Irrigation District] goes to Pyramid Lake." Subsequent OCAP developed periodically from 1975 through 1997 were designed to maximize the use of the Carson River water supply to serve the Newlands Project, and minimize the use of the Truckee River as a supplementary supply. The Newlands Project is currently operated under the OCAP as revised in 1997.

The 1997 OCAP set forth a procedure for annually determining the maximum amount of water allowed to be diverted out of the Truckee Canal and Lahontan Reservoir to meet Newlands Project demand based on actual land irrigated in previous years and land anticipated to be irrigated in the upcoming year. The OCAP direct the amount of water that can be diverted from the Truckee River to Lahontan Reservoir each month through a system of storage targets on Lahontan Reservoir. Storage targets are calculated based on recent year demand. In addition, targets from January through May take into account the forecasted spring runoff from the Carson River basin. Each month, the storage targets are calculated and the amount of supplementary water necessary from the Truckee River, if any, is estimated. This amount is permitted to be diverted through the Truckee Canal to the extent that water supply from the Truckee River and the capacity of the canal allow. The canal's capacity is currently limited by Reclamation due to safety concerns, as well as by an interim temporary restraining order issed in 2008, both as a result of the January 5, 2008 breach of the canal in Fernley, Nevada.

The OCAP also set forth a system of incentives for efficient operation of the Newlands Project distribution facilities. The incentive system is based on targets for efficiency in the delivery of water from the Truckee Canal to the headgates of irrigators in the Truckee Division, and from Lahontan Reservoir to the headgates of the irrigators in the Carson Division. The Truckee-Carson Irrigation District is permitted to retain for its own use a portion of the water saved above and beyond the efficiency targets, and is required to make up for extra water used in delivery if the efficiency targets are not met.

The OCAP additionally set forth procedures for storing water in Stampede Reservoir that would have typically been diverted to the Carson Division of the Newlands Project during the winter and early spring. This water becomes known as Newlands Project Credit Storage within Stampede Reservoir. If the water is later needed to meet storage targets on Lahontan Reservoir during the subsequent irrigation season, it is released and diverted through the Truckee Canal at that time. If it is not needed during that irrigation season, the water is converted to water stored for the benefit of the Pyramid Lake fishery. To date, this portion of the OCAP has never been exercised.

The OCAP further enforce the maximum limitations on the quantity of water applied to Newlands Project lands set forth by the Orr Ditch Decree as well as the Alpine

Decree. The Alpine Decree is the final decree in *United States v. Alpine Land & Reservoir Company et al.*, issued in 1980. It adjudicated the water rights on the Carson River, and set forth limitations on the amount of water to be applied to the land. For lands in the Newlands Project, the Alpine Decree applied the same limitations as found in the Orr Ditch Decree of 3.5 acre-feet per acre to bottom lands and 4.5 acre-feet per acre on bench lands. The net consumptive use for the Newlands Project was set at 2.99 acre-feet per acre. Exact delineation of the bench and bottom lands within the Newlands Project was accomplished through a later court case. In the opinion issued by Judge Thompson in the Alpine case, the water rights in the Newlands Project were found to be appurtenant to the land, and the opinion further found that individual land owners owned the water rights, rather than the United States government.

APPENDIX B – RESERVOIR OPERATIONS ON THE TRUCKEE RIVER

Because the governing documents for the Truckee River system referenced in this appendix directly refer to exact quantities and flow rates of water in imperial units, that system of units is utilized as the primary system of units in this appendix rather than metric units.

Overview of Reservoir and River Release Operations

The general operation of the Truckee River reservoir system focuses around the concepts of the “Floriston Rates” and “Reduced Floriston Rates” outlined in the Truckee River Agreement. The Truckee River Agreement requires that the rates of flow in the Truckee River be kept at 500 cfs between March 1 and September 30 of each year. From October 1 to the end of February they are 400 cfs. They are reduced to 350 cfs between November 1 and March 31 if the level of Lake Tahoe is below 6,226.0 feet but above 6,225.25 feet. They are futher reduced to 300 cfs during that timeframe if the level of Lake Tahoe is below 6,225.25 feet.

In order to meet the Floriston Rates, natural flow into the river is first used. This natural flow is essentially all water flowing in the basin except water permitted to be stored as privately owned stored water adverse to the Floriston Rate, or water released as privately owned stored water. If the natural flow does not meet the Floriston Rate, water is released from Lake Tahoe, Boca Reservoir, and Prosser Reservoir in a priority order. Boca Reservoir is first used to meet the Floriston Rate requirements from November through March, or if the level of Lake Tahoe is above 6,225.5 feet. Lake Tahoe is used first to meet the requirement from April through October if its elevation is below 6,225.5 or any time of the year if release from Boca Reservoir does not meet the requirement. Any time that release from Lake Tahoe is not needed to meet Floriston Rates, Lake Tahoe is operated to maintain minimum flows below the dam of 50 cfs between October 1 and March 31 and 70 cfs between April 1 and September 30, while exchanging water with Prosser Reservoir as described in the Tahoe-Prosser Exchange Agreement. If water is not available for exchange, Lake Tahoe releases are terminated. In addition to releases from Lake Tahoe and Boca Reservoir, Tahoe Exchange Water from Prosser Reservoir is also used to meet the Floriston Rates requirements, generally between June and October in order to ensure that all Tahoe Exchange Water has been released from Prosser Reservoir prior to fully drawing the reservoir down to its winter flood control level.

Independence Lake is operated by the Truckee Meadows Water Authority and Donner Lake is operated by the Truckee Meadows Water Authority in association with the Truckee-Carson Irrigation District. Releases from these reservoirs are generally considered releases of privately owned stored water, which does not make up part of the water released for Floriston Rate requirements unless requested by the owner of the water. In this sense, the water “floats” on top of the water making up the Floriston Rate requirements, and is not available for diversion or use by any party other than the owner or their designee. Additionally, under the Interim Storage Agreement, the Truckee

Meadows Water Authority has the ability to release their water in exchange for water to be stored in Stampede or Boca Reservoirs.

Water is generally released from Stampede and Prosser Reservoirs for the benefit of the Pyramid Lake fishery according to the flow regime criteria set forth in the Short-Term Action Plan for Lahontan Cutthroat Trout. The water released from Stampede Reservoir is referred to as "Fish Water" and does not make up part of the water released for the Floriston Rate requirements. Similar to privately owned stored water, this water "floats" on top of the water making up the Floriston Rate requirement, and is not available for diversion from the river at Derby Dam or any other diversion facility unless permitted by the various parties to the Memorandum of Agreement-Truckee River Water Management. The stored water released from Prosser Creek Reservoir for the purposes of the Pyramid Lake fishery is any water not classified as "Tahoe Exchange Water."

