STOCHASTIC DYNAMIC PROGRAMMING OPTIMIZATION MODEL FOR OPERATIONS PLANNING OF A MULTIRESERVOIR HYDROELECTRIC SYSTEM

by

Amr Ayad

M.Sc., Alexandria University, 2006

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

(Civil Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

January 2018

Abstract

This thesis presents a Stochastic Dynamic Programming (SDP) modeling algorithm to model six hydropower plants in British Columbia (BC), Canada. The main output of the algorithm is the water value function for the two biggest reservoirs in BC, Williston and Kinbasket reservoirs. The AMPL programming language was used to implement the algorithm. Extensive testing has shown that the program is able to solve the problem producing acceptable water value and marginal value functions up to a problem size of ~ 164 million states per time step using the computing resources available on one of the BC Hydro's servers.

The objective of the work presented here was to assess the sensitivity of solution efficiency and precision for several storage state and decision space discretizations. The impact of introducing a storage state-space corridor, as an alternative of the traditional fixed storage state-space, was investigated. In addition, the sensitivity of the modeling results to different spill penalty values was analyzed. It was found that finer state-space increments give more precise results but the granularity was limited to the computing resources available. Introducing the storage state-space corridor provided several advantages; nevertheless, care should be taken in the design of such corridors so that the solution efficiency and accuracy are not jeopardized. Also, recommendations on the use of suitable spill penalty value are provided.

Flexibility is one important feature of the modeling algorithm. This flexibility is a result of optimizing the algorithm and the organization of the code, which provided control over the increment of the state-spaces and the storage corridor, the ability to run the problem for one storage reservoir while fixing the state of the other storage reservoir and the ability of the user to

run the model either directly on a personal computer/server using the command prompt or by using a scheduling program to optimize the use and sharing of computing assets.

Further enhancements of the algorithm will enable the model developed in this thesis to handle much larger problems but will likely still suffer from the limitations due to the inherent curse of dimensionality in modeling using the SDP algorithm.

Lay Summary

The researcher has developed a computer program that uses advanced mathematical modeling technique, called Stochastic Dynamic Programming, to produce price signals representing the value of water stored in British Columbia's biggest reservoirs such as Williston Reservoir in the Peace region in British Columbia (BC) and Kinbasket Reservoir in the Columbia region in BC. These price signals are intended to inform the operators of the generating stations downstream of these reservoirs of the optimal way to dispatch the generation, within a certain time window, through comparing the value of the energy produced to the value of energy in the wholesale electricity markets connected to BC.

To ensure that this computer program is working properly, the researcher has tested the program using several case studies with different input parameters. The results of the tests have shown that the program gives acceptable results within certain limits.

Preface

The work presented in this research is part of BC Hydro's *Water Value Project* in collaboration with a research team from the University of British Columbia and with financial support from BC Hydro and the Natural Sciences and Engineering Research Council (NSERC) in Canada for the research supervisor, Dr. Ziad Shawwash of the University of British Columbia.

This thesis is based on the development and testing of a model applied to the BC Hydro system. The author, Amr Ayad, is the lead investigator responsible for model design, development, and analysis of the results and composition of the manuscript in Chapter 1 through Chapter 6.

A version of Chapter 3 is a fully detailed documentation of the modeling developed as part of this research. This documentation is used internally at BC Hydro (Ayad, et al., 2012). Part of this documentation could be also found in (Abdalla, et al., 2013).

A version of Chapter 5 was published in the conference proceedings of HydroVision International 2013 (Ayad, et al., 2013).

Dr. Shawwash provided guidance throughout the building of the model, gathering the data and analyzing the results. Dr. Shawwash also has contributed to editing and review of all the chapters included in this manuscript.

Many others at BC Hydro and University of British Columbia have provided guidance, and the author hereby acknowledges their support and contributions.

Much of the computations and analysis was performed using the hardware and software resources at BC Hydro.

Table of Contents

List of Tables

List of Figures

List of Symbols

T_S: Starting time step in months,

T_E: Ending time step in months,

t: Time step, where t ∈ $T_S, \dots, T_E$ in months,

j: Plant number; since there are two plants in the model, j of GM Shrum =1 while j for Mica=2,

$Q_P\_Max_{j,t}$: Maximum physical plant j release limit for each time step t in m^3/sec,

$Q_P\_Min_{j,t}$: Minimum physical plant j release limit for each time step t in m^3/sec,

$Q_{turb\ j,t}$: Turbine release in m^3/sec,

$Q_{turb}\_Max_{j,t}$: Maximum physical limit on turbine release in m^3/sec,

$Q_{min}\_Max_{j,t}$: Maximum physical limit on turbine release in m^3/sec,

$Q_{spill\ j,t}$: Spill outflow in m^3/sec,

$a_{j,t}$: Action (Decision) which represents the releases for each time step t from each plant j in m^3/sec,

$W\_upper_{j,t}$: Absolute maximum storage limit for plant j for time step t in m^3/sec-day which is deduced from the dynamic storage trajectory,

$W\_lower_{j,t}$: Absolute minimum storage limit for plant j for time step t in m^3/sec-day which is deduced from the dynamic storage trajectory,

$S_{j,t}$: Discretized starting storage value for the plant j in the current time step t in m^3/sec-day,

$W_{j,t+1,S_{j,t}}$: Discretized terminal storage value for the plant j in the next time step t+1 for the starting storage $S_{j,t}$ in m^3/sec-day,

$i_{j,t}$: Natural inflow for each plant in each time step t in m^3/sec,

L_t: Domestic Load for each time step t in MWh,

$HK_{j,t,S_{j,t},W_{j,t+1},S_{j,t}}$: Coefficient governing the generation storage relationship,

$G_{j,t,S_{j,t},i_{j,t},W_{j,t+1},S_{j,t},a_{j,t}}$: Energy generation from plant j in time t for the pair of storage states $S_{j,t}$ and $W_{j,t+1},S_{j,t}$ and inflow value $i_{j,t}$ and decision $a_{j,t}$ in MWh,

$G\_min_{j,t}$: $Minimum\ physical\ generation\ limit\ in\ MW,$

$G\_max_{j,t}$: $Minimum\ physical\ generation\ limit\ in\ MW,$

G_PCN_t: Energy generation from Peace Canyon plant in MWh,

G_REV_t: Energy generation from Revelstoke plant in MWh,

G_ARD_t: Energy generation from Arrow Lakes plant in MWh,

$G\_IPP\_Therm_t$: Energy generation from the independent power producers , thermal plants and other sources in MWh,

$Contr\_Exp_t$: Forward energy sales for each time step t in MWh,

$Contr\_Imp_t$: Forward energy purchases for each time step t in MWh,

$Spot\_Buy_{t,S_{1,t},S_{2,t},i_{1,t},i_{2,t},W_{1,t+1},S_{1,t},W_{2,t+1},S_{2,t},a_{1,t},a_{2,t}}$: Amount of energy that ought to be bought from the spot market in MWh,

$Spot\_Sell_{t,S_{1,t},S_{2,t},i_{1,t},i_{2,t},W_{1,t+1},S_{1,t},W_{2,t+1},S_{2,t},a_{1,t},a_{2,t}}$: Amount of energy that could be sold to the spot market in MWh,

$Trans\_Exp\_Limit_t$: Transmission lines limits on exporting in MW,

$Trans\_Imp\_Limit_t$: Transmission lines limits on importing in MW,

Inr: Interest rate,

γ: Discount rate,

$Exp\_Price_{t,i_{1,t},i_{2,t}}$: Export price according to the incoming inflows to the plants in $/MWh,

$Imp\_Price_{t,i_{1,t},i_{2,t}}$: Import price according to the incoming inflows to the plants in $/MWh,

$Cont\_Price_{t,i_{1,t},i_{2,t}}$: Contract price according to the incoming inflows to the plants in \$/MWh,

$EX\_Imp\_Cost_{t,S_{1,t},S_{2,t},a_{1,t},a_{2,t}}$: The expected cost for importing from the spot market in million \$,

EX_Exp_Rev:$_{t,S_{1,t},S_{2,t},a_{1,t},a_{2,t}}$: The expected revenue from exporting to the spot market in million \$,

$Cont\_Rev\_Rev_{t,S_{1,t},S_{2,t},a_{1,t},a_{2,t}}$: The revenue/ cost of the forward long-term contracts in million \$,

$Policy\_Income_{t,S_{1,t},S_{2,t},a_{1,t},a_{2,t}}$: The expected revenue/ cost of operation in million \$,

$T\_P_{t,S_{1,t},S_{2,t},i_{1,t},i_{2,t},W_{1,t+1,S_{1,t}},W_{2,t+1,S_{2,t}},a_{1,t},a_{2,t}}$: The state transition probability,

$PV_{t,S_{1,t},S_{2,t}}$: Value of water in storage in million \$,

$MVW_{t,S_{1,t},S_{2,t}}$: Marginal value of energy in \$/ MWh,

$FB\_i_{j,t,S_{j,t}}$: The forebay corresponding to certain starting storage state in m,

$FB\_f_{j,t,S_{j,t},i_{j,t},W_{j,t+1,S_{j,t}}}$: The forebay corresponding to certain terminal storage state in m,

List of Abbreviations

AMPL	A Mathematical Programming Language
ARW	Arrow Lakes Reservoir
BC	British Columbia
BC Hydro	The British Columbia Hydro and Power Authority
CCP	Chance-Constrained Programming
cms-d/cms-day	Cubic meters per second days
CPU	Central Processing Unit
CRO/ Flocal	Commercial Resource Optimization/ Flow calculation Software
CRT	The Columbia River Treaty
CSUDP	Colorado State University Dynamic Programming Software
DDP	Dual Dynamic Programming
DP	Dynamic Programming
DPSA	Dynamic Programing with Successive Approximation
HK	Energy Conversion Factor
GWh	Giga Watt Hour
IDP	Incremental Dynamic Programming
IPP	Independent Power Producers
KBT	Kinbasket Reservoir
LDR	Linear Decision Rules
MCM	Marginal Cost Model
LP	Linear Programming

MW	Mega Watt
PCN	Dinosaur Reservoir
REV	Revelstoke Reservoir
RLROM	Reinforcement Learning Reservoir Optimization Model
ROTR	Run-of-the-river plant
SDPOM6R	Stochastic Dynamic Programming Optimization Model for 6 Reservoirs
SDP	Stochastic Dynamic Programming
SDDP	Stochastic Dual Dynamic Programming
SLP	Stochastic Linear Programming
SSDP	Sampling Stochastic Dynamic Programming
STC	Site-C
UBC	University of British Columbia
WSR	Williston Reservoir

Glossary

Capacity	Maximum sustainable power that could be produced at any instant, usually measured by MW
Energy	Electricity produced/consumed over a period of time, usually measured by GWh
Forebay	The water level in a given water reservoir as measured at the generating station
Freshet	The period during which the snowpack melts and the resulting inflows feed into watersheds and reservoirs. In British Columbia and Northwest United States, this period is typically from late April to July
Hydroelectric	A system that generates electricity from free-falling water through turbines and generators
Multipurpose	Usually associated with water reservoirs that serves more than one purpose such as generating hydroelectricity and agriculture…etc.
Multireservoir	A group of water reservoirs
Planning Horizon	The length of time in the future the model is run for
Plant/ Project	A group of units (generators and turbines) connected to a man-made reservoir for the purpose of generating electricity
Power	Electricity produced/consumed at any instant in time, usually measured by MW
Run-of-the-river plant	A plant that has a reservoir with little to no storage capability
Shoulder Months	Months between seasons where neither domestic load nor prices are high; such as September and October
Stage	A unit of time over which an optimization process is undertaken in a DP or SDP model
State	A unit in a given space, for example storage space within a reservoir

Acknowledgments

I sincerely thank my supervisor Dr. Ziad Shawwash for his support, guidance and invaluable advice throughout my studies. He shared with me his experience and knowledge in both academia and in life. I am also grateful to Dr. Alaa Abdalla for his support during the time he was the manager of Reliability and Planning Group at BC Hydro. I wish to express my thanks to all the graduate students who were involved in the BC Hydro research program at the Department of Civil Engineering, UBC. The casual and formal discussions with them have provided me with invaluable insights in research and in life.

I would like to express my deep gratitude to Dr. Dave Bonser, my team lead at BC Hydro, for his advice, continuous support and most importantly, for listening to me and giving me the time to express my thoughts in both good times and bad times. I also thank my colleagues at work, Tim Blair and Andrew Keats, for their encouragement and support. Special thanks go to my previous manager, Kelvin Ketchum, and current manager at BC Hydro, Darren Sherbot, for supporting me and accommodating my needs during the ups and downs in my research and in my life.

Last but not least, I would like to thank my wife, Hend, from the bottom of my heart for her continuous support, encouragement and dedication. She has sacrificed her health and her career for me to succeed. Her sacrifices are unquantifiable and invaluable. Thank you my lovely wife.

Dedication

To my late mother, who supported me and prayed for me all the time up until the moment she left this world while I was thousands of kilometres away from her…

Chapter 1: Introduction

1.1. Research Goals

The main goal of this research is to develop an algorithm to solve the medium-term/long-term stochastic optimization problem using a practically acceptable representation of the inherent stochasticity and uncertainty in the modeled reservoir system. The algorithm is meant to be used as a potential decision support tool for operations planning of large-scale multireservoir systems such as the BC Hydro system. The second goal for the research is to provide a benchmarking tool for other more sophisticated models developed by the UBC/ BC Hydro research team. A further goal is to test the limits of the Stochastic Dynamic Programming technique that is used to develop the algorithm using the computer resources and programing capabilities available at the time this research was conducted. The driver is to provide guidance for future implementation of algorithms based on the same technique. The development of the modeling in this research was done in consideration of some of the challenges and the gaps outlined in the next section.

1.2. Challenges and Gaps

There are several shortcomings to the current models that were surveyed in the literature and the ones developed in-house and currently in use at BC Hydro. Some of the gaps identified are:

1. It is hard to reasonably represent the inherent stochasticity and uncertainty in reservoir systems without extensive computation cost;

2. Some of the best models that are currently used still need some manual guidance and/or several trial and error simulations in order to achieve the best possible outputs, which might jeopardize the final product by introducing human and other inherent errors;
3. Due to the curse of dimensionality and/or other modeling shortcomings, many of the models currently in-use cannot cover the desirable planning horizon or the actual state-space especially for long/medium-term planning purposes without jeopardizing the accuracy or the proper representation of the system modeled;
4. Several models and techniques seem very promising and have good potential, such as heuristic techniques, but unfortunately they have not been tried on large systems which typically entail more challenges; and
5. There is a need to develop more accurate estimates of the value of water in storage reservoirs for use in long-term capacity expansion planning studies and to improve the system operation in operations planning.

It is not claimed that the model developed in this research is able to cover all of the gaps mentioned above, but rather it is thought to add to the pool of knowledge of the UBC/BC Hydro team and reasonably represent the complexity of the systems modeled within the expected limiting factors of availability of computing resource and shortcomings of the technique and programming language used. The next section lists the contributions of this research.

1.3. Contributions of the Research

The following contributions are thought to be achieved by the current research:

1. Representing the stochasticity and uncertainty in the system in an acceptable form;

2. Concurrent modeling of six of the main generating facilities in the BC Hydro system on the Peace and Columbia Rivers which results in good representation of the system;
3. Providing practically acceptable representation of the water value functions that reflect the value of water in storage;
4. Preforming proper and extensive testing of the limits of the Stochastic Dynamic Programming technique and the AMPL programming language;
5. Extending the planning horizon up to 36 months with a monthly time step which is not possible for some of the models used currently that have comparable problem size; and
6. Developing generic and flexible code that could be easily enhanced, extended and used for different purposes including benchmarking and sensitivity analysis.

1.4. Implementation

An implementation of the Stochastic Dynamic Programming technique is used to develop the core SDPOM6R model. AMPL programming language is used to develop the model. The details of the approach and its implementation can be found in Chapter 3 of this manuscript.

1.5. Organization of the Thesis

The majority of Chapter 2 is dedicated to the survey of the dynamic programming optimization technique, which is the technique used in developing the model in this research. The development of the modeling approach is detailed in Chapter 3. The source of most of the materials included in this chapter is a report written by the author and co-authored by his supervisor and the author’s manager at BC Hydro (Ayad, et al., 2012). A briefing of the same materials is also included in (Abdalla, et al., 2013). Samples of the output and the results of the model are laid out and briefly discussed in Chapter4. Chapter 5 includes the results and the discussion of the extensive testing of the model developed in this research. The material of this chapter is adopted from a paper that was included in the proceedings of the HydroVision International Conference (Ayad, et al., 2013). Finally, the conclusion, and recommendations for future work are discussed in Chapter 6.

Chapter 2: Survey of Literature

2.1. Introduction

In this chapter, a survey is conducted on the different optimization techniques used in the fields of reservoir operation and operations planning. A number of the techniques are briefly introduced while others are thoroughly investigated due to their relative importance and relevance to the technique applied in this research.

Following this introduction, a brief and general review of the reservoir operation and management models is conducted in section 2.2.

Since the Dynamic Programming technique is used to develop the model in this research, the rest of the sections in this chapter are dedicated to this technique. The first few sections discuss the theories and principles the technique is based on. The last two sections of this chapter discuss the different variations and applications of the technique. They are sorted into two main categories: Deterministic Dynamic Programming techniques and Stochastic Dynamic Programming techniques.

2.2. Modeling of Reservoir Operation and Management

Scientists and engineers were, and still, interested in optimizing the operation of storage reservoirs. This interest ramped up in the early 70's with the increased access to computers. Computers enabled the development of variety of new approaches aiming at deriving the optimal operating policy (least cost/ highest profit). There are other reasons that drove the development of various reservoir optimization techniques for the most efficient use of water (Wyatt, 1996) such as: the increases complexity of the reservoir systems, the rise of energy prices in the 70's and the emerging public awareness of the ecological issues and their relation to water resources.

(Yeh, 1985) stated that the adoption of optimization techniques to be used in planning, management and design studies of water systems is one of the most important advances in the field of water resources during the 60's through 80's. Many of the studies conducted were successful in practice, especially for planning purposes while the same level of success was not attained in operations optimization (Yeh, 1985). Before that time, the most used approach to handle the operation of simple reservoirs systems, such as a single-purpose single reservoir, was the Critical Period Analysis (Hall, et al., 1969; Duranyildiz & Bayazit, 1988; Christensen & Soliman, 1989; Wyatt, 1996). Despite being simple, the single-purpose single reservoir system faces several challenges in operation either technical challenges, seasonal variations in parameters of the system, or running against license constraints. For the more complicated systems, such as multi-purpose multireservoir systems, several optimization techniques were developed. These include simulation, linear, nonlinear, and dynamic programming, applied separately or in combinations. (Wyatt, 1996) stated that "The advantage of these methods over critical period analysis can be attributed to their considering operating costs over the entire whole of the flow series simulated, rather than just minimising costs during the most critical periods of reservoir draw-down".

Nowadays, optimization techniques are adopted for the operations planning of all the reservoir systems regardless of their complexity. The selection of the most appropriate method is yet challenging. The main issue is that the deficiencies of these techniques are hard to quantify and it depends to large extent on the characteristics of each system (Wyatt, 1996). When the complexity of the system increases the number of possible operating policies and variable combinations increase exponentially, which increases the computational effort associated (Wyatt, 1996).

In order to make sure of the feasibility of the optimal solutions deduced by the optimization models or on other words, simulation modeling studies should also be performed (Labadie, 2004). For that, using a combination of both simulation and optimization models would be of a great benefit and in some cases a necessity in order to obtain the optimal policy.

2.3. Bellman's Principle of Optimality

From (Bellman, 1957), Bellman's principle of optimality is such that " An optimal policy has the property that whatever the initial state and the initial decisions are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision".

Bellman also defined Dynamic Programming as" the theory of multistage decision processes". The word "Dynamic" here means that this approach can handle the sequential or multi-stage decision problem and that is why it is efficient in making sequences of interrelated decisions (Nadalal & Bogardi, 2007).

2.4. Principle of Progressive Optimality

(Howson & Sancho, 1975) were the first to suggest this principle to use it to solve multistate dynamic programming problems. It is a successive approximation using a general two overlapped stages solution. One of the advantages of the algorithm is that it requires little storage resources. As might be inferred, it depends on or could be considered as an extension of the Bellman's Principle of Optimality. The Principle of Progressive Optimality states that "The optimal path has the property that each pair of decision sets is optimal in relation to its initial and terminal values" (Howson & Sancho, 1975). Using the principle of progressive optimality makes it unnecessary to discretize the state space (Yeh, 1985).

(Turgeon, 1981a) applied this principle to solve the short-term multireservoir operations scheduling problem.

A case study was performed using the application of the principle on four hydroelectric plants in series and it proved effective. Head variations, spills and time delays between upstream and downstream reservoirs were all taken into consideration (Turgeon, 2007). The author summarized the characteristics of the problem he was tackling as follows: nonlinear objective function, non-separable production functions, state and decision variables are bounded and the problem is stochastic due to the inflows into the system and the electricity demand. The advantages of this method compared to the traditional Principle of Optimality according to the same researcher are:

1. No discretization is required for the state variables;
2. Dimensionality problem is non-existing;
3. Non-convexity (such as in in production function) and discontinuity (like in cost function); are solvable using this technique unlike other techniques;
4. Convergence is monotonic and global optimum is reached; and
5. Relatively easy programming and fast execution.