Martis Creek Reservoir is generally operated to limit storage in the reservoir due to concerns over the stability of the dam. To the extent practicable, all inflow is immediately released.

All reservoirs are operated to ensure that they meet flood control requirements during winter and spring months. During floods, water is temporarily stored in Prosser Creek, Stampede, Boca, and Martis Creek Reservoirs in an attempt to limit flow in Reno to no more than 6,000 cfs. After the flood event, the water is then released to get the reservoirs back to the flood control limits.

Derby Dam and the Truckee Canal are operated subject to the OCAP. Derby Dam is permitted to divert water to meet the needs of the Truckee Division of the Newlands Project using all available water arriving at Derby Dam that is not privately owned stored water, subject to prior appropriations, or designated as water for the benefit of the Pyramid Lake fishery. If needed to meet storage targets at Lahontan Reservoir, additional water may be diverted and conveyed to Lahontan Reservoir subject to those same limitations.

<u>Overview of Reservoir Storage Operations</u>

Subject to flood control limitations, Donner Lake may store water adverse to Floriston Rate requirements. Independence Lake may also store up to 3,000 acre-feet of water adverse to Floriston Rates. Once Floriston Rate requirements are met, Lake Tahoe has the first priority to store all additional water flowing into it subject to its maximum lake surface elevation requirements. If Floriston Rates requirements continue to be met, up to 25,000 acre-feet may be stored in Boca Reservoir.

Once Boca Reservoir has stored 25,000 acre-feet of water, all additional inflow must be passed to the Truckee Canal to meet the greater of its diversion demand under the OCAP or its capacity. If those demands are met, Boca Reservoir may be filled by an additional 15,850 acre-feet to its capacity of 40,850 acre-feet. If all requirements and demands continue to be met, Independence Reservoir may fill up to an additional 14,500 acre-feet, subject to water availability. Finally, if all requirements, demands, and storage priorities have been met, Stampede Reservoir and Prosser Creek Reservoir may store all additional available water. Prosser Creek Reservoir may also begin to store water earlier in the priority system under the Tahoe Prosser Exchange if conditions allow.

APPENDIX C – BASIC WQSA RELEASE POLICIES

Table C.1 – Basic release policies from agent learning; using 30-year historic dataset, SARSA, γ = 0.0, OFU exploration, after 58 model runs

WQSA Storage MCM (acre-feet)	Temperature °C										
	<22	22-22.5	22.5-23	23-23.5	23.5-24	24-24.5	24.5-25	25-25.5	25.5-26	26-27	>27
0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)
0.09 (70)	0 (0)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)
0.17 (140)	0 (0)	1 (35)	2 (70)	2 (70)	2 (70)	2 (70)	2 (70)	2 (70)	2 (70)	2 (70)	2 (70)
0.26 (210)	0 (0)	1 (35)	2 (70)	3 (105)	3 (105)	*	*	3 (105)	3 (105)	3 (105)	3 (105)
0.35 (280)	0 (0)	1 (35)	2 (70)	3 (105)	*	*	*	*	*	*	*
0.43 (350)	0 (0)	1 (35)	2 (70)	*	*	*	*	*	*	*	*
0.52 (420)	0 (0)	*	2 (70)	*	*	*	*	*	*	*	*
0.6 (490)	*	*	*	*	*	*	*	*	*	*	*
0.69 (560)	0 (0)	1 (35)	*	*	*	*	5.9 (210)	6.9 (245)	*	7.9 (280)	*
1.38 (1,120)	0 (0)	1 (35)	2 (70)	*	4 (140)	*	5.9 (210)	7.9 (280)	*	9.9 (350)	11.9 (420)
2.76 (2,240)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5.9 (210)	5.9 (210)	6.9 (245)	7.9 (280)	9.9 (350)	11.9 (420)
5.53 (4,480)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	6.9 (245)	7.9 (280)	8.9 (315)	*
8.29 (6,720)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	*	*	*	*
11.05 (8,960)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	*	*	*	*
13.81 (11,200)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5.9 (210)	*	*	*	9.9 (350)	*
16.58 (13,440)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	6.9 (245)	7.9 (280)	9.9 (350)	11.9 (420)

Table values are WQSA release in cms (cfs)

* values indicate not enough experience with a state to form a policy

Table C.2 – Basic release policies from idealized agent learning with alternate water supplies; using 30-year historic dataset, SARSA, γ = 0.0, OFU exploration, after 64 model runs

	Temperature °C										
WQSA Storage MCM (acre-feet)	<22	22-22.5	22.5-23	23-23.5	23.5-24	24-24.5	24.5-25	25-25.5	25.5-26	26-27	>27
0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)
0.09 (70)	0 (0)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)
0.17 (140)	0 (0)	1 (35)	2 (70)	2 (70)	2 (70)	2 (70)	2 (70)	2 (70)	2 (70)	2 (70)	2 (70)
0.26 (210)	0 (0)	1 (35)	2 (70)	3 (105)	3 (105)	3 (105)	3 (105)	3 (105)	3 (105)	3 (105)	3 (105)
0.35 (280)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	4 (140)	4 (140)	4 (140)	4 (140)	4 (140)	4 (140)
0.43 (350)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5 (175)	5 (175)	5 (175)	5 (175)	5 (175)
0.52 (420)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	5.9 (210)	5.9 (210)	5.9 (210)	5.9 (210)
0.6 (490)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	6.9 (245)	6.9 (245)	6.9 (245)	6.9 (245)
0.69 (560)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	6.9 (245)	7.9 (280)	7.9 (280)	7.9 (280)
1.38 (1,120)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	6.9 (245)	7.9 (280)	9.9 (350)	11.9 (420)
2.76 (2,240)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	7.9 (280)	7.9 (280)	9.9 (350)	11.9 (420)
5.53 (4,480)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	6.9 (245)	7.9 (280)	8.9 (315)	9.9 (350)	11.9 (420)
8.29 (6,720)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	7.9 (280)	7.9 (280)	9.9 (350)	11.9 (420)
11.05 (8,960)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	7.9 (280)	7.9 (280)	9.9 (350)	11.9 (420)
13.81 (11,200)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	6.9 (245)	8.9 (315)	9.9 (350)	11.9 (420)
16.58 (13,440)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	6.9 (245)	7.9 (280)	9.9 (350)	11.9 (420)
Table values are WQSA release in cms (cfs)											
* values indicate not enough experience with a state to form a policy											

Table C.3 – Analytically derived table of basic release policies

WQSA Storage MCM (acre-feet)	Temperature °C										
	<22	22-22.5	22.5-23	23-23.5	23.5-24	24-24.5	24.5-25	25-25.5	25.5-26	26-27	>27
0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)
0.09 (70)	0 (0)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)	1 (35)
0.17 (140)	0 (0)	1 (35)	2 (70)	2 (70)	2 (70)	2 (70)	2 (70)	2 (70)	2 (70)	2 (70)	2 (70)
0.26 (210)	0 (0)	1 (35)	2 (70)	3 (105)	3 (105)	3 (105)	3 (105)	3 (105)	3 (105)	3 (105)	3 (105)
0.35 (280)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	4 (140)	4 (140)	4 (140)	4 (140)	4 (140)	4 (140)
0.43 (350)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5 (175)	5 (175)	5 (175)	5 (175)	5 (175)
0.52 (420)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	5.9 (210)	5.9 (210)	5.9 (210)	5.9 (210)
0.6 (490)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	6.9 (245)	6.9 (245)	6.9 (245)	6.9 (245)
0.69 (560)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	6.9 (245)	7.9 (280)	7.9 (280)	7.9 (280)
1.38 (1,120)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	6.9 (245)	7.9 (280)	9.9 (350)	11.9 (420)
2.76 (2,240)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	6.9 (245)	7.9 (280)	9.9 (350)	11.9 (420)
5.53 (4,480)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	6.9 (245)	7.9 (280)	9.9 (350)	11.9 (420)
8.29 (6,720)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	6.9 (245)	7.9 (280)	9.9 (350)	11.9 (420)
11.05 (8,960)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	6.9 (245)	7.9 (280)	9.9 (350)	11.9 (420)
13.81 (11,200)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	6.9 (245)	7.9 (280)	9.9 (350)	11.9 (420)
16.58 (13,440)	0 (0)	1 (35)	2 (70)	3 (105)	4 (140)	5 (175)	5.9 (210)	6.9 (245)	7.9 (280)	9.9 (350)	11.9 (420)
Table values are WQSA release in cms (cfs)											

APPENDIX D – LONG-TERM AGENT PERFORMANCE

This appendix illustrates the performance of the long-term reinforcement learning agent using the 30-year historic hydrology. The agent is provided the full flexible exchange of WQSA water as described in Section 5.1.1, and up to 98.7 MCM (80,000 acre-feet) of storage is protected from displacement or spill. Various agent configurations are shown in Figures D.1 through D.9.

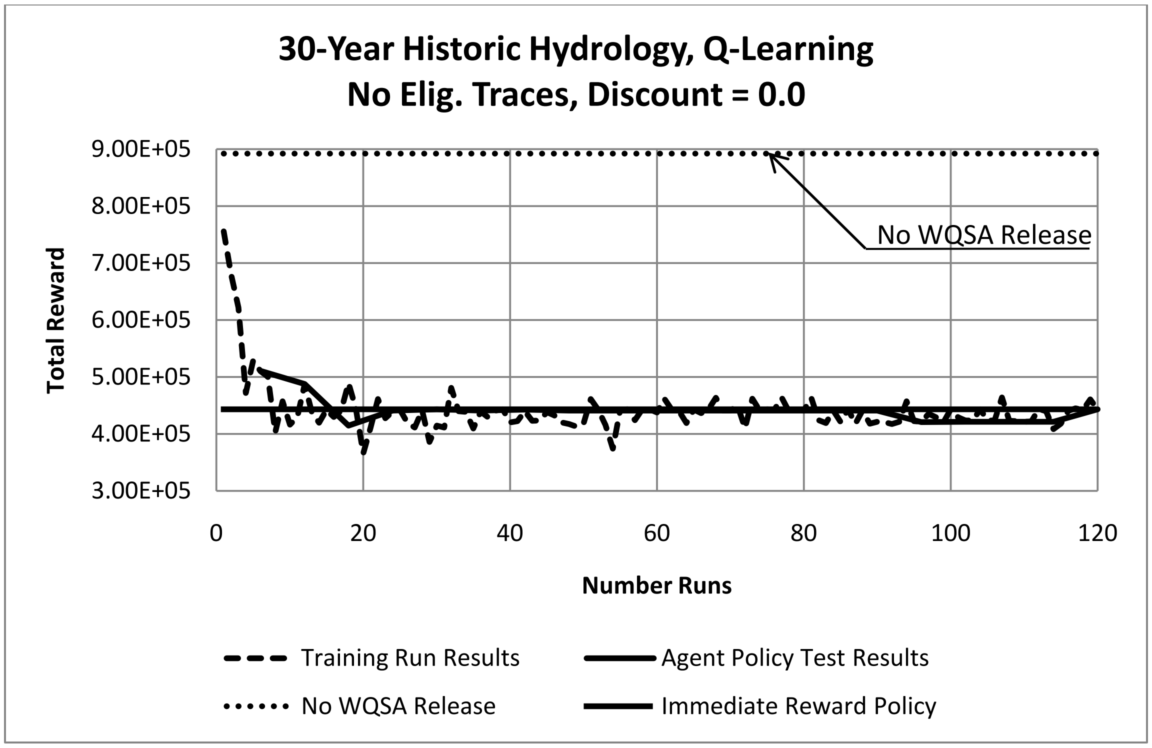


Figure D.1 – Agent performance on 30-year historic hydrology with 98.7 MCM (80,000 acre-feet) firm storage, using Q-Learning with no eligibility traces, γ = 0.0