2.5. Advantages and Challenges of Dynamic Programming (DP) Technique

Dynamic Programming (DP) has the capabilities to decompose the problem into sub-problems that can be solved sequentially over the planning periods (Abdalla, 2007). The DP approach is based on the Bellman's Principle of Optimality. The number of discrete DP variables equals the number of state values times the number of decision variables which is guaranteed to be less than in LP (Yakowitz, 1982).

Applications of the DP are very broad; however, the technique suffers from two curses that limit its applications to solve problems. The first curse is the curse of dimensionality which means that the problem size increases exponentially with increasing of the state-space which makes solving the problem in reasonable time very computationally expensive and time consuming (Bellman, 1957). (Yakowitz, 1982) stated that "the exponential growth in memory and CPU time requirements with increase in dimension of the state vector (i.e., the 'curse of dimensionality') is the greatest single hindrance to dynamic programming solution of large-scale optimal control problems". Some attempts were made to overcome the curse of dimensionality such as: making coarse grid, use of dynamic programming successive approximation, incremental dynamic programming, differential dynamic programming (Labadie, 2004). These variations will be discussed in later sections of this chapter. The advances in the computational capabilities of modern computers are one of the best solutions to the dimensionality problem. With those advances, the impact of this curse is alleviated but not eliminated. The second curse is the curse of modeling which means that when the real system that is being modeled gets complex, it is hard to model it using the DP technique (Bertsekas, 1995; Wyatt, 1996). The solution is to limit the number of storage states employed in the model when dealing with two or three reservoir systems (Wyatt, 1996). In other words, the solution to this problem is to under-represent the system or to approximate it.

According to (Wyatt, 1996; Nadalal & Bogardi, 2007; Abdalla, 2007; Pereira & Pinto, 1991), advantages of DP include:

1. It can be extended to multistage problems as well as stochastic case;
2. It handles discrete values and the nonlinearity;
3. The computational effort increases linearly when increasing the number of stages in the model;
4. It is suitable for problems where the decision variable takes a discrete or an integer form; and
5. It can handle nonlinearity, non-convexity, and even the discontinuity of the relations between the objective functions and constraints.

DP is widely applied and well-suited to the reservoir operation and operations planning problems. Its popularity comes from the possibility of translating of the water resources features such as nonlinearity and stochasticity into a DP formulation (Yeh, 1985).

2.6. Deterministic Dynamic Programming Techniques

2.6.1. Incremental Dynamic Programming Models (IDP)

In conventional DP, the state variables (usually set as reservoir storage or forebay in the reservoir operation problem) are discretized. Simultaneous derivation of operation policies for all the reservoirs and having a dense discretization is required in order to have close-to-global optimum operation policy in these systems (Nadalal & Bogardi, 2007). The disadvantage of this is that it makes it hard to use the conventional DP because of the curse of dimensionality (Nadalal & Bogardi, 2007) as previously mentioned.

The Incremental Dynamic Programming (IDP) technique was introduced by (Larson, 1968). Instead of using the entire state-space to search for the optimal solution as the DP does, IDP uses a pre-specified number of state variables to visit. In other words, the IDP algorithm restricts the state space to a corridor around the current given solution (Labadie, 2004). This idea has inspired the author of the research at hand in developing some solutions to increase the capacity of his model, which is discussed in Chapters 3, 4 and 5. IDP uses the recursive equation of DP to search for a better trajectory starting with some arbitrary feasible solution (initial trial trajectory) which serves as the first approximation of the optimal trajectory. The IDP creates what is called "corridor" around this initial trajectory. The corridor specifies the state variables to be visited in each time step in which the width of the corridor is the difference between the upper and lower bounds created around the state variable based on the initial trajectory. The trajectory obtained from this iteration is used as the new trial trajectory for the new iteration. The computation cycle continues until a convergence to the global optimal solution occurs. The convergence criterion is pre-specified for the system to prevent infinite calculations as the IDP solution might exhibit monotonic behavior (Nadalal & Bogardi, 2007).

IDP has some shortcomings such as: hardship of interpolation over the corridor and selection of discretization intervals and sensitivity of the IDP to the initially assumed storage trajectories (Labadie, 2004).

2.6.2. Differential Dynamic Programming Models

(Jacobson & Mayne, 1970) developed the Differential Dynamic Programming technique for the purpose of overcoming the dimensionality problems in DP. This technique uses an analytical solution, such as Taylor's series expansion, instead of discretization of the state space (Labadie, 2004; Abdalla, 2007), which makes it more suitable for application on the multireservoir systems. (Yeh, 1985) stated that when the system dynamics are not linear and the objective function is not quadratic, then the Differential Dynamic Programming is one of the best options. The differentiability of both the objective functions and the constraints is required to apply this technique (Labadie, 2004) which limits the application of this approach.

2.6.3. Dual Dynamic Programming Models (DDP)

DDP is inspired by the Benders' Decomposition Algorithm. (Pereira & Pinto, 1991) summarized the steps of the two-stage DDP algorithm as follows:

1. Set the initial value of approximate future value (cost) function, upper bound and lower bound;
2. Solve the approximate first stage problem;
3. Calculate the lower bound, if the convergence criterion is satisfied: stop otherwise, go to the next step;
4. Solve the second stage problem (calculate the approximate future value function and update the value of the upper bound);
5. Increment the number of vertices through which the approximate future value function is constructed; and
6. Go to the Step 2 again

The advantages of the DDP compared to other techniques such as the conventional DP are:

1. Discretization is not necessary;
2. It provides upper and lower bounds for each iteration;
3. It could be extended to solve multistage problems; and
4. It could be also extended to the stochastic case (SDDP), which will be discussed later in this chapter.

2.7. Stochastic Dynamic Programming Techniques

2.7.1. Conventional Stochastic Dynamic Programming (SDP)

Stochastic Dynamic Programming (SDP) is one of the most powerful and commonly used techniques to aid decision making in reservoir operation. SDP is well-established in long-term planning of multireservoir systems (Yeh, 1985). The inflows, electricity demands, and market prices are examples of stochastic variables that may be considered in the reservoir operations planning problem.

The optimal operating policy in SDP is derived using the Bellman's backward recursive relationship (Bellman, 1957). The convergence is determined by two criteria (Nadalal & Bogardi, 2007): stabilization of the incremental change in the optimal value according to the Bellman recursive formula and stabilization of the operating policy. The objective is usually to maximize the total benefit, which consists of current benefits coming from operations at present plus the discounted value coming from future use of stored water within the given planning/operating horizon.

As mentioned before, there are two major problems with using the SDP technique to solve large-scale problems: the curse of dimensionality and the curse of modeling.

(Arvanitidis & Rosing, 1970) developed one of the earliest applications of SDP in reservoir operation which had a primary goal of determining the optimal monthly hydropower generation of a hydroelectric system. The authors focused on the most important variables to alleviate the curse of dimensionality. The model output was compared to a well-established rule-curve operation.

(Stedinger, et al., 1984) introduced a medium-term monthly SDP model that forecasted the current period inflows using available information at that period. The Aswan High Dam on the Nile river basin in Egypt was used as a case study.

(Tejada-Guibert, et al., 1993) applied the SDP technique for three reservoirs and five thermal plants using a Markov Chain[1] and a discrete distribution that approximated a normal distribution. Penalty functions were used for power and water shortages.

(Druce, 1989; Druce, 1990) developed the Marginal Cost Model (MCM) for operations planning of the BC Hydro system using the SDP technique. The model uses weather sequences with equal probabilities to develop the monthly marginal value of water in the Williston Reservoir for a medium-term planning horizon. The uncertainty in inflows and market prices is accounted for in the model. At the time the model was created the Williston reservoir marginal value derived from the model results was used as a proxy for the system marginal price[2]. Later, and after several years of development by the System Optimization Group at BC Hydro, this model is now part of a bigger suite of models where it is coordinated with other models representing the other components of the system in order to derive the system marginal price.

---

[1] It entails the assumption that: the probability of an occurrence happening at a given stage in time depends only on the previous stage

[2] The word" Price" is usually used to refer to the marginal value of energy as opposed to value of water used to produce this energy

(Wyatt, 1996) developed two models, one for power-generation reservoir systems and another for water supply reservoir systems. SDP was used in the two models along with a simulation model.

(Turgeon, 2005; Turgeon, 2005) investigated the effect of incorporating multi-lag autocorrelation of inflows and the potential use of multi-lag autocorrelation for a single hydrologic variable for the solution of the SDP problem. One of the findings is that the flood and shortage risks decreased as power generation increased.

(Nadalal & Bogardi, 2007) applied the SDP technique to maximize the expected power generation from the Rantembe Reservoir in Sri Lanka. Operating policies were derived from an SDP model and then reservoir operation was simulated using historical inflow data. An improvement to the objective function was noted when storage discretization was refined but with the shortcoming of experiencing a polynomial increase in computational time.

Unlike other mathematical programming techniques, such as linear and non-linear programming, very few general purpose dynamic programming (DP) solvers are available. An example of software available for solving DP and SDP problems is the CSUDP model, which is generalized dynamic programming software developed at the Colorado State University (USA). This software can handle “multidimensional problems, stochastic problems, and certain classes of Markov decision processes” (Labadie, 2003).

The general SDP procedure, considering the inflow as the only stochastic variable, is illustrated in Figure 1.

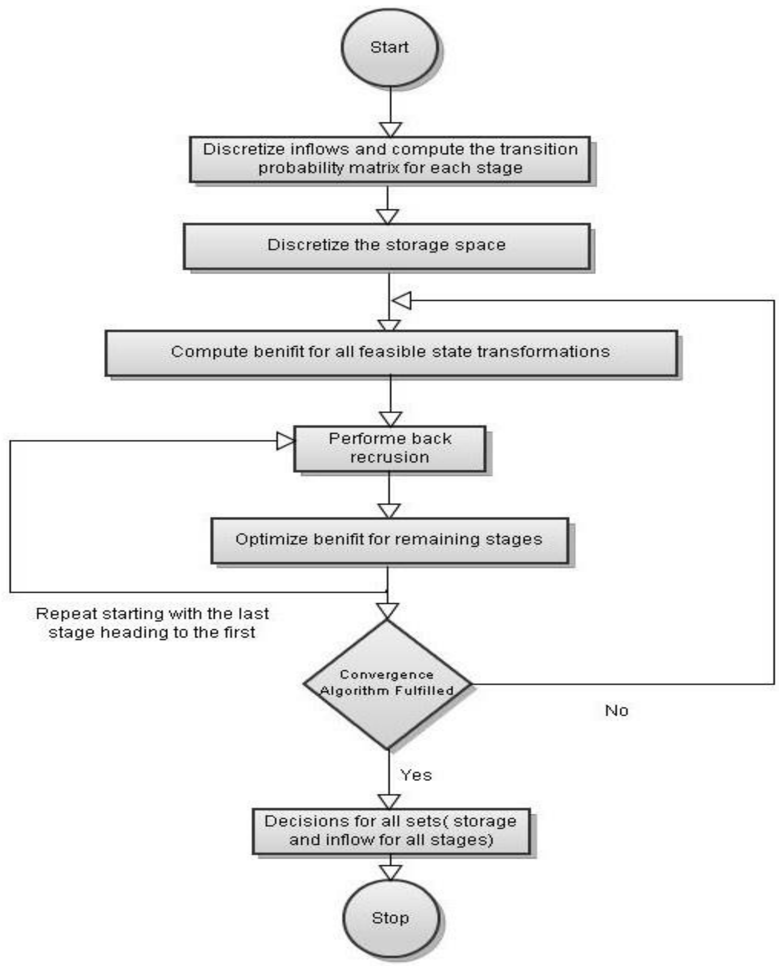


Figure 1: SDP Procedure (Nadalal & Bogardi, 2007)

2.7.2. Stochastic Dynamic Programming with Function Approximation

One of the most successful approaches used to alleviate the complexity of the reservoir system modeling problem is function approximation. Also, it is considered one of the most effective solutions to the dimensionality problem. In this method, state-space discretization is not needed any more as the near optimal value is expressed at each state/stage point in functional form. The value function can be approximated in many ways using linear function, polynomial function, piecewise-linear piecewise-polynomial or splines.

(Lamond, 2003) used the piecewise-polynomial functions to approximate the future value function. He applied this algorithm on a single hydroelectric reservoir with finite and discrete time horizon assuming a piecewise -inear concave reward for the production function.

2.7.3. Dynamic Programming with Successive Approximation (DPSA)

Stochastic Dynamic Programming using Successive Approximation (DPSA in short) is used to handle the problem of reservoirs in parallel (Turgeon, 1980; Christensen & Soliman, 1989). It optimizes one reservoir at a time. Unfortunately, the major drawback of this approach is that it does not take the dependence of operation of reservoirs on each other's energy content (storage) (Christensen & Soliman, 1989); in addition, with DPSA, the computation time and resources needed for the problem to converge are relatively large.

2.7.4. Aggregation and Decomposition SDP

(Arvanitidis & Rosing, 1970) and later (Turgeon, 1981b) adopted the method of aggregation/decomposition of a group of reservoirs in series into one equivalent reservoir. Each reservoir contributions were weighted according to its energy conversion factor (HK). The aggregation procedure was performed on storage, inflows and outflows.

This approach was proposed as a solution to the computational infeasibility problem the authors faced when applying a conventional SDP algorithm on more than three reservoirs.

The criticism to this method is that it does not account for parameters such as local constraints of reservoirs, which limits the application of this approach to the most systematic reservoir systems. In spite of that, the Aggregation and Decomposition SDP technique proved effective in long-term planning studies in cases where decomposed reservoir systems are sufficiently similar (Christensen & Soliman, 1989).

(Archibald, et al., 1997) added a conditional probability that allows switching between inflow scenarios at the beginning of each week. The modeling was done using four-state variables instead of two in the work of (Turgeon, 1981b).

2.7.5. Chance-constrained Programming Model and the Linear Decision Rules

Chance-constrained stochastic Programming (CCP) is a technique that applies the probability conditions on constraints. It is mostly suited for application on multipurpose reservoirs. The main advantage of this technique is alleviation of the problem of estimating the cost function, (Yeh, 1985). The CCP implicitly converts the stochastic problem to an equivalent deterministic problem that could be solved more easily (Abdalla, 2007).

Linear Decision Rules (LDR) can be considered as an add-in to the CCP. They relate the releases to storage and remove the dependency on random storage levels which allows the releases to be specified at the beginning of each time period (Yeh, 1985; Labadie, 2004). In other words, the optimization is no longer dependent on storage variables; alternatively, it depends on a decision parameter. In addition, LDR eliminates the mathematical complexity in CCP formulation. On the other hands LDR is considered as an additional constraint in itself and it does not take the complete stochasticity of the streamflow into consideration. Applying the LDR, the number of constraints gets smaller which reduces the probability of converging to an optimal policy, (Yeh, 1985).

2.7.6. Stochastic Dual Dynamic Programming (SDDP)

The SDDP is a combination of Stochastic Dynamic Programming (SDP) and stochastic linear or nonlinear programming using the Duality Theory[3] (with conservation of the convexity condition). The algorithm is based on the approximation of the cost-to-go functions (value functions) of SDP using a piecewise-linear function. The approximation mechanism can be done in two ways. The first is derived from Benders' decomposition method as in (Pereira & Pinto, 1991). The second method is performing the approximation on a grid as in (Read & George, 1990; Tilmant, et al., 2008). Using SDDP with the latter type requires it to be performed on a relatively coarse grid to avoid increasing the number of inflow alternatives exponentially (Lamond & Boukhtouta, 1996)[4]. The approximated cost-to-go function is obtained from the dual solutions of the problem at each stage. One of the most remarkable features of the algorithm is that it does not require the state- space to be discretized. By this, dimensionality problem is alleviated.

SDDP can be described as a two-stage problem. In the first stage, a decision is taken given a trial decision, and then a number m of second stage problem will exist (Pereira & Pinto, 1991; Lamond & Boukhtouta, 1996). For the multistage problems of the second stage, each sub-problem represents one stage corresponding to one period. At each stage, sub-problems of one period are being solved.

---

[3] It states that an optimization problem is viewed as a primal problem or a dual problem. Solving the dual problem provides a lower bound to the solution of the primal one.

[4] Assuming that inflows are the only stochastic variable in the problem

(Pereira & Pinto, 1991) applied the SDDP model on 39 hydroelectric plants in Brazil, 22 of which have reservoirs while the rest are run-of-the-river plants. The planning horizon used was 10 periods. Inflows were represented as independent random variables. The total number of variables and constraints were close to 150,000 for each stage problem.

(Tilmant, et al., 2008) applied a SDDP model on the Euphrates River. An assumption was made that the system is interconnected and fully integrated between Turkey and Syria which is not the case in reality. The reason, as the authors stated, is that they wanted to show how much benefits can be achieved from an integrated system planning approach. The modeled water usage included power generation and irrigation.

(Guan, et al., 2017) implemented the SDDP technique for the BC Hydro System. The model uses stochastic inflows and the Columbia River Treaty (CRT)[5] and other agreements to generate the water value function. For this purpose, two independent models were developed: the inflow model to generate forecasts of inflow volumes in the freshet period and monthly inflows and the CRT model to model operations for storage for flood control and other accounts. The SDDP implementation was benchmarked against several operations planning studies. An extensive testing and sensitivity analysis was performed to ensure the robustness and practicality of the model.

(Dias, et al., 2010) stated that "Nowadays, the SDDP methodology is used in many countries, as in the case of the Brazilian power system, where the SDDP with aggregated reservoirs is still the official methodology used for determining the long-term hydrothermal system operation, the short run marginal cost, among other applications."

---

[5] Details on the CRT can be found in Chapter 3

(Lamond & Boukhtouta, 1996) stated that it is not recommended to use SDDP in cases of nonlinearity or non-convexity. Also, (Dias, et al., 2010) explained that although the SDDP is one of the fastest techniques when it comes to computer time, it might give solutions that are far from the optimal solution, obtained by other techniques such as SDP, in case of not estimating the cost-to-go function properly for all the important parts of the problem's state-space.

2.7.7. Sampling Stochastic Dynamic Programming (SSDP)

(Kelman, et al., 1990) were the first to propose the Sampling Stochastic Dynamic Programming technique (SSDP). They defined it as "a technique that captures the complex temporal and spatial structure of the streamflow process by using large number of sample streamflow sequences". The authors presented this technique as a solution to the problems of poor representation of the system stochasticity and computation limitations that are inherently existent in traditional techniques .SSDP was originally designed for online operation using forecasted stream flows but later was extended to operations planning using historical stream flows (Lee & Labadie, 2007). The technique uses streamflow scenarios to represent the stochastic inflow processes. Like the deterministic optimization techniques, this approach still assumes the perfect foreknowledge in updating the optimal value function (Lee & Labadie, 2007). In other words, the current scenario continues with certainty into the future and the optimal value function is developed for the specific streamflow scenario. Moving from one inflow scenario to another requires the knowledge of the transition probability from each flow to another. One of the challenges in using the SSDP is initializing the terminal optimal value function as a boundary condition otherwise the model will empty all the reservoirs by the end of the time horizon (Lee & Labadie, 2007).

(Kelman, et al., 1990) developed a SSDP model that handles the complexity of the streamflow process by using a large number of sample streamflow sequences. The authors included what they called the "best inflow forecast" in the model as a hydrologic state variable to improve the reservoir operating policy. The model was applied on a case study to check its effectiveness on a hydroelectric system at the Feather River in California.

(Lee & Labadie, 2007), in their comparative state-of-the-world study, used the SSDP as one of the benchmarking techniques. The SSDP performance was good in some aspects while performed poor in others. To enhance its performance, the authors suggested using more reliable inflow forecasting models to be fed to the SSDP model which shows how sensitive the technique is to the inflow scenarios used.

(Schaffer, 2015) developed a SSDP model to maximize the value of water in storage in the BC Hydro system. The author investigated the use of different hydrologic inputs on the SSDP model performance such as: historical record data, inflows and forecasts generated from an autoregressive lag-1 model, and BC Hydro's ensemble streamflow prediction forecasts. Results revealed the significance of using forecasts earlier in the freshet period compared to the rest of the water year.

(Blair, 2017) developed a SSDP model for the Columbia River System, BC, Canada. Building on (Faber & Stedinger, 2001), the model (MUREO) incorporates probabilistic persistent reservoir constraints. The constraints are non-optional, persist over multiple stages, and are either a function of historic inflows, or a function of seasonal volume forecasts from a future stage. The model has two state variables: the non-treaty storage account[6] and the Kinbasket Reservoir storage. The model could be run for horizon for up to 6 years on a monthly time step. The implementation took advantage of the recent cloud computing and storage capabilities such that the user is able to run it either on a local computer or on the Amazon Cloud. To optimize the operations planning of the BC Hydro system, MUREO is run in an iterative fashion to coordinate with the aforementioned MCM (Druce, 1989; Druce, 1990) model and other models representing the rest of the BC Hydro system.