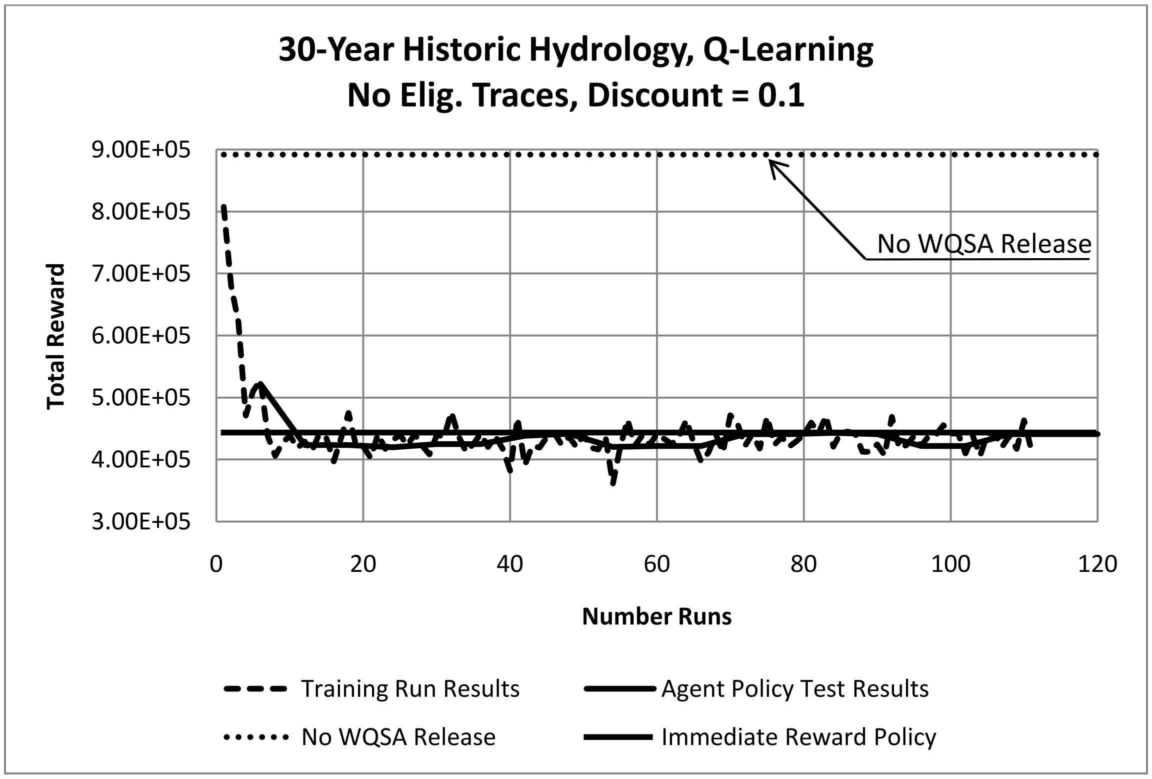


Figure D.2 – Agent performance on 30-year historic hydrology with 97.8 MCM (80,000 acre-feet) firm storage, using Q-Learning with no eligibility traces, γ = 0.1

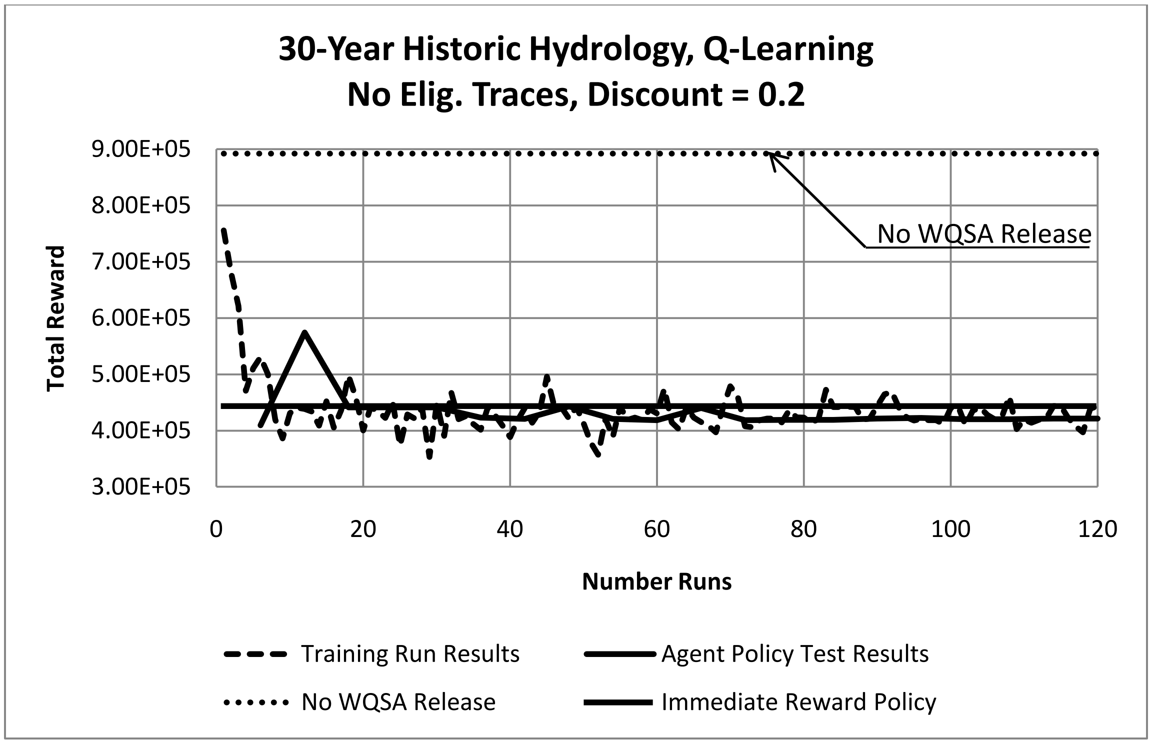


Figure D.3 – Agent performance on 30-year historic hydrology with 97.8 MCM (80,000 acre-feet) firm storage, using Q-Learning with no eligibility traces, γ = 0.2

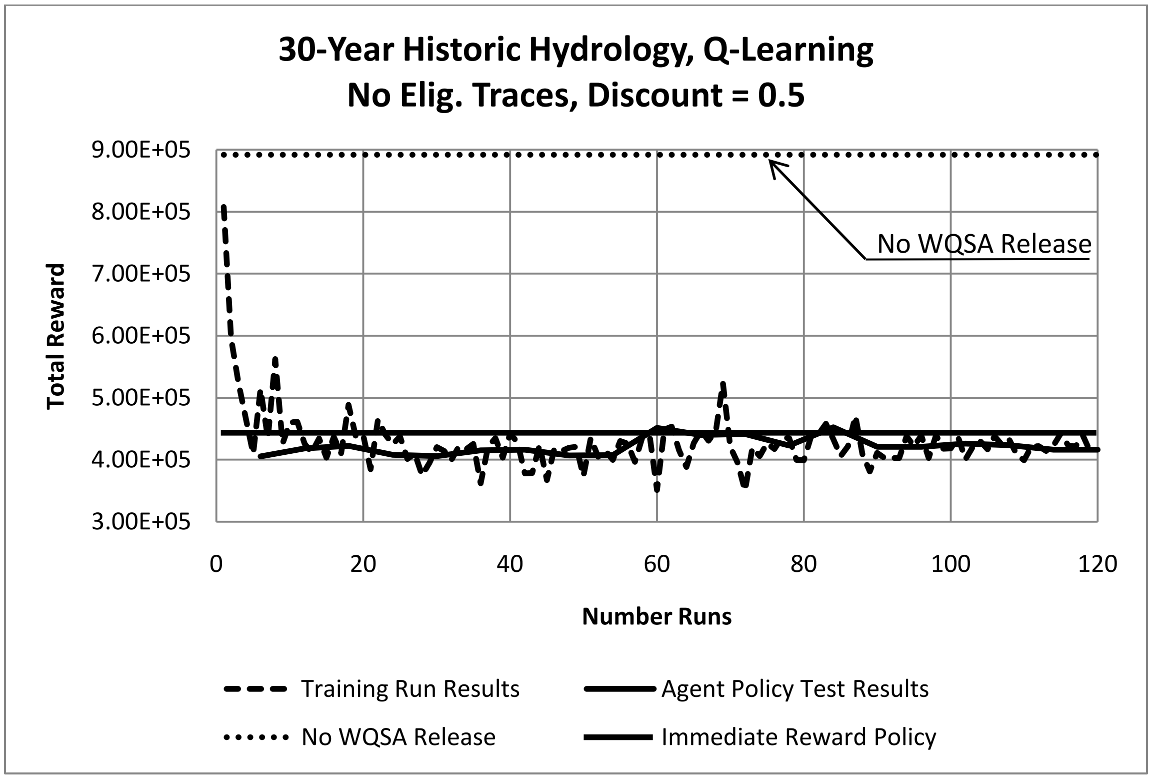


Figure D.4 – Agent performance on 30-year historic hydrology with 97.8 MCM (80,000 acre-feet) firm storage, using Q-Learning with no eligibility traces, γ = 0.5

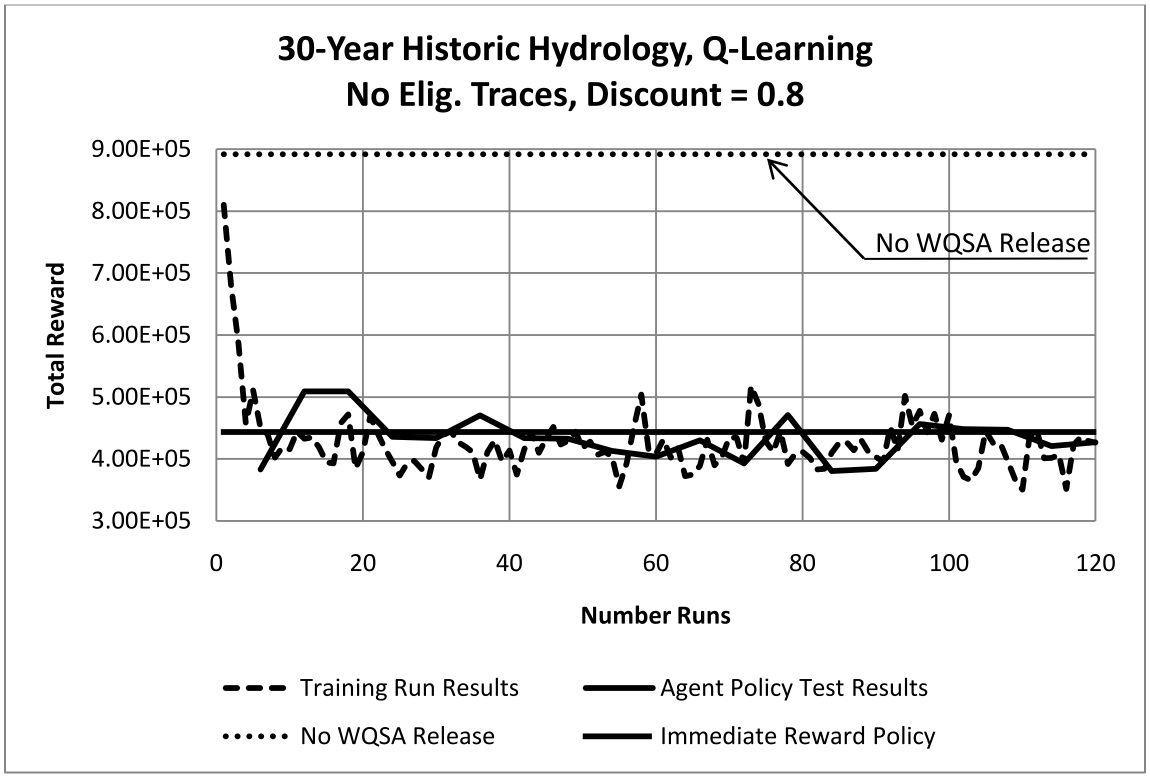


Figure D.5 – Agent performance on 30-year historic hydrology with 97.8 MCM (80,000 acre-feet) firm storage, using Q-Learning with no eligibility traces, γ = 0.8

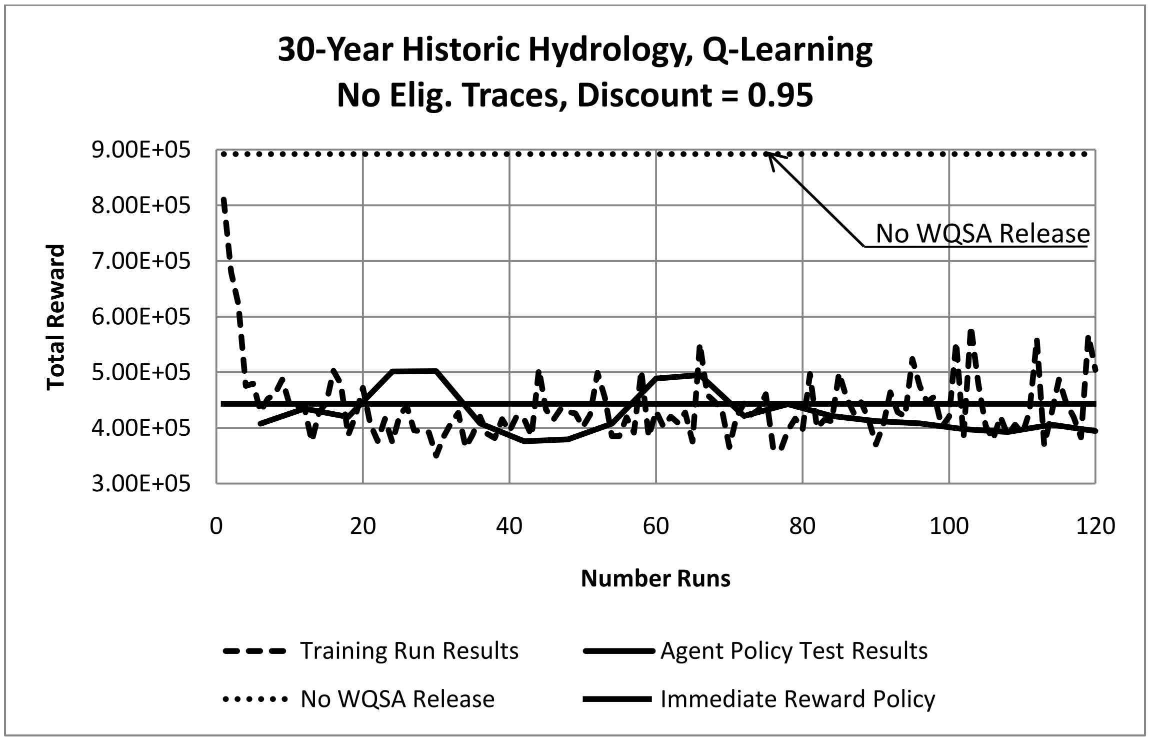


Figure D.6 – Agent performance on 30-year historic hydrology with 97.8 MCM (80,000 acre-feet) firm storage, using Q-Learning with no eligibility traces, γ = 0.95