[6] Treaty here refers to the CRT

Chapter 3: Development of a Stochastic Dynamic Programming Optimization Model for Operations Planning of a Multireservoir Hydroelectric System

3.1. Background on the BC Hydro System

3.1.1. BC Hydro's System

The Province of British Columbia is one of the leading producers of hydroelectric power in Canada. The total installed generating capacity of the BC Hydro system is 12.05 GW (BC Hydro, 2017) of which more than 90% is hydropower. BC Hydro serves 95% of the population in British Columbia and produces about 80% of the total power generated in the province (BC Hydro, 2013; BC Hydro, 2017). There are 61 dams and more than 30 hydro plants in the BC Hydro system.

Figure 2: A Map of British Columbia Illustrating the Main Power Generation Plants and Local and Interconnected Transmission Lines, (BC Transmission Corporation, 2010)

The major river systems in BC are: the Peace system meeting 34% of electrical demand, the Columbia system meeting 31% of electrical demand, the Kootenay Canal and Seven Mile plants meeting 13% of electrical demand, and 23 small hydropower plants meeting 16% of electrical demand (BC Hydro, 2000)[7]. As of 2013, the remaining 6% of demand is served by independent power producers (IPPs) and thermal generating facilities (gas-fired and combustion turbines). The majority of the energy produced by the power system is from renewable sources with close to 2,000 MW coming from run-of-river projects, biomass projects and other renewable resources. BC Hydro meets the domestic electrical load of its service area and also trades energy in regional markets in Alberta, the Northwest USA and California through its subsidiary Powerex (BC Hydro, 2000; BC Hydro, 2013).

Optimizing the operation of the main storage reservoirs in the BC Hydro system is quite challenging due to the uncertainties that must be dealt with given the significant multi-year reservoir storage capabilities. The existence of a transmission network connecting the system with regional markets adds one more dimension to the complexity of the system.
It is not an easy task to optimize the planning of operations of the system under the various constraints that the BC Hydro system encounters such as: the physical generation constraints, environmental and non-power requirements, water licenses and international treaties, to name a few.

3.1.2. Columbia River Treaty

[7] These percentages are averages and vary from year to year

The Columbia River Treaty (CRT) between Canada and the United States was ratified in 1964. The implementation of the treaty is the responsibility of the Canadian entity (BC Hydro) and the two American entities (Bonneville Power Administration and the US Army Corps of Engineers). The main features of the CRT include: building large storage reservoirs (completed in the 60's and 70's), streamflow regulation, sharing flood control benefits, sharing power generation benefits, determining the authorities on evacuation of flood control space, water diversion, mechanisms to resolve emerging disputes and the options to terminate or extend the treaty. Of concern to this research, the CRT imposes a number of constraints on Canadian reservoir operations and these constraints are included in operations planning models developed/used by BC Hydro.

3.1.3. Representation of the BC Hydro System in the Modeling Algorithm

Six plants and their associated reservoirs on two river systems are explicitly included in the optimization model. Three on the Peace River: G.M. Shrum (GMS) and Williston Reservoir, Peace Canyon (PCN) and Dinosaur Reservoir downstream of GMS, and Site-C[8] (STC) and Site-C Reservoir downstream of PCN. The other three are on the Columbia River: Mica (MCA) and Kinbasket Reservoir, Revelstoke (REV) and Revelstoke Reservoir downstream of MCA, and Arrow Lakes Hydro (ALH) and Arrow Lakes Reservoir downstream of REV. All these plants are optimized except ALH due to the complexity inherent in modeling the CRT. ALH generation is fixed along with power generation from other sources in BC.

---

[8] Site-C is currently under construction and is included in some long-term resource forecasts.

A storage plant[9] is defined as a plant that has multi-year storage capacity and thus its storage is modeled as a state variable in the optimization model. GMS and MCA are modeled as storage plants because they are immediately downstream of the two largest reservoirs in the BC Hydro system. A run-of-the-river plant (ROTR) is a plant that does not have much storage capability and it simply passes all the water it receives within a period that is less than the time step modeled. PCN, STC, REV and ALH [10] are modeled as run-of-the-river (ROTR) plants.

3.2. Approach and Context

The model developed as part of the current research is particularly concerned with the two largest reservoirs in the BC Hydro system, Williston and Kinbasket reservoirs. In addition to these two reservoirs there are four other reservoirs that are represented as ROTR. Several constraints and characteristics that are related to the seix reservoirs in particular or to the BC Hydro system in general are included in the model. The name of the model is Stochastic Dynamic Programming Optimization Model for Six Reservoirs or SDPOM6R for short.

The development of this model was part of a capital project at BC Hydro, *The Water Value Project* (Abdalla, et al., 2013), to develop multi-reservoir stochastic optimization models to generate water value and marginal value functions that best represent the expected

---

[9] The words “plant” and “project” are used synonymously to refer to the same facility.

[10] ALH is an exception here since although as it has quite large storage capabilities it was modeled as a ROTR plant to simplify the algorithm.

value of water in storage and the marginal value of the multi-year storage reservoirs in the system. The objective is to obtain the optimum operating policies to maximize the revenue of BC Hydro from reservoir operation.

This model was developed primarily to be used as a benchmarking tool for multiagent reinforcement learning model (MARL), which is under development by the same team as part of the *Water Value Project*, as well as against other already-developed or under-development models such as Reinforcement Learning Reservoir Optimization (RLROM) Model, Stochastic Dual Dynamic Programming (SDDP) model, Sampling Stochastic Dynamic Programming (SSDP) model and Stochastic Linear Programming (SLP) model.

3.3. Objectives of the Model Development

The SDPOM6R model is developed to:

1. Improve on currently used models of multireservoir long/medium term operations planning;
2. Increase the number of reservoirs taken into consideration in the model (better representation of the real system);
3. Evaluate the marginal value of water for multi-reservoirs (better operations planning); and
4. Capture part of the complexity and the inherent uncertainties of the system.

3.4. Modeling of the Problem

3.4.1. State Variables

State variables in the model are the storage states, which are structured in two groups in the model: initial storage states and terminal storage states.

The initial storage state is fixed along all the time steps to enable the backward recursion value iteration procedure. The range of the initial storage states for each reservoir is controllable and can be changed to obtain a measure of the sensitivity of reservoir operation within a certain range of storage. For normal operations planning, the range is usually chosen to cover the storage state trajectory that will be described later in this manuscript.

The terminal storage states for each reservoir vary with initial states, time steps, release decision and inflow values. For each time step and initial state, each single release decision and each single inflow value, there is a unique range of terminal states covering all the possible states that can be visited given the historic operation and the physical limits. The bounds of the range of the terminal storage states are also controllable and usually lie within the storage state trajectory.

3.4.2. Decision Variables

There is one decision variable per reservoir used in the SDPOM6R, which is the total release from each plant. The discretized values of the total release are calculated from the plant spill limits which are provided to the model in a data file. For each set of total release data, another set of values is deduced from the turbine release limits. This set is then used to calculate the generation corresponding to that release given the condition that it cannot exceed the following two values: the discretized plant spill value (the decision variable), and the difference in storage between the starting storage state and the terminal storage state given a specific value of the inflow.

3.4.3. Stochastic Variables

The expected inflow to each reservoir per time step is represented by a discrete probability distribution functions. This distribution is developed from 60 years of historical monthly inflows. These historical values are obtained from the BC Hydro records, CRO/Flocal and other sources internal to BC Hydro. The number of bins used in each discrete distribution is variable depending on the range of inflow values at hand and the inflow step (increment) adopted. A frequency analysis was carried out for different discretized inflow increments that were determined by the range of historical monthly inflows. In the current implementation of the algorithm, these increments determine the discretization of other state spaces such as the storage state-space and plant release decision-space. Several references, such as (Nadalal & Bogardi, 2007), recommend that if the actual distribution does not have a zero bin then a zero bin with very low probability should be added to the distribution to achieve a representative state probability transition matrix. In winter months (December, January and February) the range of historical inflows is normally narrow and 4 to 5 bins were found to be sufficient to cover the distribution, whereas in the freshet months (May and June), when the inflow is snowmelt-dominated, the distribution range is large and the number of bins range from 8 to 11. At all times the minimum number of bins was set to 4. **Figure 3** illustrates an example of the discretized June inflow probabilities for the Williston Reservoir.

Discretization of the inflow-space is performed only for the storage reservoirs, i.e. Williston and Kinbasket reservoirs. The inflow to ROTR reservoirs was calculated using monthly regression equations relating their inflow to the inflow and probability of their associated upstream storage projects.

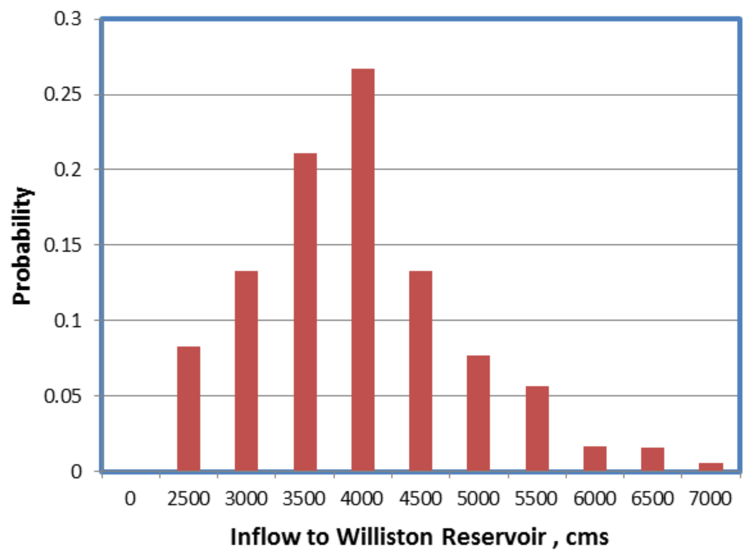


Figure 3: Probability Distribution for the Inflows to the Williston Reservoir for the Month of June

3.4.4. Space Discretization and Transitions

3.4.4.1. Discretization of State-space and Decision-space

One of the important considerations affecting the accuracy and shape of the value function in the SDP algorithm is how to properly discretize the state and decision space variables in the optimization problem (Nadalal & Bogardi, 2007). For the case discussed here those include: the storage state-space, the inflow-space, and the decision-space of plant releases. It is well known that finer discretization yields better results, but unfortunately this is limited by the available computational resources and the time needed to solve the problem. The sensitivity of the algorithm and the quality of the solution, to the discretization increment are discussed in Chapter 5.

3.4.4.2. Transition Matrix and State Transitions

As discussed earlier, inflow is the only stochastic variable that is explicitly represented in the SDP solution algorithm of the SDPOM6R model. The inflow probability can be used to calculate each reservoir's state transition probabilities for a given initial storage state. To calculate these state transition probabilities, the procedure outlined in the following steps was followed:

1. For each reservoir and for a given hydraulically feasible transition from one storage state to another, the probability of the state transition is set equal to the inflow probability and is used to calculate the state transition probability. If the transition is not hydraulically feasible then its probability is set equal to zero;
2. The global[11] state transition probability (joint probability) for the system from a global state to another global state is equal to the product of the transition probabilities; and
3. The global transition probabilities are then used to calculate the expected values of the different terms in the value function equation[12].

Calculation and storage of the transition matrix is one of the major challenges that arise when applying the SDP technique to this type of problems. This challenge was primarily addressed by using a dynamic storage range corridor, which is discussed later in the manuscript.

---

[11] It involves all storage reservoirs modeled (e.g., the Williston and Kinbasket reservoirs in this context)

[12] Discussed in Problem Formulation later in this chapter

3.4.5. Storage-Generation Curves (HK Curves)

One of the most important aspects of the hydropower problem is how the HK values are accurately calculated as that level of accuracy affects the generation calculations and accordingly the optimum policy. For each reservoir and from historic data, a 3rd degree polynomial regression equation is generated and used to deduce the proper HK values for each transition storage state. The equation used in the model calculates the HK value as a function of both starting storage state and terminal storage state (linearly interpolated between the two states).

3.4.6. Storage-Forebay Curves

Forebay elevation is not used in any of the core calculations or constraints in the model; instead, all the calculations are done using the storage volume of the reservoirs. However, the forebay elevation is calculated as a by-product of the model. A regression equation between the storage and the forebay elevation for each reservoir is developed as can be seen in **Figure 4**. The sources of the data used to develop the regression equations is the CRO/Flocal. These equations are then used in the model to calculate the forebay elevation as a function of the storage for both the initial and terminal states.

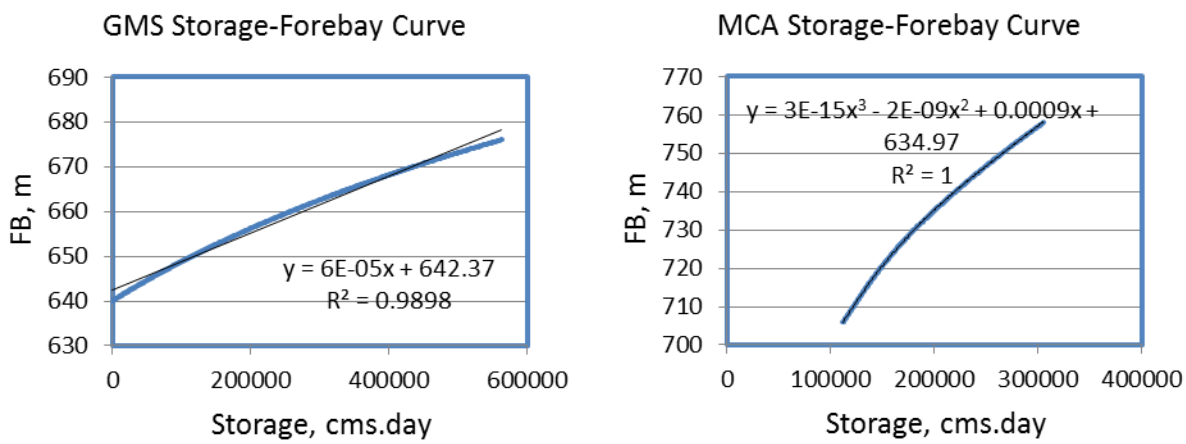


Figure 4: GMS and MCA Storage-Forebay Curves

3.4.7. Representation of Unit Outage

It is important to note that the outages that are tackled here are the scheduled/planned outages of the plants and not forced outages. The latter are covered through a coefficient for contingency reserves/availability on generation. The main source of the outage schedule data used for the outage representation in the SDPOM6R is the data files of the GOM model (Fane, 2003). First, the numbers from that file are mapped to the binary system to represent the outages of units and the duration of each outage. After that, the outages are aggregated to monthly time steps. **Table 1** shows the outage schedule for one of the plants (GMS). The outage percentage is multiplied by the generation to calculate the maximum possible generation for each time step per plant.

Table 1: Conversion of Binary Outages to a Generation Factor

Decimal	0	383	495	511	831	999	1007	1023	Total	Generation Factor
Binary	0	101111111	111101111	111111111	1100111111	1111100111	1111101111	1111111111		
Month										
1	0%	0%	0%	0%	0%	0%	68%	32%	100%	0.932
2	0%	0%	0%	0%	0%	0%	0%	100%	100%	1.000
3	0%	0%	0%	0%	0%	0%	0%	100%	100%	1.000
4	0%	0%	0%	0%	0%	0%	0%	100%	100%	1.000
5	0%	0%	0%	0%	0%	0%	0%	100%	100%	1.000
6	0%	0%	0%	0%	0%	0%	23%	77%	100%	0.977
7	0%	0%	0%	0%	0%	100%	0%	0%	100%	0.800
8	0%	48%	23%	0%	10%	19%	0%	0%	100%	0.807
9	0%	23%	0%	13%	63%	0%	0%	0%	100%	0.816
10	0%	0%	0%	35%	0%	0%	0%	65%	100%	0.965
11	0%	0%	0%	0%	0%	0%	0%	100%	100%	1.000
12	0%	0%	0%	0%	0%	0%	0%	100%	100%	1.000

3.4.8. Load-Resource Balance

Calculation of generation in the SDPOM6R model is governed by the load resource balance equation. This equation can be simplified as:

For each time step,

$\sum$ Generation (modeled hydroplants + un-modeled hydroplants + IPPs + thermal) – Load ± Forward contracts = Net Export (surplus or deficit) **Equation 1**

$\sum$ **Generation:** represents the sum of all energy feeding into the system including the modeled hydroelectric plants (GMS, PCN, MCA, REV, ALH, and STC), the Independent Power Producers (IPPs), thermal plants and other sources of energy to the system including the un-modeled hydroplants.

Load: represents the domestic demand that has to be satisfied as a first priority by the BC Hydro system.

Import/ Export: The sum of left hand side (LHS) of the equation is considered an import if it is a negative number and an export if it is a positive number, and is subject to the transmission line limits which are inputs to the model as well.

Forward contracts: are the forward sales that BC Hydro is committed to fulfill during the planning horizon, which could be either imports or exports for each time step.

Figure 5: Elements in the Load-Resource Balance Equation

Figure 5 illustrates the load, summation of IPPs, thermal, and other sources of generation and the import and export limits. Any deficit will be covered by the generation from the hydroelectric plants and imports if needed; any surplus will be exported or spilled if needed. For instance, the area under the light blue line with red markers represents the deficit that should be covered by the hydroelectric plants and the imports (Abdalla, 2007).

3.4.9. Representation of Prices

The approach followed in this work is to adjust the average forecasted monthly energy prices at the Mid-C trading hub by applying a price multiplier that captures the effect of the variability of inflow conditions from the average water conditions in the Pacific Northwest region. In wet water years, using the forecast total seasonal flow at The Dalles near the mouth of the Columbia River, the price multiplier is less than 1 and the corresponding regional electricity market prices are less than average. Under dry conditions the multiplier is greater than one and the prices are above the average. The total inflow to the system is calculated and correlated with the Columbia River inflows at The Dalles to calculate the price multipliers which are used to scale the Mid-C market prices for different scenarios using monthly regression equations as shown in **Figure 6**.

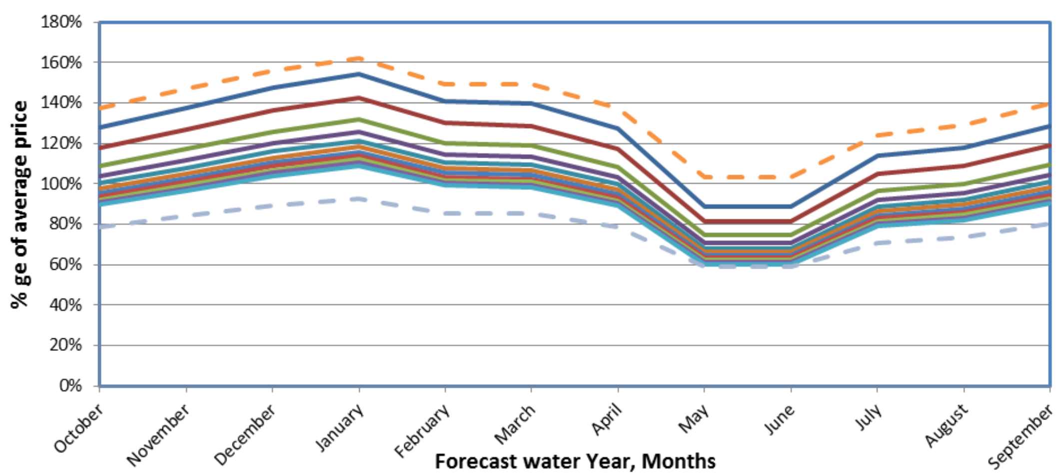


Figure 6: Prices for a Forecast Water Year for Different Total System Inflow Scenarios

The coefficients of those equations are extracted for each month in each future water year and are then used to calculate a new set of regional price scaling factors. Wheeling charges are then added or subtracted to create import and export prices, respectively, at the BC-US border.

3.4.10. State-space Discretization and Generation of Discretized Values

One of the governing factors in the current model and in the SDP technique in general is how to properly discretize the state space, as the value function and its shape are directly impacted by how the sta-e space is discretized. Furthermore, from what had been discovered through the SDPOM6R modeling process, not only the state-space but also both of the inflows and the releases need to be optimally discretized. Also, they all need to be of the same discretization step (increment) at least in each time step for each reservoir. The size of the discretization step is limited by the computation capacity of the computer/server that the model execution is performed on as well as the capacity of the coding language (AMPL in this case).

The code was written and indexed in a way that the discretization could be hybrid, which means that each reservoir can have its own discretization step and also for each reservoir the discretization step can vary by each time step.

3.4.11. Approximations

The approximations that are used in developing the code itself, in creating the data or in developing any regression equations are:

1. The maximum and minimum bounds of the state space are inputted as rough values; and these limits are first rounded to nearest discretization step; and
2. Several regression equations are used in the model such as price-inflow regression equation, HK regression equations and Forebay-storage regression equations.

3.5. Problem Formulation and Solution Algorithm

3.5.1. Objective Function and Calculation of the Value Function

The objective is to maximize the value of the hydropower resources. This is accomplished by optimizing the system dispatch to capture electricity market opportunities in the planning horizon while satisfying the domestic load. The objective function is expressed in the following equation[13].

$$\mathrm{PV}_t(\mathbf{S}_{j,j^*,t}) = \max_{\mathbf{a}_{j,j^*,t}} \left\{ \begin{array}{c} \mathrm{B}_t\left(\mathbf{S}_{j,j^*,t}, \mathbf{a}_{j,j^*,t}\right) + \\ \gamma * \sum_{\mathbf{S}_{j,j^*,t+1}} \left[\Pr_t(\mathbf{S}_{j,j^*,t}, \mathbf{S}_{j,j^*,t+1}, \mathbf{a}_{j,j^*,t}) * \mathrm{PV}_{t+1}(\mathbf{S}_{j,j^*,t+1})\right] \end{array} \right\}$$

Equation 2

where,

$$\mathrm{B}_t\left(\mathbf{S}_{j,j^*,t}, \mathbf{a}_{j,j^*,t}\right) = \mathrm{CR}_t\left(\mathbf{S}_{j,j^*,t}, \mathbf{a}_{j,j^*,t}\right) + \mathrm{IC}_t\left(\mathbf{S}_{j,j^*,t}, \mathbf{a}_{j,j^*,t}\right) + \mathrm{ER}_t\left(\mathbf{S}_{j,j^*,t}, \mathbf{a}_{j,j^*,t}\right) + \mathrm{DR}_t - \mathrm{SP}_t(\mathbf{S}_{j,j^*,t}, \mathbf{a}_{j,j^*,t})$$

Equation

[13] Characters used in equations are defined in the List of Symbols in the beginning of this manuscript.

$where, \quad j, j^* \in J \; for \; j \neq j^*$

As can be seen from **Equation 2** that the present value of water in storage, PV (·) is calculated as the sum of two terms: the expected income B (·) for the set of decisions in the current period, and the discounted expected future value of water in storage in the next stage. **Equation 3** shows that the expected income, or policy income, is the expected value of contract sales revenue (*CR*), cost of imports (*IC*), revenue from exports (*ER*), and the revenue/cost of satisfying the domestic load (*DR*). Also, a spill penalty function (*SP*) is added to discourage solutions requiring spill.

There are three classical methods that can be used to solve **Equation 2**: policy iteration, linear programming, and value iteration. Value iteration is the most commonly used method and is adopted in this work. **Figure 7** shows the application of the value iteration method in the model through a process called Backward Recursion which updates the value function starting with the last time step (stage) in the planning horizon and moving backwards.

After the value function converges, the marginal value of energy can be computed by differentiating the value function with respect to storage for a given state, as shown in the following equation, **Equation 4**.

$$\mathrm{MVW}_t(\mathbf{S}_{j,j^*,t}) = \frac{\partial \mathrm{PV}_t(\mathbf{S}_{j,j^*,t})}{\partial \mathbf{S}_{j,t} * \mathrm{HK}_{j,t}(\mathbf{S}_{j,t})} \qquad \textbf{Equation 4}$$

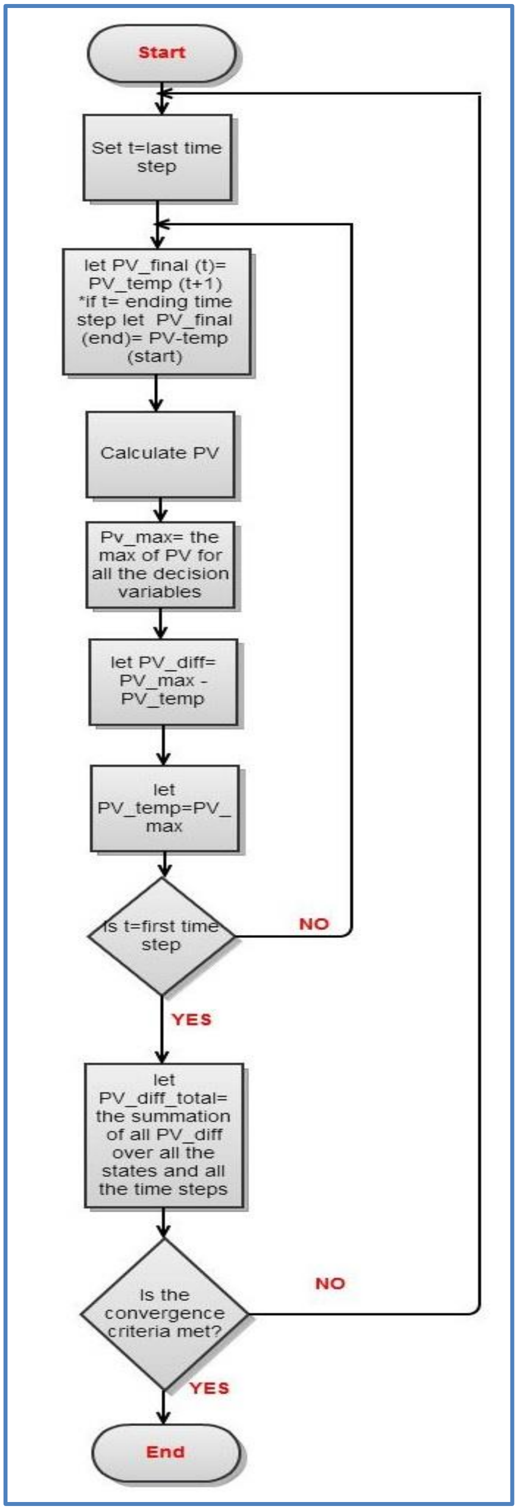


Figure 7: Backward Recursion Value iteration Procedure

3.5.2. Main Constraints

The objective function above is subjected to the following constraints:

1. *Load-resources balance:*

$$s.t. \sum_{j=1}^{\mathbf{J}} G_{j,t}\left(\mathbf{S}_{j,t}, \mathbf{S}_{j,t+1}, \mathbf{a}_{j,t}\right) + \sum_{k=1}^{\mathbf{K}} \mathrm{g}_{k,t}\left(\mathbf{I}_{j,t}, \mathbf{a}_{j,t}\right) + GR_t + CI_t - CE_t$$

$$- \mathrm{ST}_t\left(\mathbf{S}_{j,j^*,t}, \mathbf{I}_{j,j^*,t}, \mathbf{a}_{j,j^*,t}, \mathbf{S}_{j,j^*,t+1}\right)$$

$$= \mathrm{L}_t$$

Equation 5

2. *Mass (hydraulic) balance:*

For the storage projects (WSR and KBT),

$$\mathrm{s.t.}\ \mathbf{S}_{j,t} + \mathbf{I}_{j,t} - TQ_{j,t}\left(\mathbf{S}_{j,t}, \mathbf{S}_{j,t+1}, \mathbf{a}_{j,t}\right) - SQ_{j,t}\left(\mathbf{S}_{j,t}, \mathbf{S}_{j,t+1}, \mathbf{a}_{j,t}\right) = \mathbf{S}_{j,t+1}$$

Equation 6

For ROTR projects,

$$s.t.\ \mathbf{i}_{k,j,t}\left(\mathbf{I}_{j,t}\right) + \mathbf{a}_{j,t} - tq_{k,j,t}\left(\mathbf{I}_{j,t}, \mathbf{a}_{j,t}\right) - sq_{k,j,t}\left(\mathbf{I}_{j,t}, \mathbf{a}_{j,t}\right) = 0$$ **Equation 7**

where $i_{k,j,t}, tq_{k,j,t}$, and $sq_{k,j,t}$ are computed only when storage project *(j ∊* **J***)* and ROTR project *(k ∊* **K***)* are hydraulically connected.

3. *Storage limits:*

$s.t.\ LS_{j,t} \leq \mathbf{S}_{j,t} \leq US_{j,t}$

Equation 8

3. *Turbine flow limits:*

For the storage projects,

$$s.t.\ TQ\_min_{j,t}\left(\mathbf{S}_{j,t}, \mathbf{S}_{j,t+1}\right) \leq TQ_{j,t}\left(\mathbf{S}_{j,t}, \mathbf{S}_{j,t+1}, \mathbf{a}_{j,t}\right) \leq TQ\_max_{j,t}\left(\mathbf{S}_{j,t}, \mathbf{S}_{j,t+1}\right)$$

Equation 9

For ROTR projects,

$$s.t.\ tq\_min_{k,t} \leq tq_{k,j,t}\left(\mathbf{I}_{j,t}, \mathbf{a}_{j,t}\right) \leq tq\_max_{k,t}$$

Equation 10

4. *Total plant discharge limits:*

$$s.t.\ PQ\_min_{h,t} \leq PQ_{h,t}(\mathbf{a}_{j,t}) \leq PQ\_max_{h,t}$$

Equation 11

5. *Transmission limits:*

$$when\ the\ spot\ trade\ activity\ is\ import,$$

$$\mathrm{s.t.}\ \mathrm{ST}_t\left(\mathbf{S}_{j,j^*,t}, \mathbf{I}_{j,j^*,t}, \mathbf{a}_{j,j^*,t}, \mathbf{S}_{j,j^*,t+1}\right) + \mathrm{CI}_t \leq LTI_t$$

Equation 12

$$when\ the\ spot\ trade\ activity\ is\ export,$$

$$\text{s.t. } \mathrm{ST}_t\left(\mathbf{S}_{j,j^*,t}, \mathbf{I}_{j,j^*,t}, \mathbf{a}_{j,j^*,t}, \mathbf{S}_{j,j^*,t+1}\right) + \mathrm{CE}_t \leq LTE_t$$

Equation 13

6. *Generation limits:*

For the storage projects,

$$s.t.\ \ G\_min_{j,t} \leq G_{j,t}\left(\mathbf{S}_{j,t}, \mathbf{S}_{j,t+1}, \mathbf{a}_{j,t}\right) \leq G\_max_{j,t}$$

Equation 14

For ROTR projects,

$$s.t.\ \ G\_min_{k,t} \leq g_{k,t}\left(\mathbf{I}_{j,t}, \mathbf{a}_{j,t}\right) \leq G\_max_{k,t}$$

Equation 15

3.5.3. Solution Algorithm

The solution algorithm is illustrated in **Figure 8** and it consists of three main modules. The first is the *Discretizer* that discretizes the storage state-space and the release decision-space. The main data sets used by this module are the storage corridor for the storage projects for the planning horizon, which will be detailed in later sections, the discrete inflow probability distribution for the planning horizon, and the limits on plant discharge for the storage projects for the planning horizon. This module prepares the discretized starting and terminal storage states for each time step as well as the discretized releases. These outputs are used in the second module.

The second module is the *SDP* model which implements the SDP algorithm and applies the constraints in each time step. These outputs include: turbine flow, spill, total system inflow, generation, transition probability, forebay, spot electricity market trade, policy income, and the marginal values of water and energy.

The third module is the Value Iteration module which develops the water value functions for the time horizon considered.

In addition to the main modules discussed above, there are eight smaller modules that perform other calculations including the inflow regression analyses for ROTR, HK, price multiplier calculations, and a module to calculate the capacity limits for plant generation and turbine discharge limits.

The details and the code of the modules mentioned above can be found in Appendix A.1

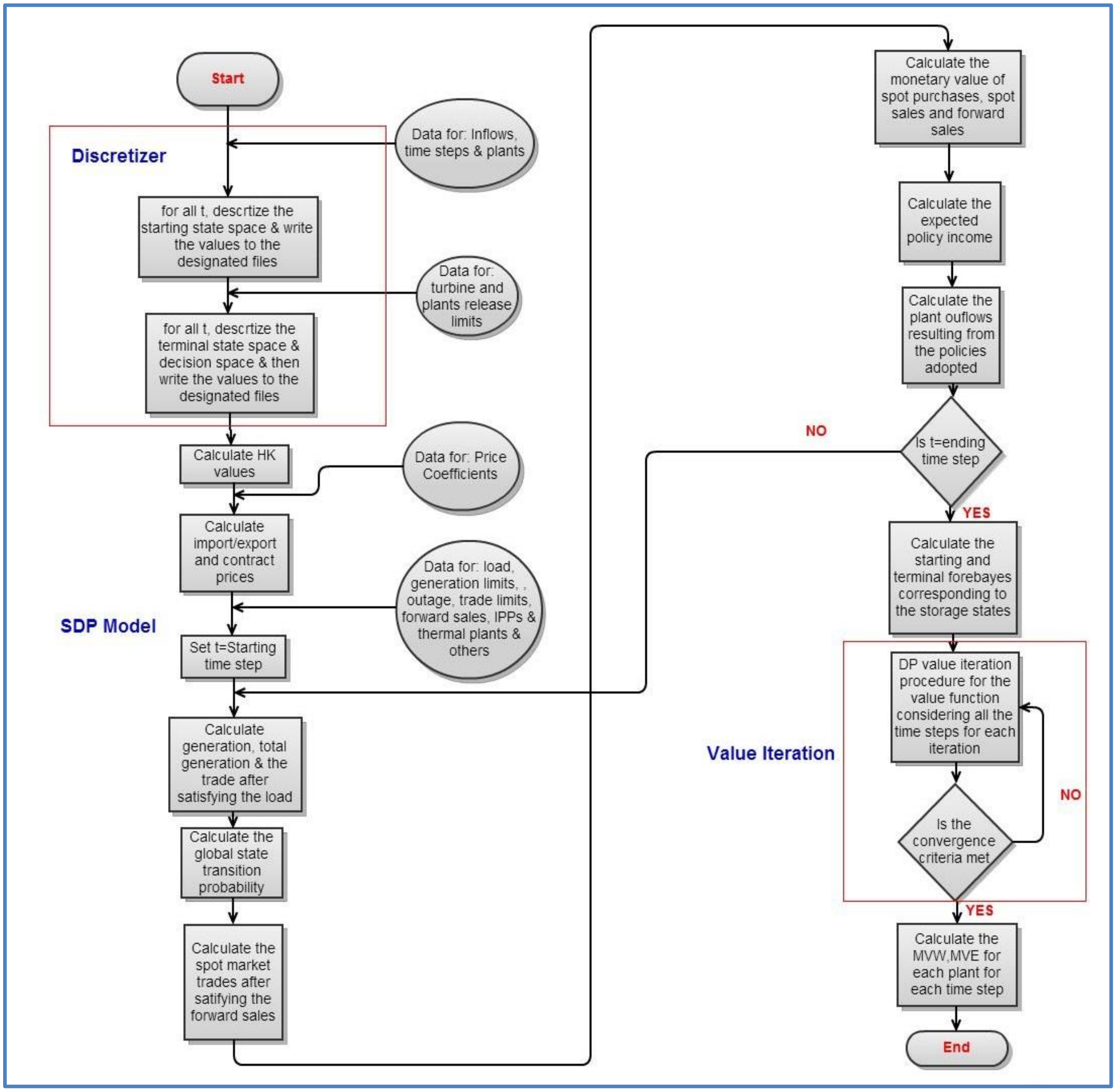


Figure 8: Flowchart of the SDPOM6R Model

3.6. Capabilities

Extensive testing of the model has shown that it is able to solve the problem for up to 36 monthly time steps (3 years) producing practicaly acceptable water value and marginal value functions up to a problem size of ~ 164 million states per time step.

It is expected that, with further enhancements of the algorithm, the model could handle a much larger problem and could also be extended to include more state variables. The development of a dynamic storage-state corridor using the simulated historical data significantly accelerated the convergence of the algorithm and allowed the solution of larger problems, but care must be taken to ensure that the derived solutions are robust and globally optimal.

Flexibility is one of the most important features of the current model, and it is well known that SDP solution algorithms are not typically very flexible and are custom built for specific systems. This flexibility is a result of enhancements made to the algorithm and the formulation of the model coding in AMPL, which provided the following advantages:

1. The increment of all the state-spaces and the storage corridor can be controlled and changed easily for each reservoir for each time step;
2. The problem can be solved for one storage reservoir while fixing the states of the other storage reservoir[14] simply by changing few control parameters; and

[14] This option allows the model to solve for one storage reservoir given a fixed state of the other reservoir.

3. The model can be run either directly on a personal computer (or a server) using the command prompt or by using a scheduling program that uses simple scripts to optimize the use and sharing of computing assets.

3.7. Limitations

Because of the nature of the SDP technique, as well as the complexity of the system investigated herein, there are several limitations of the algorithm. Some of these limitations could be overcome with the advances in computing resources and the programming languages capabilities. Other limitations are likely to persist in future versions of the model or extension of it. Some of these limitations are: the stochasticity of domestic electricity load and prices is not currently represented, the accuracy of the existing regression equations representing the prices and other variables could be improved, and the model only contains variables representing six major reservoirs in the BC Hydro system while fixing the output of other resources and therefore it simplifies the real system; in addition, several other environmental or operational constraints are not modeled.

3.8. Issues Experienced in Model Development and Code Run

Several problems were experienced in developing the model as well as in its outputs:

1. The shapes of the value function and the marginal value of water function heavily depend on the discretization used and the inflow values and their probabilities. It was a challenge to find the right combination of this data that produce a practically acceptable shape of each function;

2. As the problems of dimensionality and modeling are inherent in the SDP modeling, it was expected to experience problems related to these two curses;
3. Sometimes the resulting output files are too big to store;
4. Indexing is cumbersome which is partially due to the nature of coding in AMPL and partially due to the size of the problem at hand; and
5. Both the transition probability and value iteration calculations are sensitive to changes in indexing or other changes and tracing these sensitivities is challenging.

Chapter 4: Results

4.1. Introduction

In this chapter few samples from the model results are illustrated in graphical form. All the results illustrated are normalized for readability and data confidentiality reasons. Discussion of the results is kept to minimum as more details are presented in Chapter 5.

4.2. Sample Results for Two Cases of State-space Discretization

There are several cases that have been thoroughly investigated and tested (please see **Table 6)** in Chapter 5. For brevity, only samples of two cases of them are shown in the following graphs. The cases are:

CASE A: Storage state increment is 500 cms-day for the Williston Reservoir and 1000 cms-day for the Kinbasket Reservoir (CASE A); and

CASE B: Storage state increment is 1000 cms-day for both of the Williston Reservoir and the Kinbasket Reservoir.

Figure 9 below shows a three dimensional graphical representation of the water value functions of both of the Williston Reservoir and the Kinbasket Reservoir. As expected, the value of water in storage is lower when there is less water in both reservoirs. This value increases with any incremental increase in the amount of water in storage in one or both of the reservoirs. This increase continues until a certain point where the surface almost levels. This means that the incremental increase of water in storage has little to no effect on the total value of water in storage. It could be noted that the farthest tip of the surface (top right) slightly drops after the surface has leveled and that could be attributed to the boundary conditions.

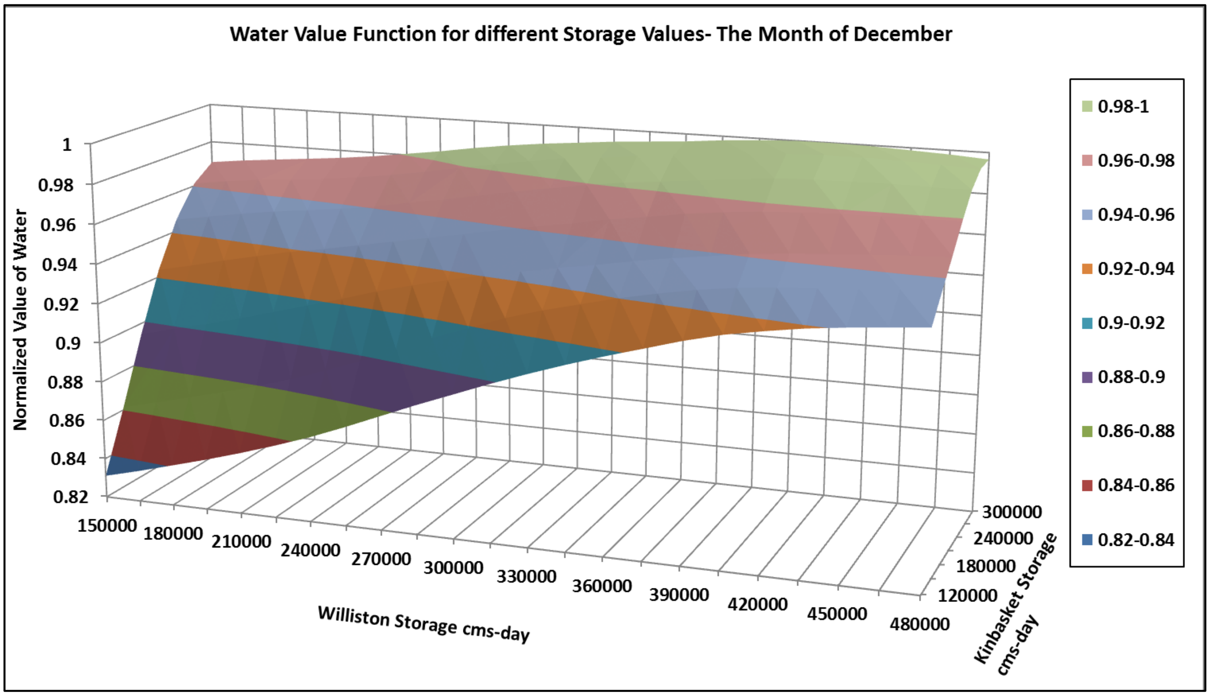


Figure 9: Three Dimensional Water Value for the Williston Reservoir and the Kinbasket Reservoir for the Month of December- CASE A

Figure 10 to **Figure 13** show the normalized value of water (left hand side) and the normalized marginal value of water (right hand side) in the Kinbasket Reservoir for different storage-states at the Williston Reservoir for CASE A. Each figure represents one of the selected months. Those months are selected to represent different stages of the water year and energy demand; Namely, October (a shoulder month), January (a winter month), May (a freshet month) and August (a summer month).