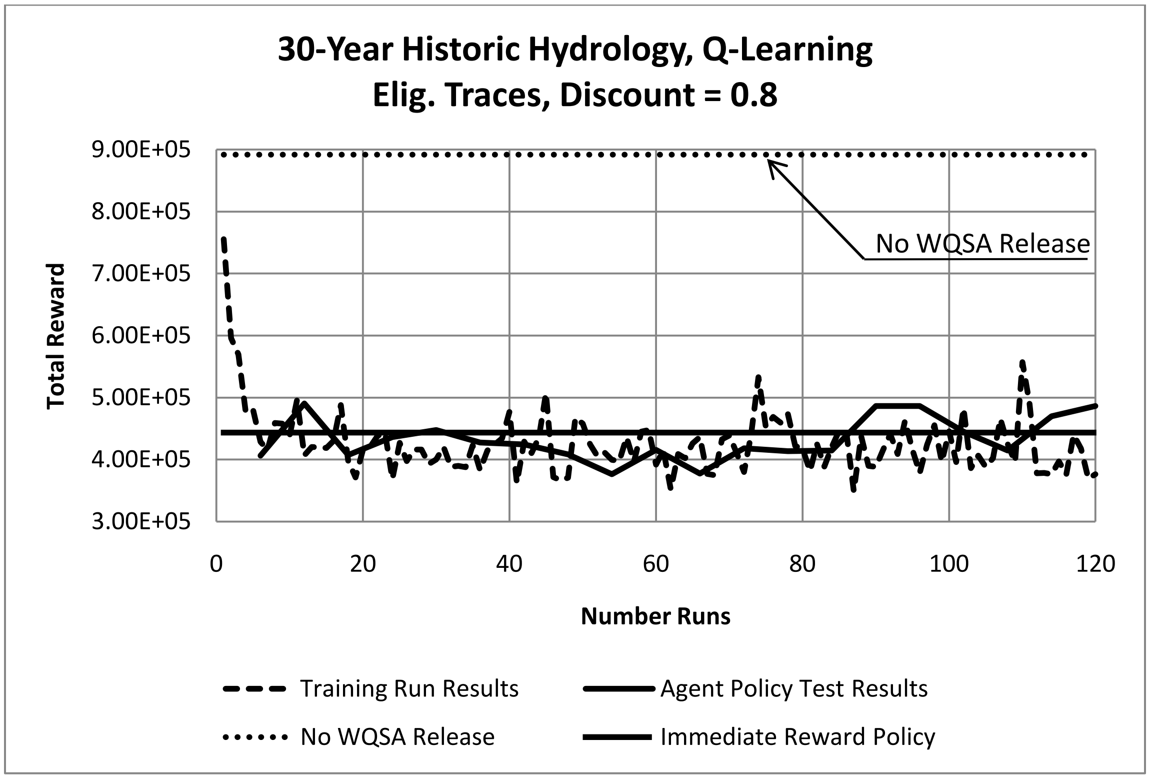


Figure D.7 – Agent performance on 30-year historic hydrology with 97.8 MCM (80,000 acre-feet) firm storage, using Q-Learning with eligibility traces, λ = 0.5, γ = 0.8

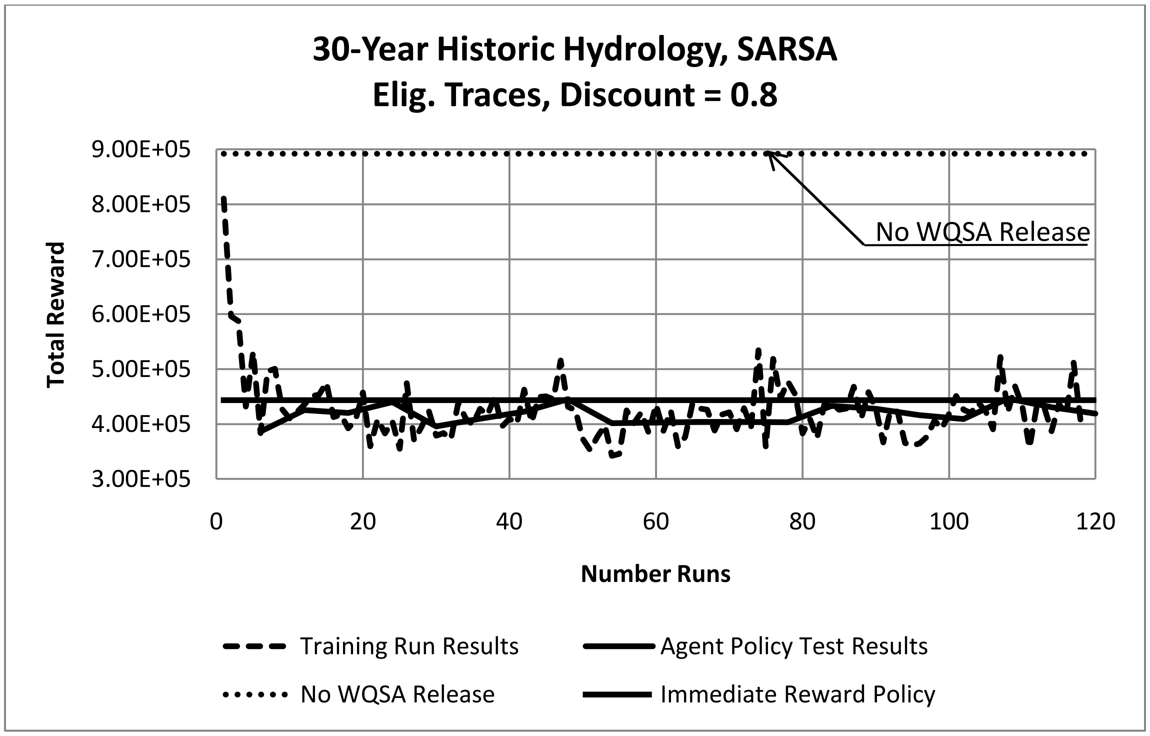


Figure D.8 – Agent performance on 30-year historic hydrology with 97.8 MCM (80,000 acre-feet) firm storage, using SARSA with eligibility traces, λ = 0.5, γ = 0.8