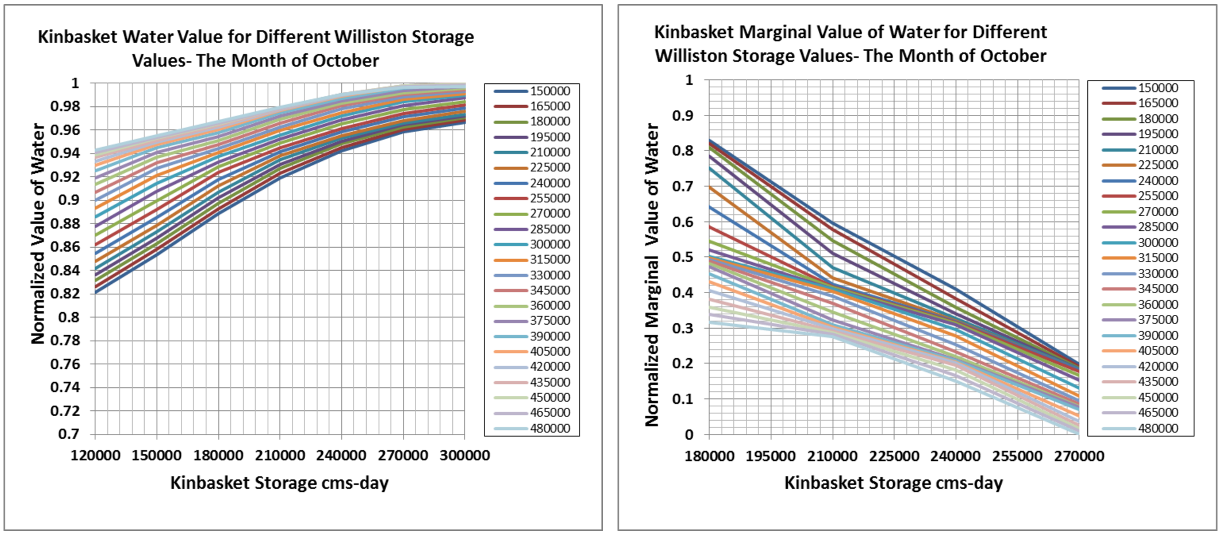


Figure 10: Water Value and Marginal Value of Water for the Kinbasket Reservoir for the Month of October- CASE A

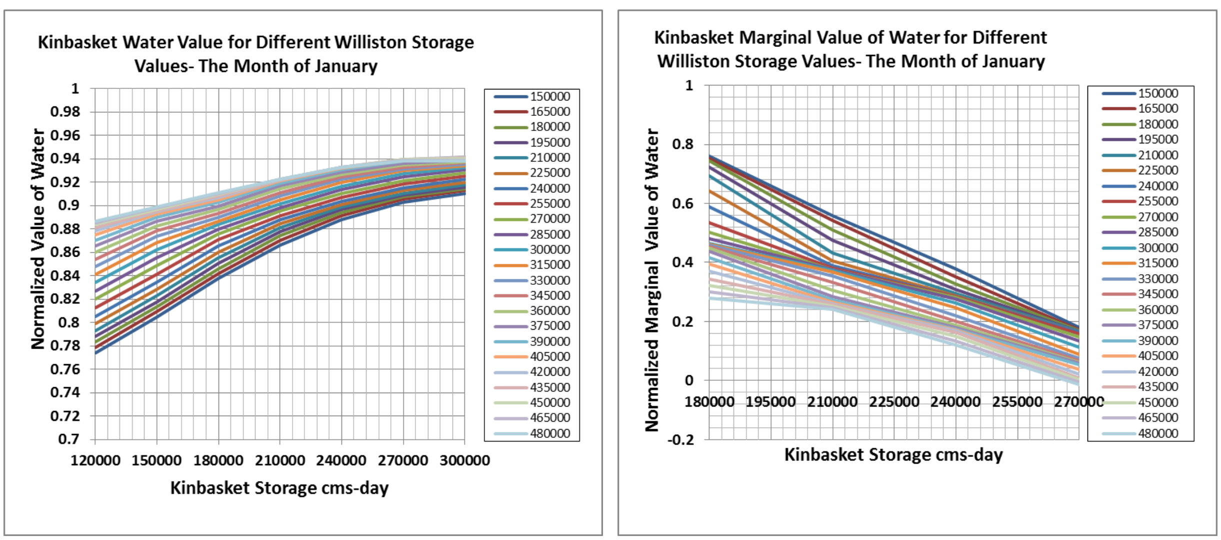


Figure 11: Water Value and Marginal Value of Water for the Kinbasket Reservoir for the Month of January- CASE A

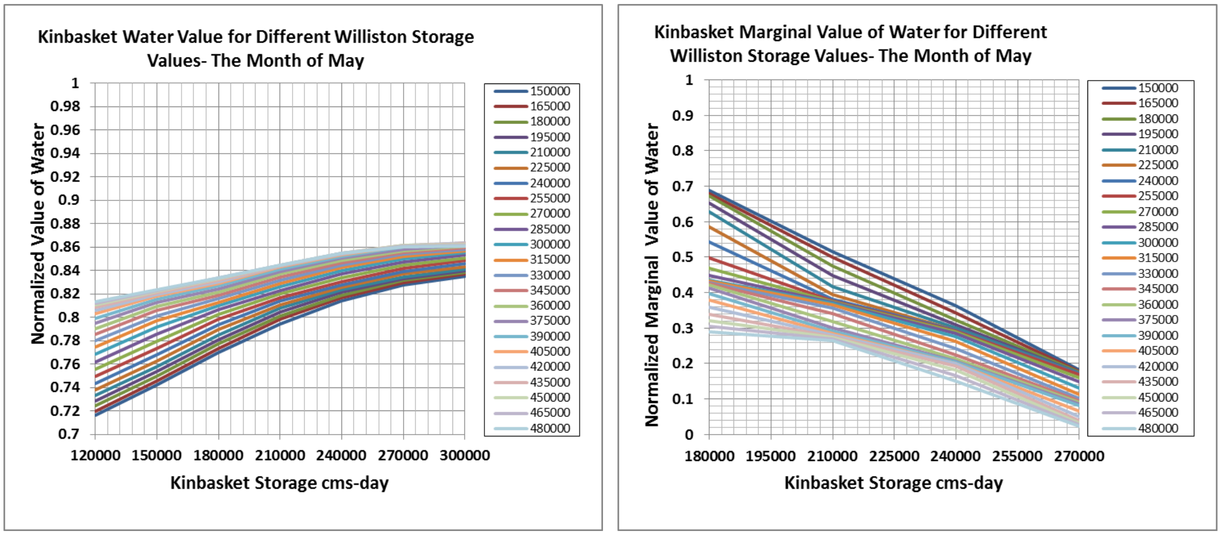


Figure 12: Water Value and Marginal Value of Water for the Kinbasket Reservoir for the Month of May- CASE A

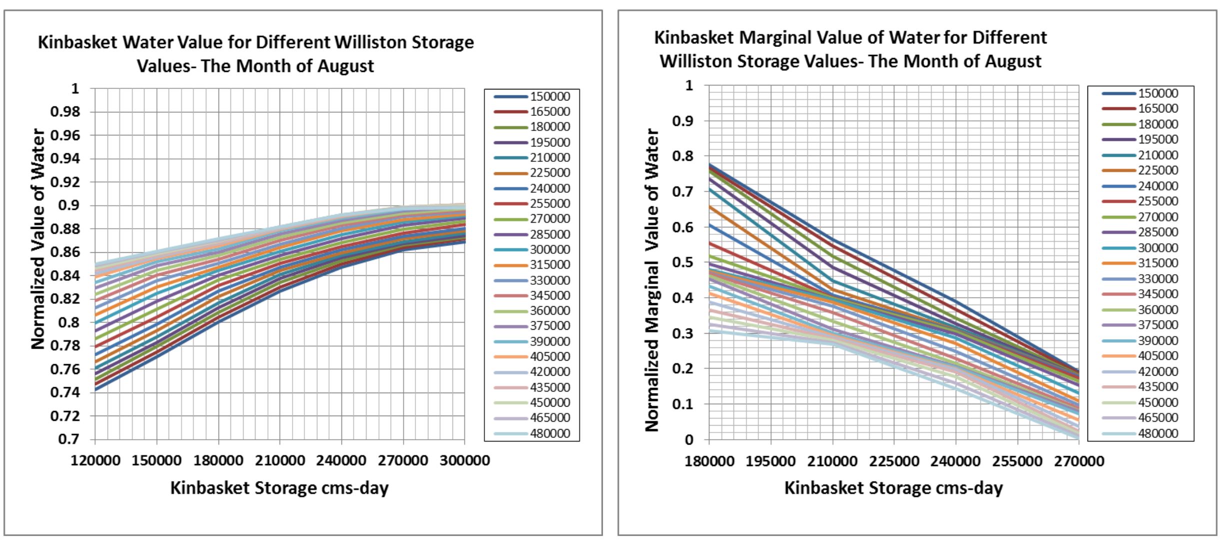


Figure 13: Water Value and Marginal Value of Water for the Kinbasket Reservoir for the Month of August- CASE A

Figure 14 to **Figure 17** show the normalized value of water (left hand side) and the normalized marginal value of water (right hand side) in the Williston Reservoir for different storage-states at the Kinbasket Reservoir for CASE A. The months selected for illustration are the same as discussed before.

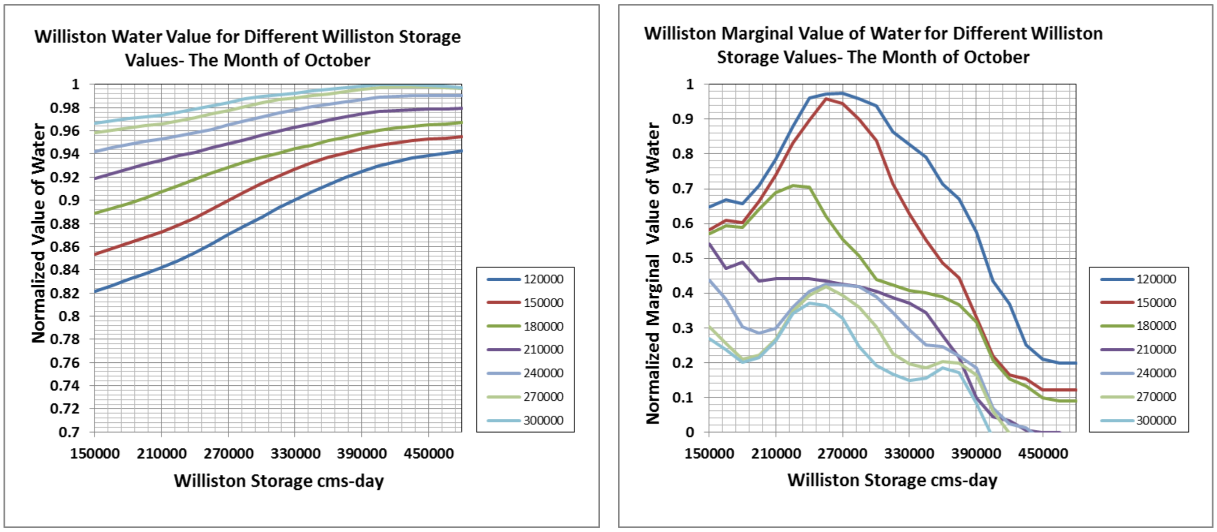


Figure 14: Water Value and Marginal Value of Water for the Williston Reservoir for the Month of October- CASE A

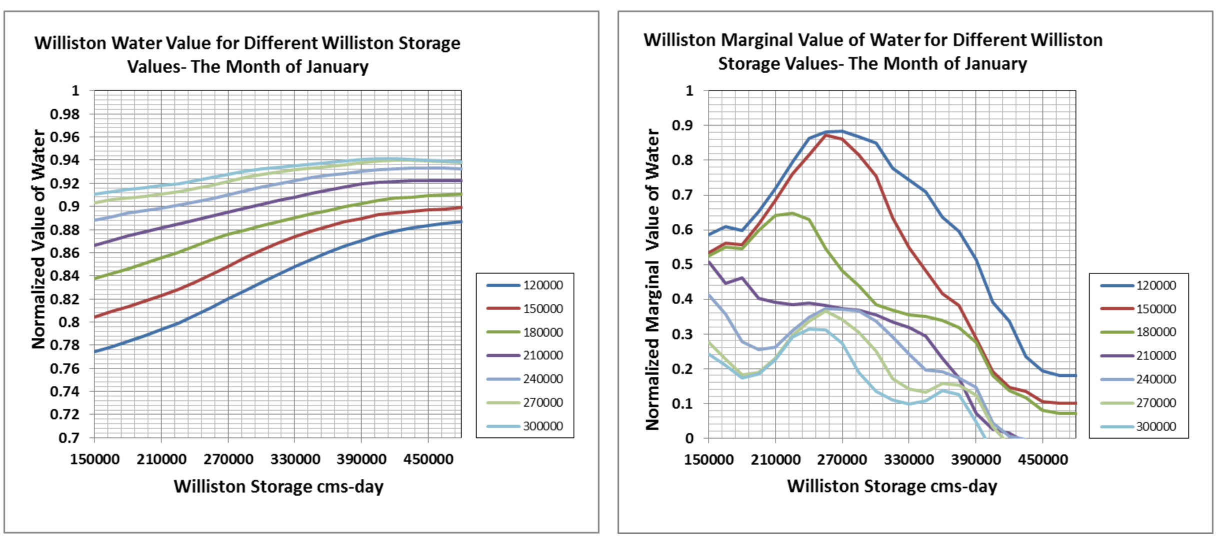


Figure 15: Water Value and Marginal Value of Water for the Williston Reservoir for the Month of January- CASE A

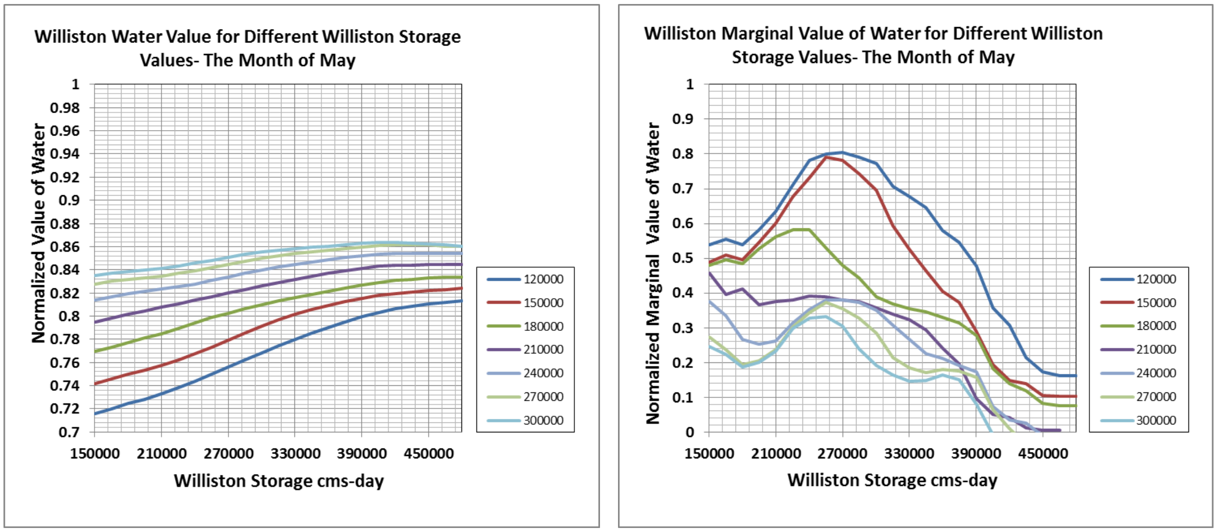


Figure 16: Water Value and Marginal Value of Water for the Williston Reservoir for the Month of May- CASE A

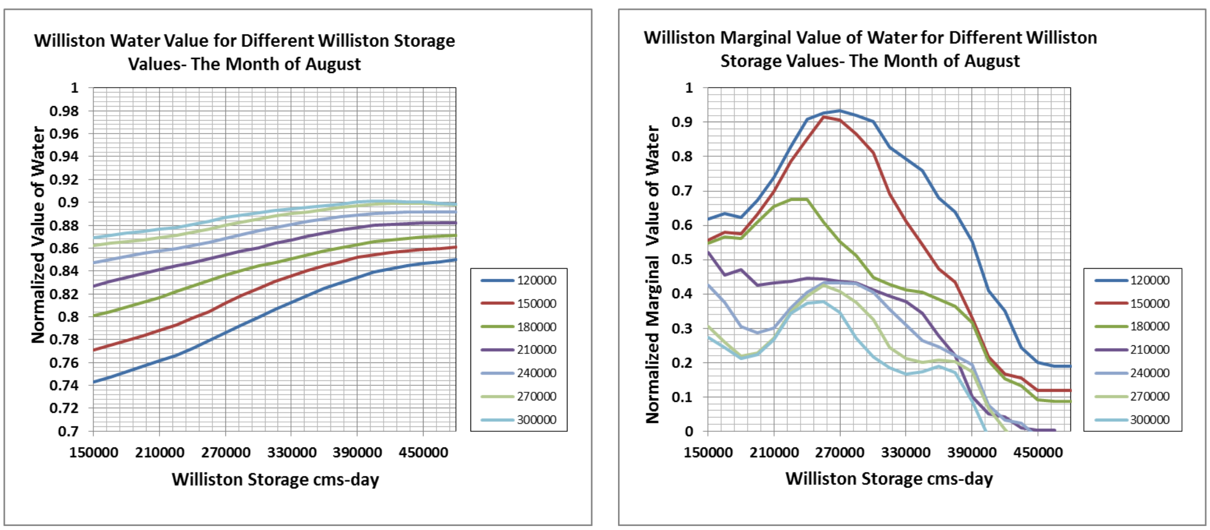


Figure 17: Water Value and Marginal Value of Water for the Williston Reservoir for the Month of August- CASE A

Figure 18 to **Figure 21** show the normalized value of water (left hand side) and the normalized marginal value of water (right hand side) in the Kinbasket Reservoir for different storage-states at the Williston Reservoir for CASE B. The months selected for illustration are the same as discussed before.

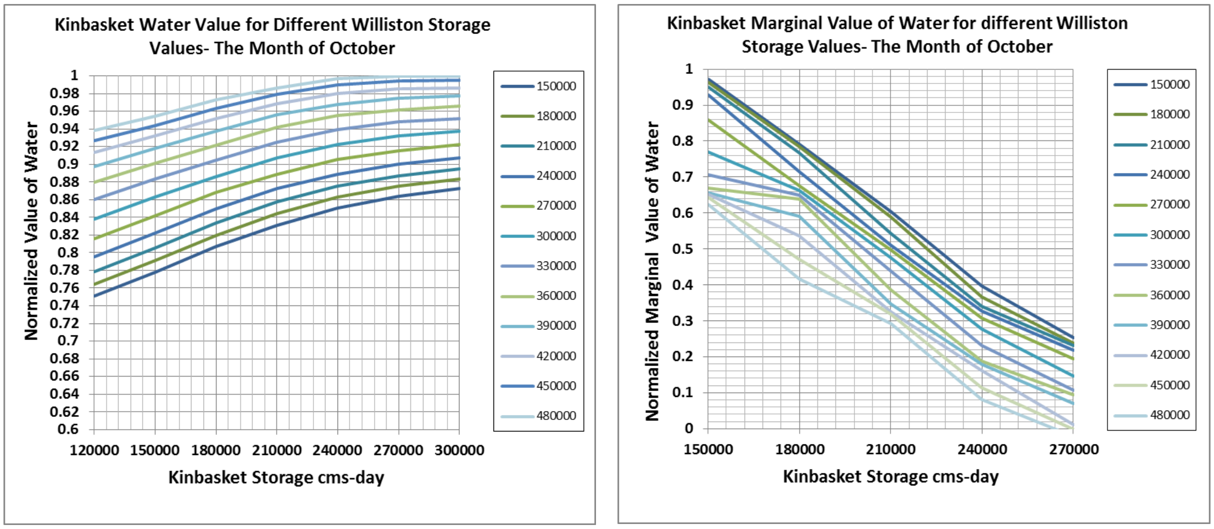


Figure 18: Water Value and Marginal Value of Water for the Kinbasket Reservoir for the Month of October- CASE B

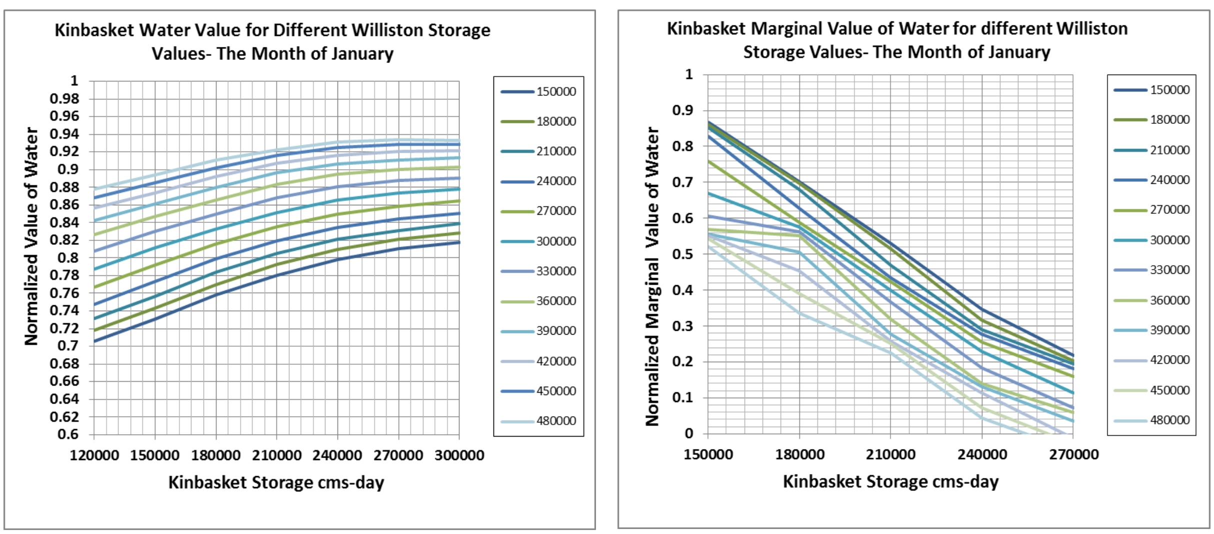


Figure 19: Water Value and Marginal Value of Water for the Kinbasket Reservoir for the Month of January- CASE B

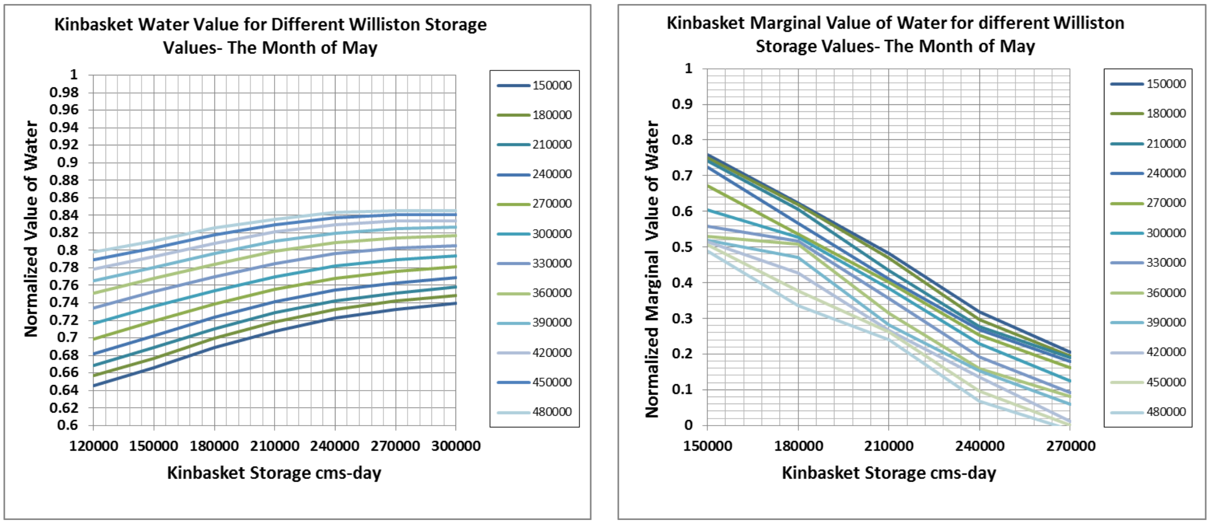


Figure 20: Water Value and Marginal Value of Water for the Kinbasket Reservoir for the Month of May- CASE B

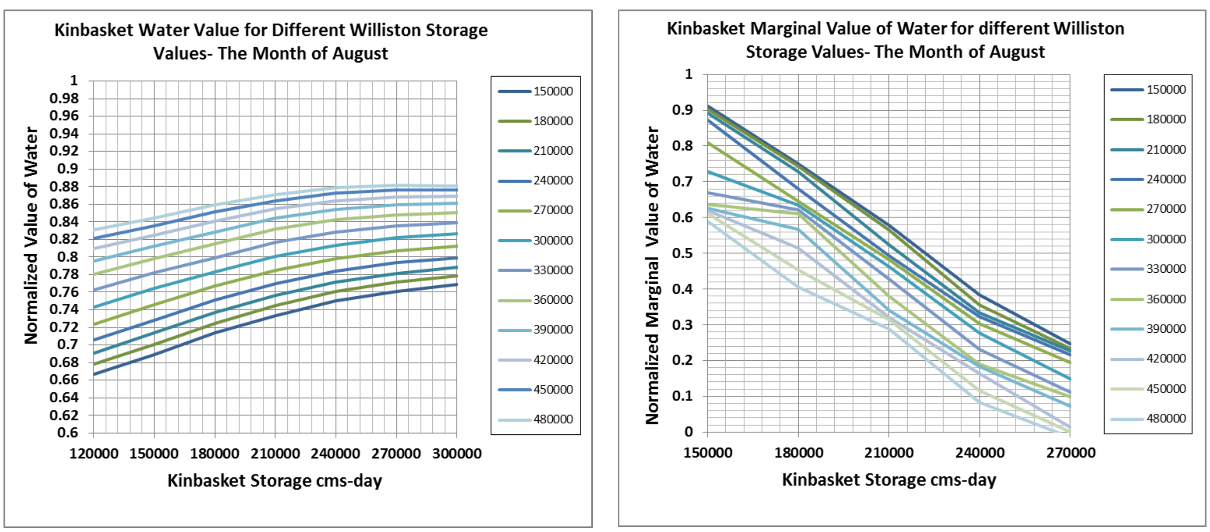
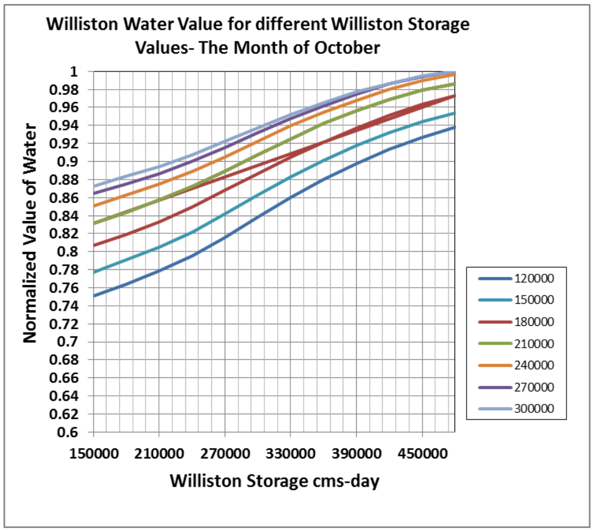


Figure 21: Water Value and Marginal Value of Water for the Kinbasket Reservoir for the Month of August- CASE B

Figure 22 to **Figure 25** show the normalized value of water (left hand side) and the normalized marginal value of water (right hand side) in the Williston Reservoir for different storage-states at the Kinbasket Reservoir for CASE B. The months selected for illustration are the same as discussed before.



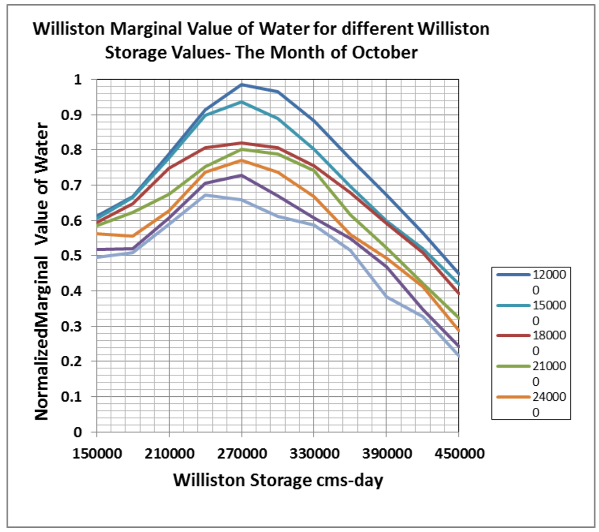
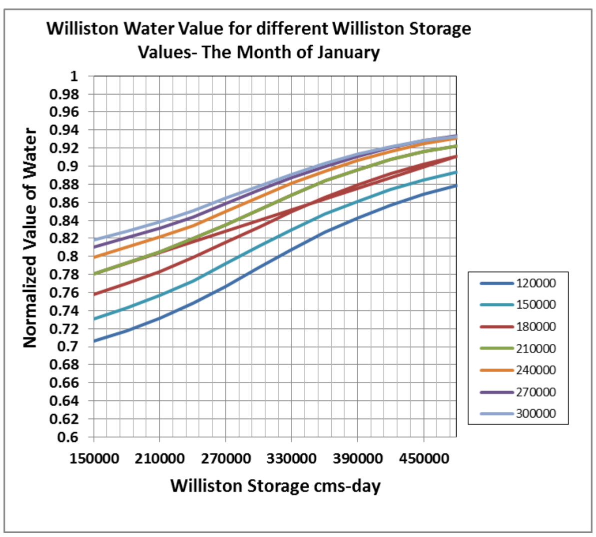


Figure 22: Water Value and Marginal Value of Water for the Williston Reservoir for the Month of October- CASE B



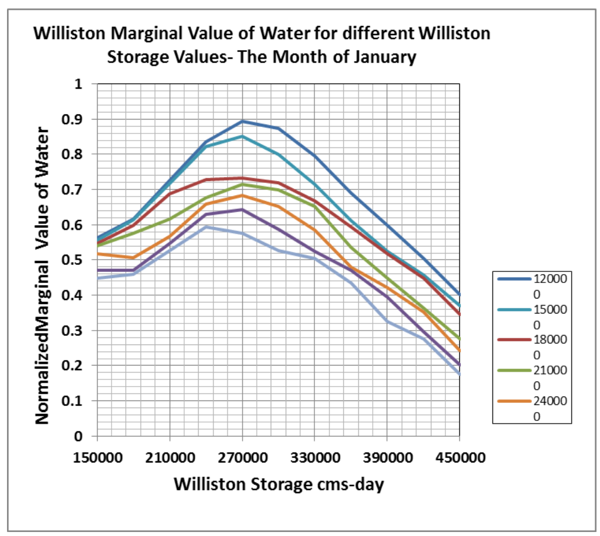


Figure 23: Water Value and Marginal Value of Water for the Williston Reservoir for the Month of January- CASE B

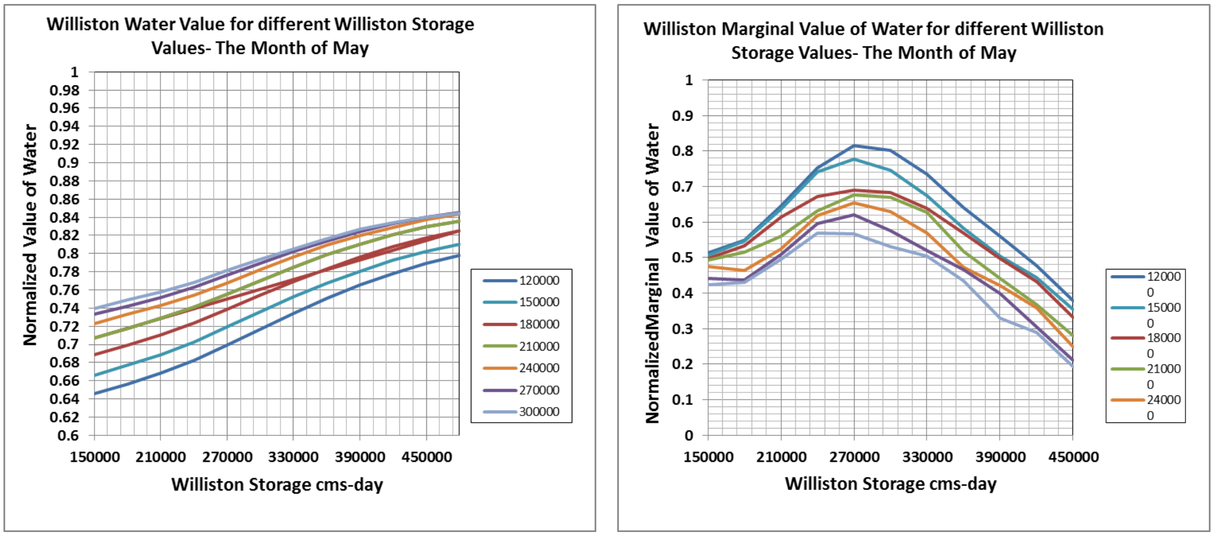


Figure 24: Water Value and Marginal Value of Water for the Williston Reservoir for the Month of May- CASE B

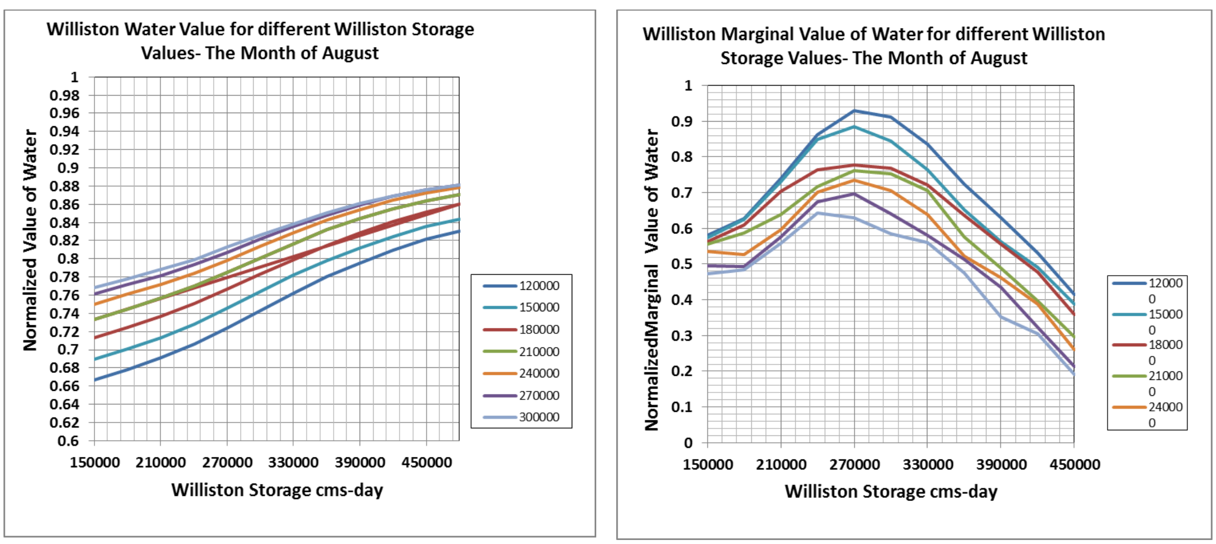


Figure 25: Water Value and Marginal Value of Water for the Williston Reservoir for the Month of August- CASE B

Figure 26 shows the variation in the monthly value of water in storage for selected storage combinations of both storage reservoirs over the water year for CASE A. Similarly, **Figure 27** shows these variations for CASE B. Comparing these two cases, it could be concluded that CASE A which has the finer discretization yields higher value of water on average as well as smoother change in the value of water from month to month.

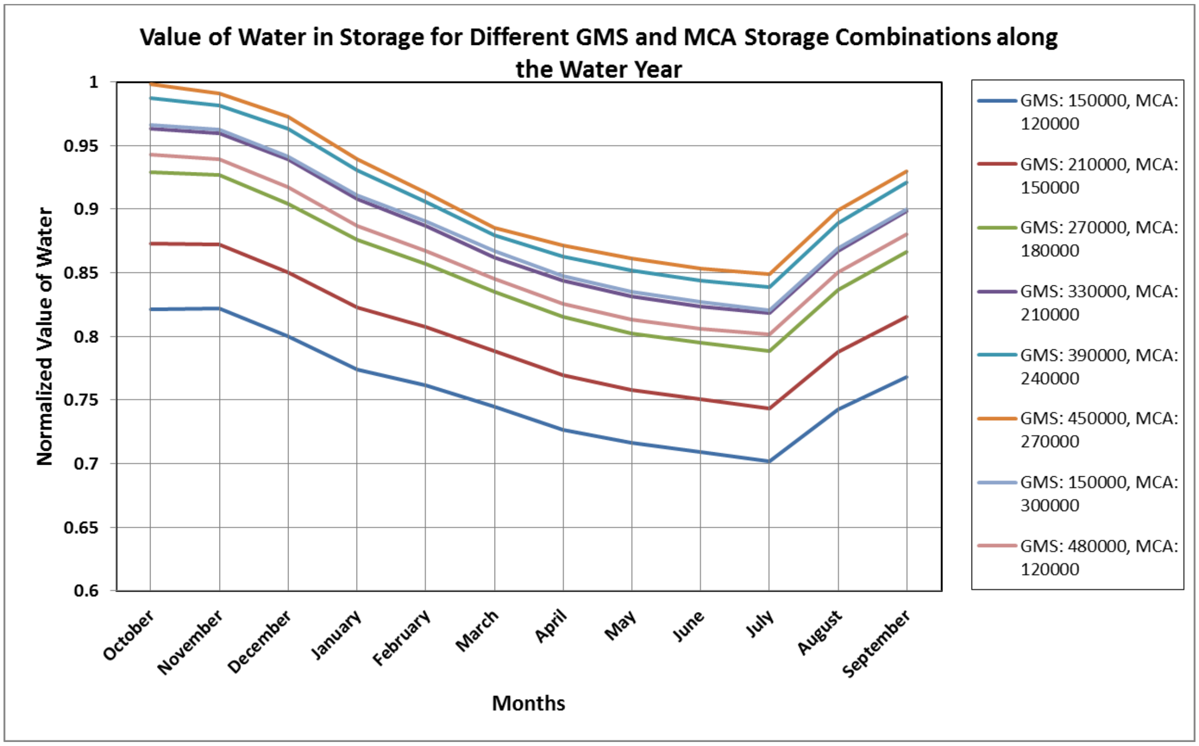


Figure 26: Value of Water in Storage of the Williston Reservoir and the Kinbasket Reservoir along the Water Year (October to September)- CASE A

On the other hand, comparing the marginal value of water in CASE A and CASE B, setting CASE A as the base case as illustrated in Figure **28**, shows that the marginal value of water in generally higher in CASE B which might be due to the coarser grid and hence the higher value of the derivatives of the value function. It could be also noticed that there is a trend to that change as it gets smaller in the freshet period and grow bigger towards the shoulder months.