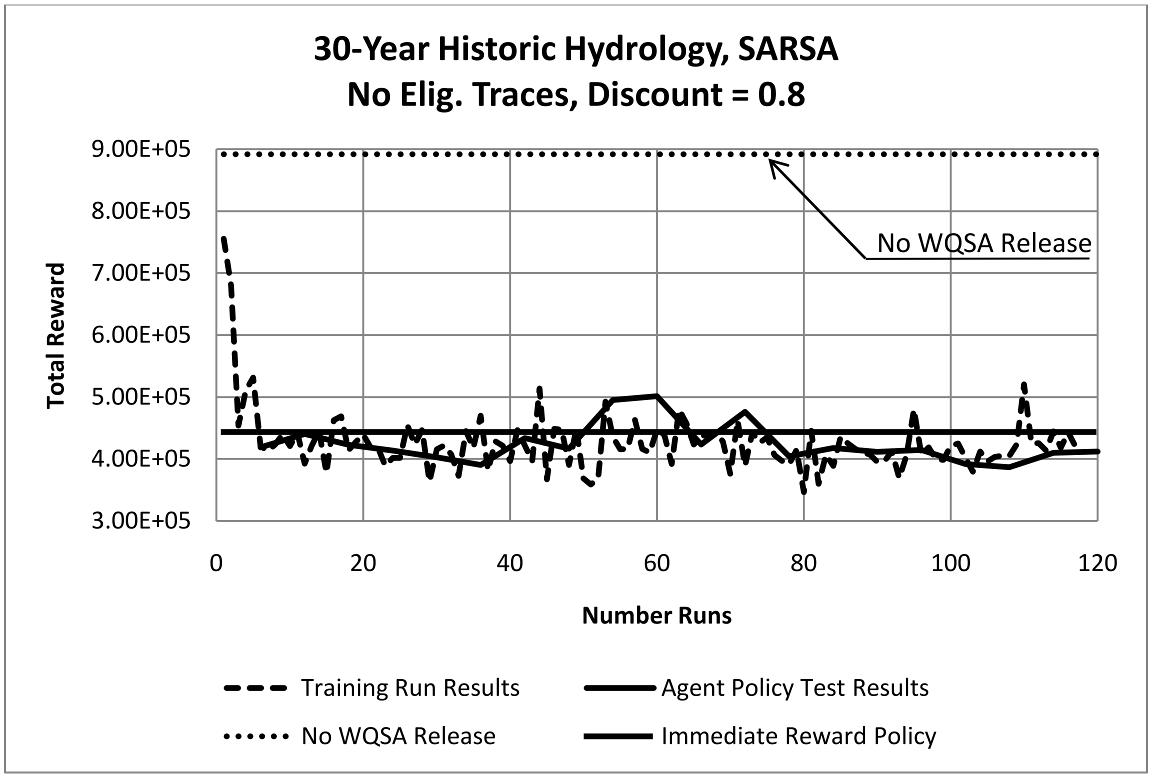


Figure D.9 – Agent performance on 30-year historic hydrology with 97.8 MCM (80,000 acre-feet) firm storage, using SARSA with no eligibility traces, $\gamma = 0.8$

APPENDIX E – IMPLEMENTATION FINDINGS

During the course of the development of the water quality reinforcement learning agent, as well as the testing and evaluation of the agent on the Truckee River case study, a number of practical implementation issues were identified and addressed. The following sections discuss these issues and how they were addressed, and are expected to aid researchers as well as practitioners encountering similar issues in future studies and applications.

E.1 – Elimination of Unnecessary Evaluation/Seasonality Issues

During early testing on the Truckee River case study it was identified that large portions of each simulated year did not require agent action due to the relatively low predicted temperatures in the river. It was found that approximately 93 percent of the timesteps in the 30-year historical hydrologic model runs did not require agent action due to temperatures below the preferred maximum. This generally occurred during the winter and early spring months, and also in years with high runoff. Based on the agent's selected exploration or action selection mechanism, the agent would test actions during those timeframes that were detrimental to the goal of storing additional water and testing actions at times when releases were necessary. The use of softmax and OFU action selection methodologies, as well as gradually reducing ε values in the ε-greedy exploration method helped mitigate the issue by causing the agent to greatly reduce its release exploration during these periods after limited exploration had been completed. However, exploratory actions were still taken, thus reducing the agent's learning rate with respect to more meaningful states encountered by the agent. In addition, using state-specific reductions to the step size parameter and ε or τ terms of the exploratory methods mitigated against the detrimental impacts of these less meaningful actions, but they were still a hindrance to the agent for the reasons stated.

It was found that the most effective method to deal with this issue was to completely eliminate agent interaction during the times of the simulated year when agent action was highly likely to be unnecessary. Rules were introduced to the simulation model that produced a zero action selection from December 1 through May 31 of each simulated year, and did not even connect to the agent interface program. This had the added benefit of greatly accelerating run times for half of the modeled time period. Though this was a somewhat obvious step to mitigate seasonality issues for the agent, its use was valuable in improving agent performance as well as computational efficiency.

E.2 – Neural Network Issues

One of the more significant practical issues encountered in the use of neural networks for function approximation was the issue of input and output scaling. Due to the fact that the activation functions used on units in the neural network produce outputs in the range between zero and one, scaling of training and calculated outputs is necessary.

In addition, scaling of inputs can help network training by making the possible range of all inputs comparable, rather than having different inputs whose magnitude of change from one input value to the next vary widely.

In the context of the reinforcement learning system discussed in Section 3.2.5, the issue of scaling becomes difficult due to the fact that the range of possible values for each input and output are not known to the agent *a priori*. Thus, as the agent begins to learn the possible range for each variable with experience, any scaling applied to the input and output values will change with time, creating a highly non-stationary problem for the neural network to attempt to solve. When coupled with a reinforcement agent that is attempting to actively use the networks for action-value updates and policy decisions, the situation becomes less than ideal, and during testing was observed to be unstable. An example of this problem would be when an update is applied to the action-value function computed by the neural network, where the update results in an action-value that exists at or beyond the extreme maximum or minimum of the range of action-value training data. The action-value update results in a change to the scaling function used by the agent, thus changing the value that will be computed by the network for the agent's next update calculation to one that may not be an accurate estimate of the function until further training occurs. Because of the speed at which the agent operates with respect to the training of the function, instability can ensue.