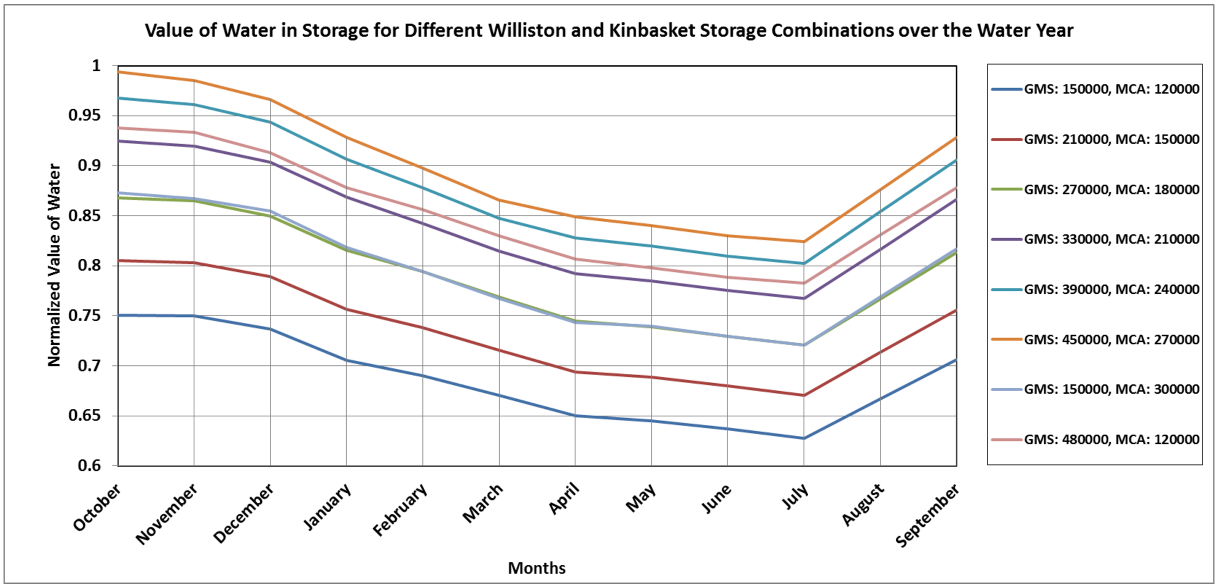


Figure 27: Value of Water in Storage of the Williston Reservoir and the Kinbasket Reservoir along the Water Year (October to September)- CASE B

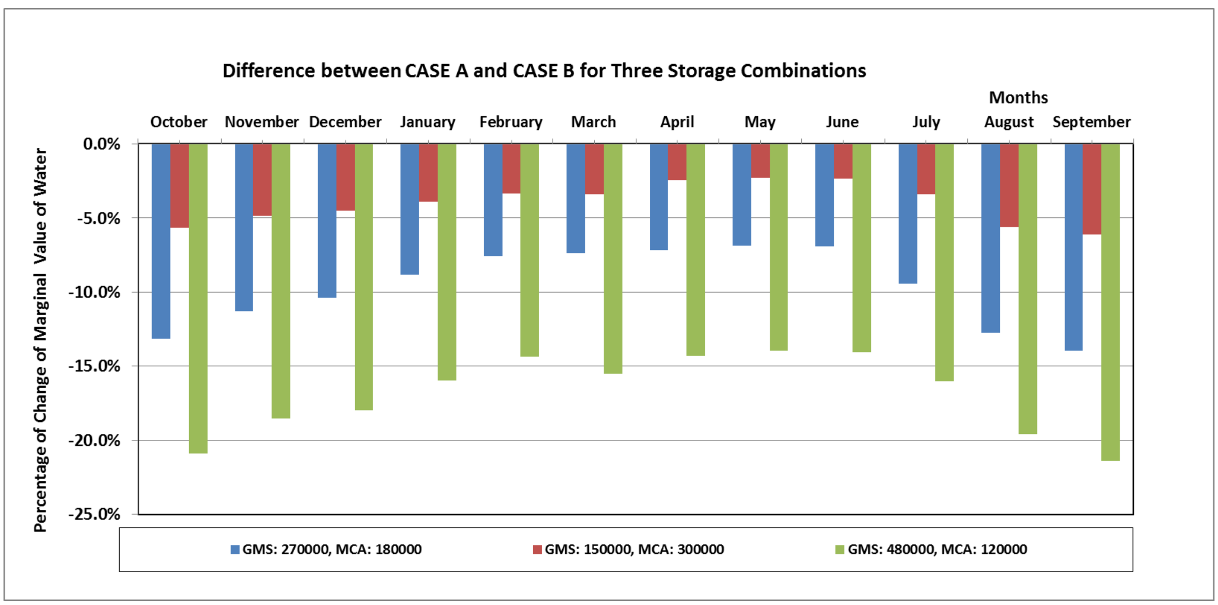


Figure 28: Percentage of Difference between the Marginal Value of Water in CASE A and CASE B for the Williston Reservoir and the Kinbasket Reservoir

4.3. Results of Intoducing the Storage State-space Corridor

One of the contributions of the algorithm used in the modeling is the use of the state-space corridor as opposed to using a fixed state-space for all the stages. This is a way to limit the state-space points visited and hence alleviating the computation effort. The details of this corridor are described in Chapter 5.

At an earlier stage of the model development, the fixed state-space was used until it was realized that a different approach is needed to alleviate the dimensionality problem to be able to solve a bigger and more complex problem. In this section, the results of using the corridor are compared to the results of using a fixed state-space. For brevity, only one case of state-space discretization is discussed which is CASE B in the previous section- Case of Storage State Increment of 1000 cms-d for both of the Williston Reservoir the Kinbasket Reservoir .These results are laid out in a graphical form.

Since the graphs of the value function and the marginal value of water for the CORRIDOR case have been illustrated in the previous section 4.2 (**Figure 18** to **Figure 25**), the graphs that are shown in the following figures (**Figure 29** to **Figure 36**) are only for the FIXED case. These graphs show that both of the value of water and the marginal value curves are sparser and more flat compared to the COORIDOR case where the same curves are more clustered. This suggests that using the CORRIDOR enabled the model to deduce a better operating policy and value of water for the different storages state-space combinations. Other advantages of the CORRIDOR are discussed in details in Chapter 5.

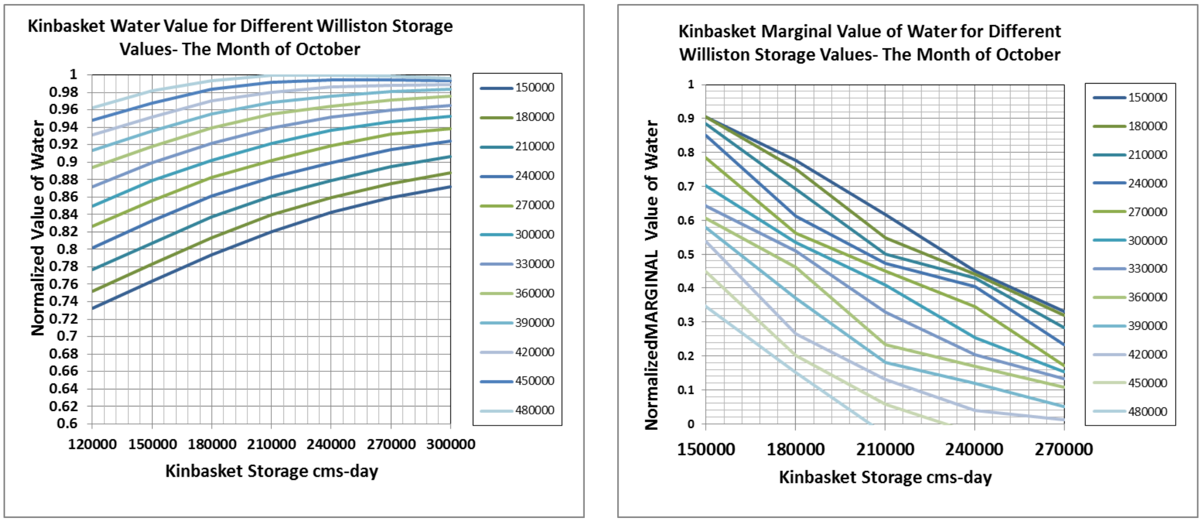


Figure 29: Water Value and Marginal Value of Water for the Kinbasket Reservoir for the Month of October- FIXED Case

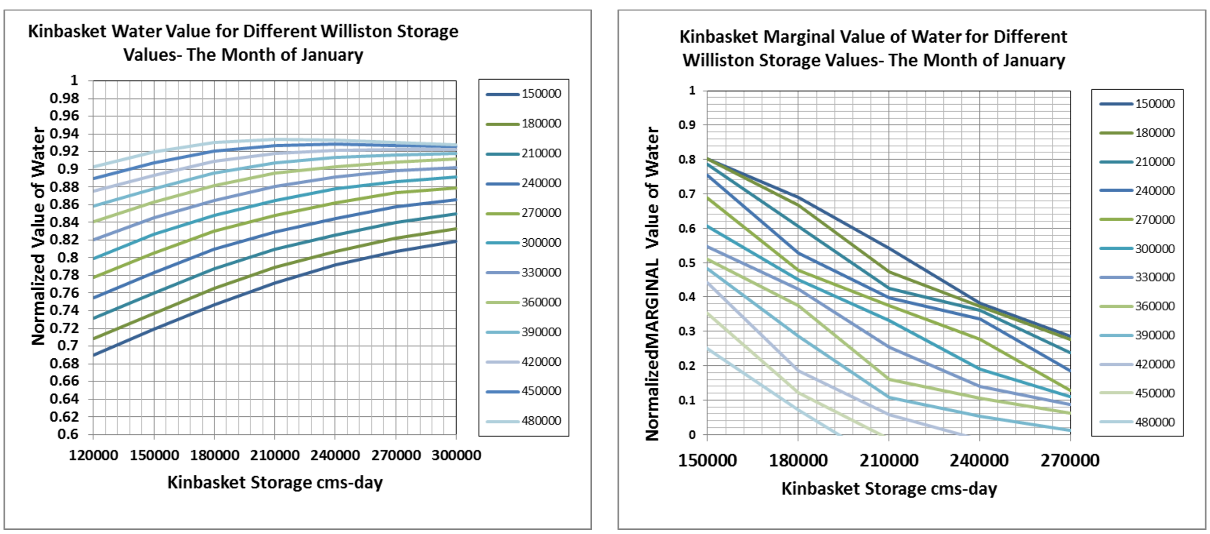


Figure 30: Water Value and Marginal Value of Water for the Kinbasket Reservoir for the Month of January- FIXED Case

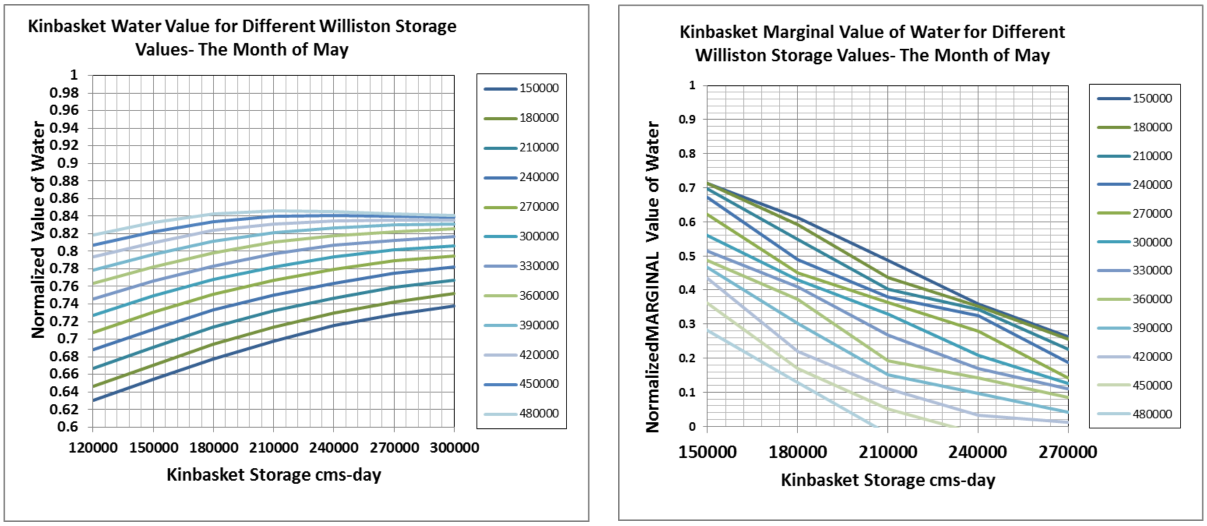


Figure 31: Water Value and Marginal Value of Water for the Kinbasket Reservoir for the Month of May- FIXED Case

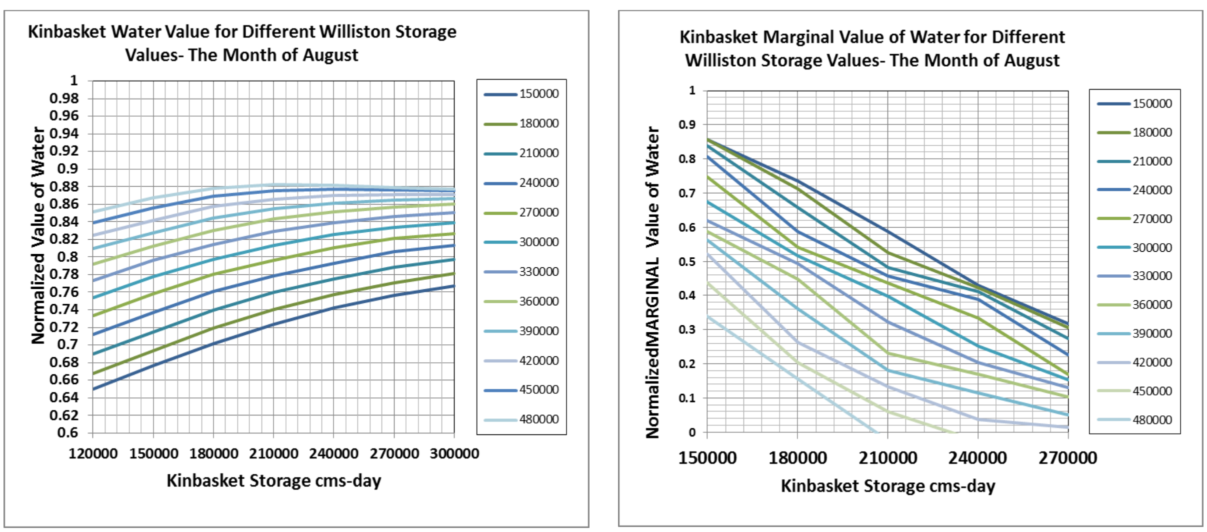


Figure 32: Water Value and Marginal Value of Water for the Kinbasket Reservoir for the Month of August- FIXED Case

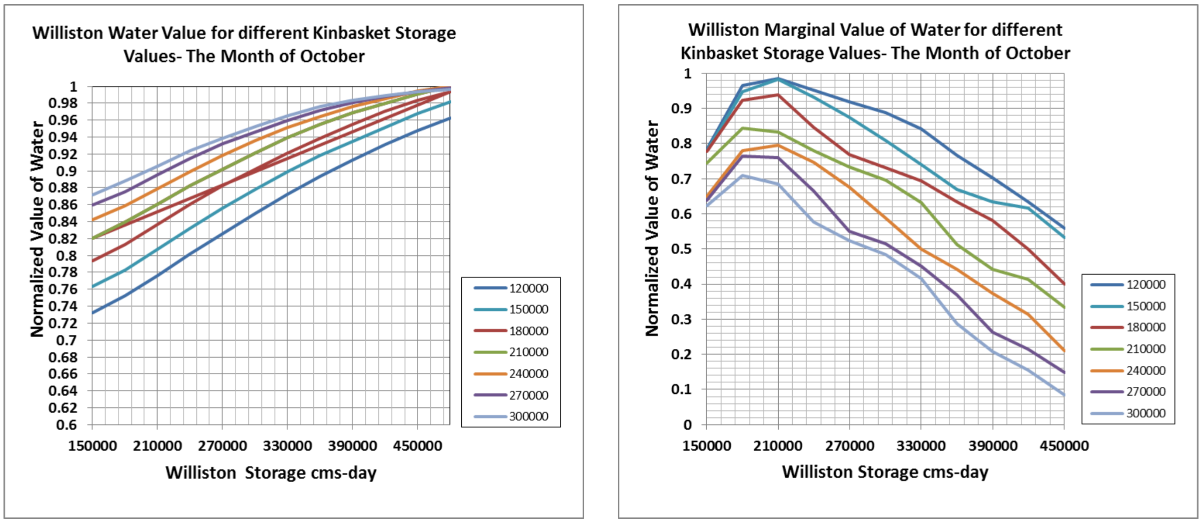


Figure 33: Water Value and Marginal Value of Water for the Williston Reservoir for the Month of October- FIXED Case

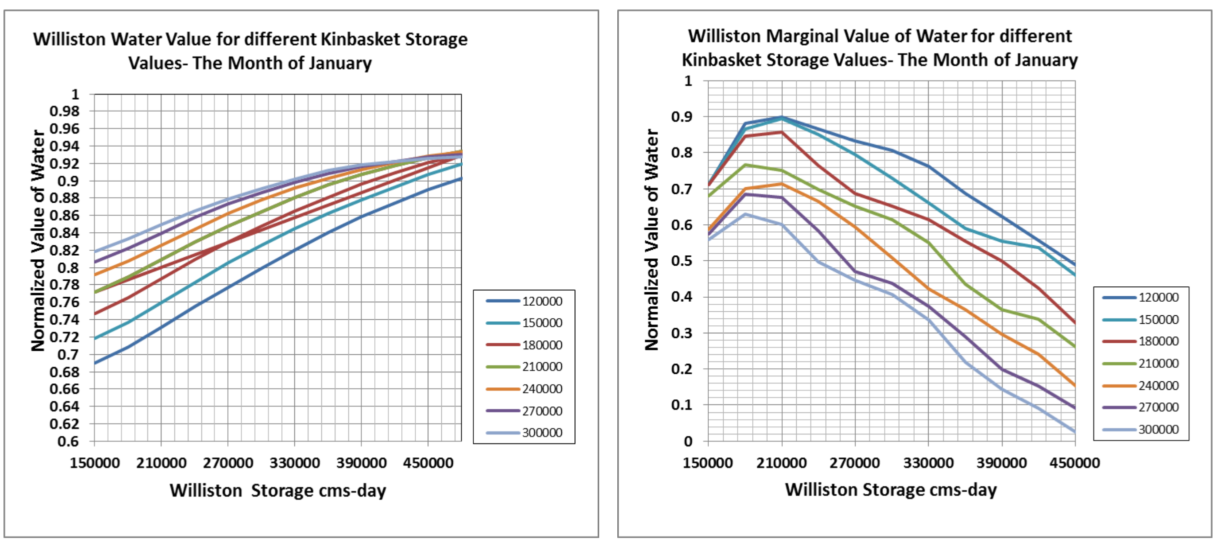


Figure 34: Water Value and Marginal Value of Water for the Williston Reservoir for the Month of January- FIXED Case

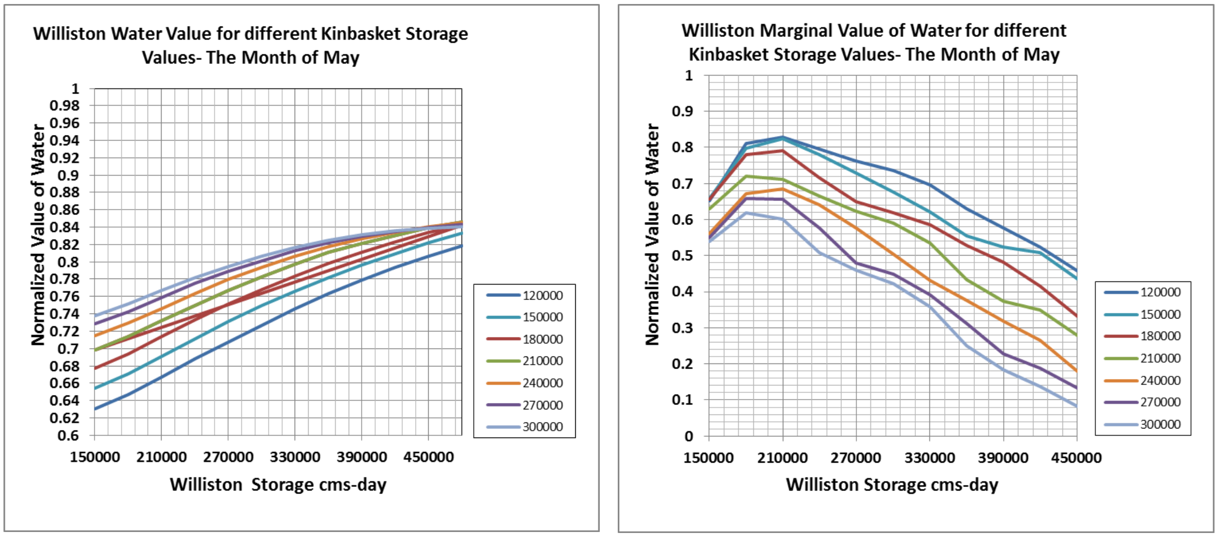


Figure 35: Water Value and Marginal Value of Water for the Williston Reservoir for the Month of May- FIXED Case

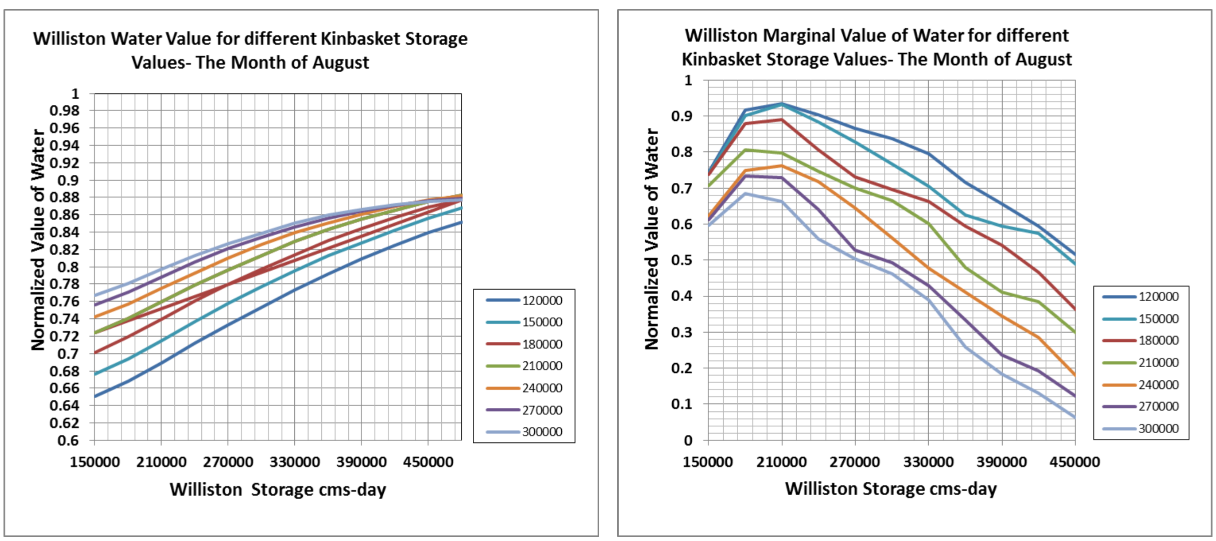


Figure 36: Water Value and Marginal Value of Water for the Williston Reservoir for the Month of August- FIXED Case

Chapter 5: Assessing the Impact of Storage State and Decision Space Discretization on Solution Efficiency and Precision of a Stochastic Dynamic Programming Algorithm in a Multireservoir Operations Planning Model

5.1. Introduction

The SDPOM6R model has gone through extensive testing. In this chapter, the results of sensitivity analyses, which were done as part of this extensive testing of the model, are discussed. The sensitivity analysis results discussed are for impact of introducing the state-space corridor, the impact of the state-space discretization and the sensitivity of results to the spill penalty function. The core material included in this chapter has been published in (Ayad, et al., 2013).