Extensive testing was not conducted to identify whether a dynamic scaling mechanism would eventually stabilize, and how long that might take. Rather, the scaling used by the neural network was accomplished for the system through the user specification of extreme values for each input variable, as well as the action-values and policy values themselves. For the input and policy variables, the determination of those extremes should be fairly straightforward from basic prior knowledge of the system and problem. For the action-value range, this requires basic knowledge of the reward functions, and the use of Equation 3.28. It is much more difficult to determine when eligibility traces are used, and assumptions may be necessary in that case.

Another difficult practical issue that arose in testing was the performance speed of the neural network as trained by genetic algorithms. As the number of training data points grows with agent experience, the number of network calculations required for each training cycle grows. When using genetic algorithms, a network calculation must be completed for each member of the solution population per training data point. It was generally found that the problem could be reasonably mitigated through the reduction of the number of members of the solution population.

E.3 – Reducing Step Size Parameters

One significant issue identified in testing on the Truckee River case study was the ability for the agent to act in a highly variable Truckee River environment.   Early testing showed strong detrimental effects on the stability of the agent's action-value function stemming from some of the immediate step-like functions of normal Truckee River reservoir release operations for other purposes than water quality. An example of this type of problem is in the event of a very dry year, the flow in the river will continue at a typical rate until the very day that the general water rights storage supply has been depleted. This is locally known to water managers as the loss of the "Floriston Rate."

On that day, flow in the river will drop remarkably, causing the water temperature to spike, also causing a very strong negative influence on the action-value update calculation for the state and action encountered on the previous timestep. Depending on the reinforcement learning step size parameter and discount rate, it was found to take quite some time for the action-value of that state and action to re-stabilize. This and other issues with variability on the Truckee River case study illustrated the importance of a reducing step size parameter. Through the use of a gradually reducing step size parameter, the problem was generally mitigated, since these events were the exception rather than the norm, and the overall action-value for the particular states affected would generally stabilize on the more normative condition and eventually not be greatly influenced by these more rare conditions.

E.4 – Discussion on Partitioning of the State-Action Space

One of the primary issues faced by the agent in the Truckee River case study was the difficulty in having sufficient opportunity to explore the entire action domain. This is caused by the fact that the storage of WQSA water only occurs in years that the hydrologic situation causes diversions to occur out of the Truckee River and through the Truckee Canal. Additionally, the total amount of water able to be stored is limited by the amount of water rights purchased. Also, the time of year that the WQSA water is able to be stored generally coincides with the time of year that the water is most likely needed for use. Even in the event that additional water supplies are secured, the water is subject to displacement from the reservoirs through spill as previously discussed. Generally, stability and convergence of the action-value function and agent's policy is only guaranteed if the entire state and action space can be visited a sufficient number of times. In the case of the Truckee River case study, this presented a challenge and significant issue in the development of a good representation of the action-value function and policy.

The partitioning of the state-action space as discussed in Sections 3.2.5.1 and 4.6.1 was found to be an important mechanism to ensure higher levels of experience and increased opportunity for exploration across the state space. Though the challenge remained present, it provided for the shared learning of all experiences within the broader categories developed for each state variable and the action variable, allowing for increased experience in those categories, and serving as a form of generalization.

In addition to helping overcome the experience and exploration issues described, the system of partitioning the state-action space had the added benefits of simplifying the various processes associated with the tabular implementation. The use of the feature extraction concepts discussed in Section 3.2.5.1 allowed partition categories to have a system of regular intervals. This provided for simplified generation of initial function values, simplified calculation of the "$\max_a Q(s_{t+1}, a_t)$" term in the action-value update Equation 3.14 under Q-Learning, and more efficient mechanisms for conducting the policy improvement procedure.

The benefits associated with partitioning also extended to the neural network function approximation implementation. The use of partitioning and feature extraction simplified the process of developing training data for the neural networks at regular intervals, theoretically improving the ability to approximate the function throughout the state and state-action space rather than at locations where higher densities of training data would be generated. Similar to the tabular implementation, partitioning simplified the process of finding the "$\max_a Q(s_{t+1}, a_t)$" term in the action-value update Equation 3.14 under Q-Learning. This also facilitated the policy improvement procedure by providing a limited number of locations where the action-value function was required to be evaluated in order to determine the agent's policy.

Finally, the use of partitioning and feature extraction allowed the development of policies that are more likely to be used by water managers in a real-world setting. The final policies were generated in the form of simple look-up tables, which is more similar to other types of policies generally used by water managers, particularly in the Truckee River basin.

E.5 – Tuning of Rewards and Partition Bins

Reinforcement learning texts note the importance of the development of good reward functions. In testing on the Truckee River case study, it was found that one of the more important factors in the agent's learning and development of stable action-value and policy functions was consistency of rewards. Though this should seem obvious, it does not always practically occur, particularly when partitioning or feature extraction is used. Depending on the size and/or design of the partitions or categories used for state representation, the agent can take actions that produce different results and therefore different rewards from what appear to be the exact same state to the agent. Depending on the design of the reward function, this can create difficulties for an agent in deciding what action has the best reward signal, causing the agent to oscillate between policies. It was found through testing that the problem can largely be mitigated by ensuring that the reward function takes the state and action partitioning or feature extraction into account, and is likely to generate consistent rewards. Likewise, it is important to ensure that the state and action partitioning is designed in a way that produces similar results for any action taken in any given state.

As previously discussed, for the final state and action discretization on the Truckee River case study, the state and action partitioning was generally designed around the actions and storage volume required to change downstream river temperature by 0.5° C. This discretization provided a means to meet the various temperature targets discussed in Section 4.5.2.3, while not allowing the agent to over-release by an excessive amount, thus conserving as much water as could be reasonably expected. Even with careful tuning of the discretization and reward function, the inconsistencies described above are the reason that the basic policies shown in Tables C.1 and C.2 are slightly different than the analytically derived policy shown in Table C.3.

E.6 – Issues with Distributed Computing and Parallel Processing

The system of multiple environment models and agent interface programs discussed in Section 3.2.5 provides for more rapid agent learning through the shared experience of the agent's actions in each of the environment models running concurrently. The system also presented a variety of practical challenges. The most notable challenge was with the use of common files, such as the various data and tracking files, as well as log and other system files. With many different programs attempting to read and write to the files at the same time, the system was created in a way that prevented the programs from attempting to update the same file at the same time, or read a file that was actively in the process of being updated. This practical implementation mechanism ensured system integrity.

Other practical issues included the challenges associated with the processing of action-value updates, particularly when those updates are generated so quickly that many were needed to be processed in a single “batch”. These issues were magnified if eligibility traces were introduced. Those challenges led to the procedures described in Section 3.2.5.1, which successfully mitigated the issues.