5.2. Assessing the Impact of Introducing a Storage State-space Corridor

Seventy three years of historical data were simulated to generate potential monthly storage states for the Williston and Kinbasket reservoirs. From this data, the upper and lower bounds of monthly storage levels were determined. **Figure 37** shows the storage corridor for the Williston Reservoir while **Figure 38** shows the storage corridor for Kinbasket Reservoir.

Storage buffers[15] are added to the storage bands shown in the figure above. The resulting data is then used to generate discretized storage states for both reservoirs for each time step using the *Discretizer* module as discussed earlier.

---

[15] Usually one state up and one state down at each stage; the exact value depends on the chosen increment.

It can be noted that both reservoirs are drafted during the winter period and then refilled during the freshet period. This drafting/refilling operation takes into consideration the prevailing hydrologic regime in these basins, the domestic electrical demand, and the seasonal trends in market prices.

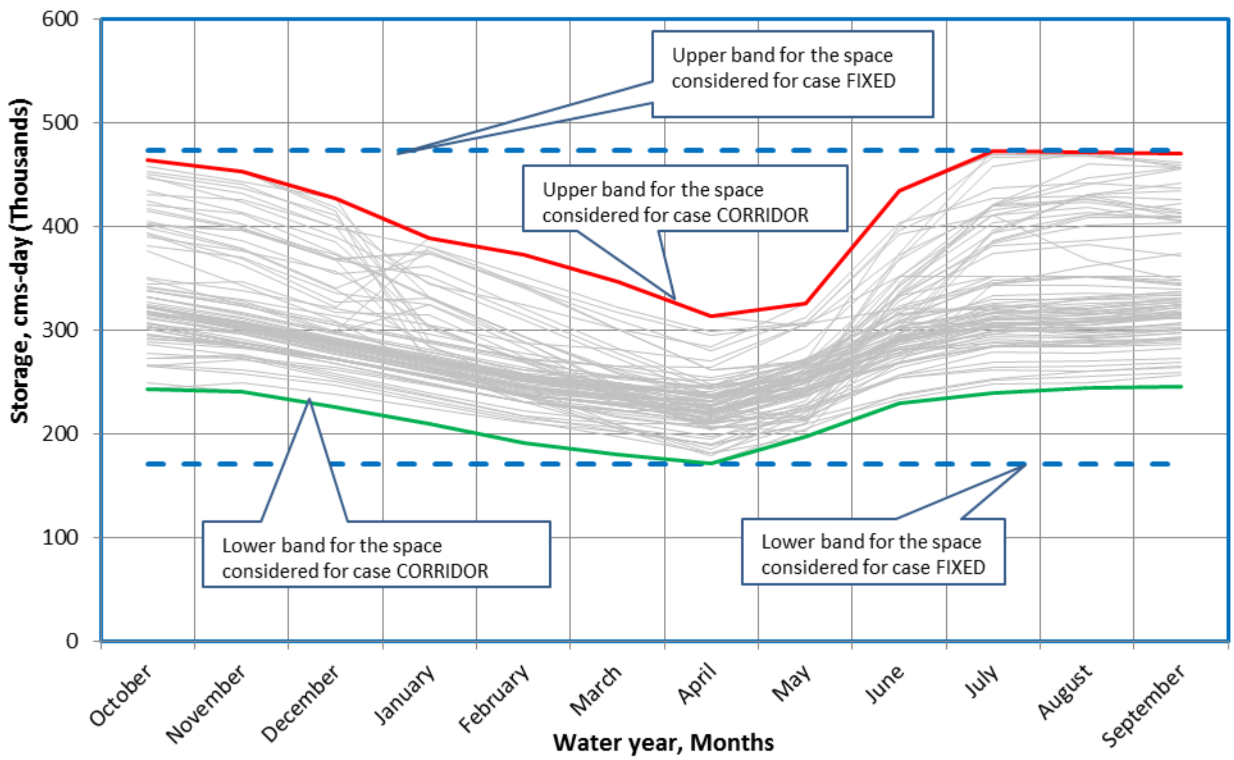


Figure 37: Simulated historical storage bands for the Williston Reservoir

Using this state-space corridor provides several advantages: the first is to have a realistic state-space for each time step which allows better representation of the real system and the second is to have a smaller problem size therefore reducing the required computational resources needed to solve the problem. By using the state-space corridor, it is also possible to extend the planning horizon, to run the model with finer space discretization, and to add more state variables in future implementations. However, care must be taken to ensure that the selected state-space corridor does not cause the algorithm to choose sub-optimal solutions or result in infeasibilities for some state transitions, particularly near the upper and lower storage limits.

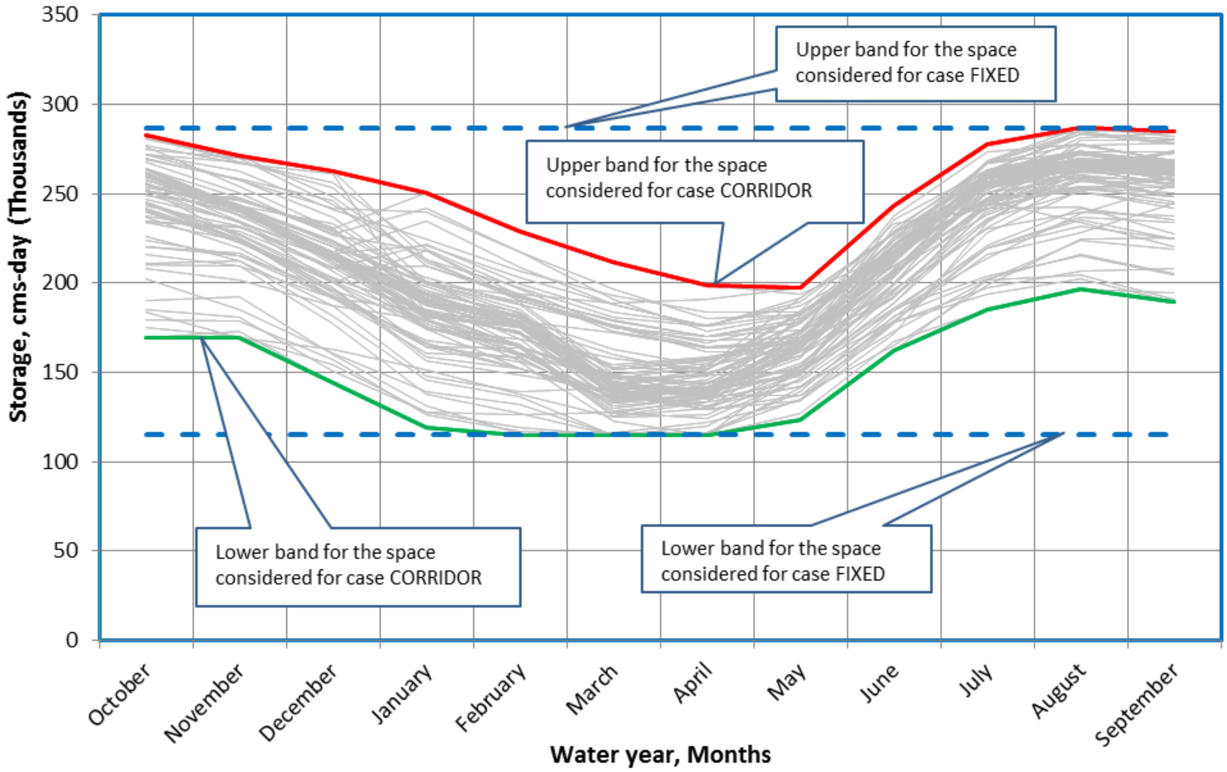


Figure 38: Simulated historical storage bands for the Kinbasket Reservoir

A comparison between running the model for a fixed state-space[16] (FIXED case) versus using the storage corridor (CORRIDOR case) was performed to assess the impact of using the corridor on the problem solution efficiency and precision of the output. The storage states-space was discretized at 1000 cms-d for both reservoirs. The reason for using this increment was that it was not possible to run the FIXED case for smaller increments because

[16] By using only the physical maximum and minimum storage values at all stages in the entire horizon.

of the problem's dimensions. The following results compare runs for 12, 24 and 36 time steps[17].

Comparing the two cases, it was found that it took on average about double the time to run FIXED cases as compared to CORRIDOR cases and that the trend of the increase in time was linear for both cases as illustrated in **Figure 39**.

Setting the FIXED CASE As the base case, and for a 12 time-step run, the average difference in the value of water in storage was -1.12% between the two cases, while the average difference in the marginal value of energy was 8.96% and -5.81% for Williston and Kinbasket respectively.

Repeating the same analysis for the 24 time steps run, the corresponding differences were -1.04%, 10.36%, and -3.42%. For the 36 time-step trial, the corresponding differences were -1.04%, 10.87% and -0.12%. It should be noted that the differences in the marginal values of energy for Williston Reservoir were

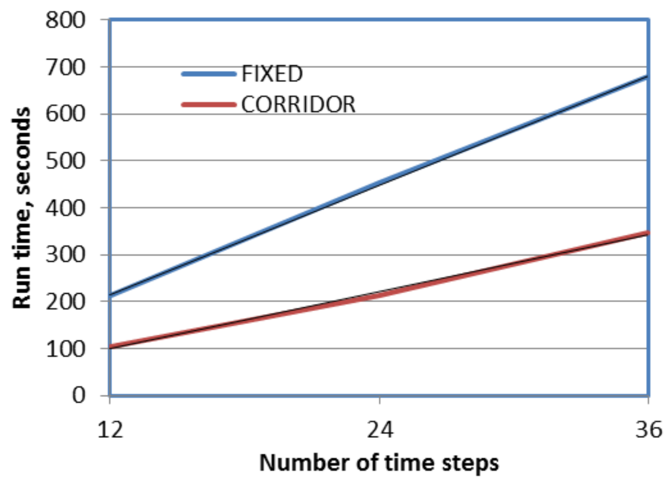


Figure 39: Run Times for FIXED Case vs. CORRIDOR Case

[17] All the runs in this manuscript are done on a server with 48 GB of RAM.

higher than those for the Kinbasket Reservoir. This is due to the larger operation range of the Williston as compared to Kinbasket (approximately 1.65 larger).

As a result of this work, it is apparent that using the storage state-space corridor can significantly impact the accuracy of the results, but care must be taken in defining the corridor and specifying the state discretization increments for such problems.

5.3.Assessing the Impact of State-space Discretization

The solution efficiency of the problem as well as the precision of the output is impacted by the state-space discretization in the SDP technique. The smaller the space increments, the more precise the output, but that comes at the cost of increasing the dimensions of the problem and hence may jeopardize the solution efficiency of the problem. A trade-off between the accuracy of the output and the solution efficiency of the problem is tested in this section.

Table 2: Storage State Discretization Cases

Case	**Storage state increment for the Williston Reservoir, cms-d**	**Storage state increment for the Kinbasket Reservoir, cms-d**	**Problem size per time step, million**
1	500	1000	**23.33**
2	750	750	13.12
3	750	1000	6.62
4	1000	500	14.02
5	1000	750	5.04
6	1000	1000	**2.54**

A group of six cases were investigated, all for a planning horizon of 24 months and all using the selected storage state-space corridor discussed above. The discretization parameter tested is the increment of the storage states, which is the same as the increment of the inflow discrete distribution and the increment of the discretized decision space. The different cases and the corresponding problem size are shown in **Table 2**. Comparing the run time for each case showed an exponentially increasing trend - i.e. as the problem size increases the run time increases exponentially as shown in **Figure 40**. Case 1, with the largest problem size, was taken as the base case to compare the other cases to. It should be noted that cases with larger problem sizes were tested (up to ~164 million states per time step) but they are not analyzed here.

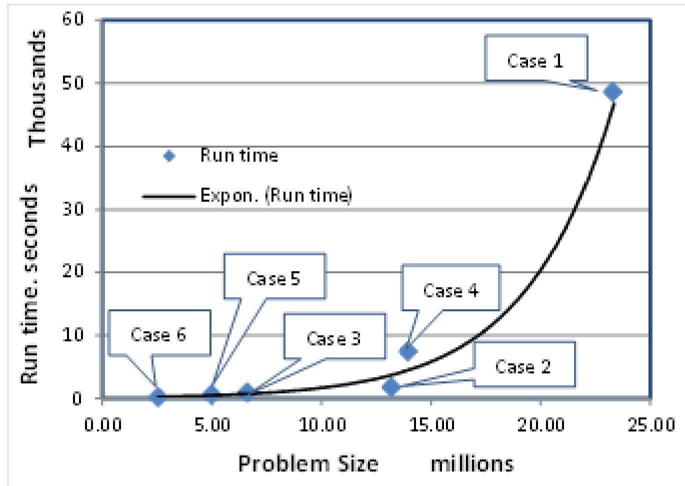


Figure 40: Problem Size vs. Run Time for Different Cases

It was noted that refining the discretization of the problem space remarkably enhances the smoothness, curvature, and shapes of both of the value function and the marginal value of energy function.

Given the above results, it can be concluded that finer discretization of the state-space will yield more accurate estimates of the value of water in storage and marginal value of water functions.

5.4.Sensitivity of Results to Spill Penalty Values

As discussed earlier, a penalty in the form of an import price multiplier is used to estimate the cost of spill. Eight cases were tested for the same storage state-space corridor for 24 stages at an increment of 1000 cms.d for both storage reservoirs to investigate the sensitivity of the model to the penalty used. **Table 3** shows the cases tested. Case 1 was taken as the base case to compare the other cases to.

Table 3: Different Cases for the Penalty Function

Case	Penalty Multiplier
1	0
2	0.20
3	0.5
4	1
5	1.50
6	2.00
7	10.00
8	50.00

In general, the higher the penalty the lower the value of water in storage is. At the higher storage states the effect became more pronounced because, at those states, the penalty had more effect as the reservoir was more likely to spill in those states. It was also noticed that the value of water in storage was 85% lower than the base case, on average, when the penalty value was set to very high values (cases 7 and 8).

On the other hand the marginal value of energy for both reservoirs increased for cases 2 through 6 with the highest increase for case 5, which corresponds to a penalty multiplier of 1.5. Because the penalty is very high for cases 7 and 8, the marginal value of energy decreased for both reservoirs for these cases.

One important note is that, if high and low storage states are excluded, the ratio of the marginal value of energy in Kinbasket to the marginal value of energy of Williston lies within reasonable limits for Cases 4, 5, and 6 [18] while the ratio is significantly higher for other cases, particularly Cases 1, 2, and 3.

It could be concluded, based on the tested cases, that using a penalty multiplier of 1 to 2 is the best option to obtain reasonable[19] marginal values of water of storage as compared to historically observed results.

---

[18] Reasonable limits are assumed to be the ratio of total HK of the Columbia River system to the total HK of the Peace River system, which is equal to ~1.23 on average.

[19] Comparing them to the actual market prices and looking into actual trade schedules for the spot market and other historically observed results.

Chapter 6: Conclusions, Recommendations and Future Work

6.1. Conclusions

Extensive testing of the model developed as part of the current research has shown that it is able to solve the problem for up to 36 monthly time steps (3 years) producing practical water value and marginal value functions up to a problem size of ~ 164 million states per time step.

It is expected that, with further enhancements of the algorithm, the model could handle a much larger problem and could easily be extended to include more state variables. The development of a dynamic storage-state corridor using the simulated historical data significantly accelerated the convergence of the algorithm and allowed the solution of larger problems but care must be taken to ensure that the derived solutions are robust and globally optimal.

Flexibility is one of the most important features of the current model. It is well known that SDP solution algorithms are not typically very flexible as they are custom-built for specific reservoirs and systems. The flexibility of the SDPOM6R model is a result of enhancements made to the algorithm and the formulation of the model coding in AMPL, which provided the following advantages:

1. The increment of all the state spaces and the storage corridor can be controlled and easily changed for each reservoir for each time step;

2. The problem can be run for one storage reservoir while fixing the states of the other storage reservoir[20];
3. The modeling algorithm is generalized so that the user can easily adapt it and change model mode simply by changing few control parameters; and
4. The model can be run either directly on a personal computer (or a server) using the command prompt or by using a scheduling program that uses simple scripts to optimize the use and sharing of computing assets.

Because of the nature of the SDP technique, as well as the complexity of the system investigated herein, there are several limitations of the algorithm. Some of these limitations might be overcome while others are likely to persist in future versions of the model or adaptation of the same technique. Some of these limitations are: the stochasticity of load and prices is not currently represented, the accuracy of the existing regression equations representing the prices and other variables could be improved, and the model only contains variables representing six major plants in the BC Hydro system while fixing the output of other resources and therefore it simplifies the real system.

6.2. Proposed Future Enhancements to the Model Developed

As discussed in Chapter 3 and Chapter 5, there are several features of the model that need to be enhanced along with some features that could be added in order to get a more robust and representative model and obtain more practical functions of the water value and

---

[20] This option allows the model to solve for one storage reservoir given a fixed state of the other reservoir.

the marginal value of water. Several functions and features need to be added to the SDPOM6R such as:

1. To split the state variable for Kinbasket into two state variables: Kinbasket Reservoir storage and non-treaty storage as well as to represent the Kinbasket and Arrow Lakes flex storage accounts to better represent the Columbia River Treaty operation of the Columbia River system;
2. To include new storage limits derived from flood control curves from the Columbia River Treaty operating plans, and
3. To introduce sub-time step functionality such as: peak load hours (PLH), heavy load hours (HLH), and light load hours (LLH) in order to better capture the electricity market depth and price variability.

The following list presents the features that are included in the current modeling but need to be better represented, enhanced or modified:

1. Representation of prices to replace the currently used regression equations;
2. Representation of load stochasticity;
3. Representation of inflows;
4. Representation of the transmission limits need to be revised and enhanced; and
5. HK calculation procedure could be enhanced.

6.3. Future Work and Recommendations

An extensive literature survey has been conducted, but not included in the current manuscript, on both reinforcement learning and the multiagent reinforcement learning techniques. These two techniques are closely related and are based on the SDP technique that was used to develop the SDPOM6R Model.

Below is a list of the work which has been conducted by the author of this manuscript, but not included in it:

1. Enhancement and testing work that was performed on the models developed by (Abdalla, 2007; Shabani, 2009);
2. A framework for the use of Multi-agent Reinforcement Learning (MARL) technique.

Based on both of the literature survey and the modeling work mentioned above, the following is recommended:

1. Explore the possibility of developing a full working version of MARL model that better represent the system and its inherent stochasticity and uncertainty and ensure the proper coordination between components of the system through the artificial intelligence and the MARL approach adopted in this model;
2. Benchmark the model above against the SDPOM6R model as well as the other models developed as part of the *Water Value Project*.

Bibliography

Abdalla, A. E., 2007. *A Reinforcement Learning Algorithm for Operations Planning of a Hydroelectric Power Multireservoir System.* s.l.:Ph.D. Thesis, Department of Civil Engineering, University of British Columbia.

Abdalla, A. et al., 2013. *Water Value Project Definiton Phase,* Burnaby, BC: BC Hydro.

Archibald, T. W., Mckinnon, K. I. M. & Thomas, L. C., 1997. An Aggregate Stochastic Dynamic Programming Model of Multiple Reservoir Systems. *Water Resources Research,* 33(2), pp. 333-340.

Arvanitidis, N. & Rosing, J., 1970. Optimal Operation of Multireservoir Systems Using a Composite Representation. *IEEE Transactions on Power Apparatus and Systems,* PAS-89(2), pp. 327-335.

Ayad, A., Shawwash, Z. & Abdalla, A., 2012. *Documentation of the BC Hydro Stochastic Dynamic Programming Model (SDPOM6R).Version 2,* s.l.: BC Hydro, British Columbia, Canada.

Ayad, A. et al., 2013. *Assessing the Impact of Storage State-Space and Decision Space Discretization on the Solving Efficiency and Precision of a Stochastic Dynamic Programming Algorithm in a Multireservoir Operations Planning Model.* s.l., The Proceedings of the HydroVision International Conference, Denver, CO, USA, July 2013.

BC Hydro, 2000. *Making the Connection: the BC Hydro Electric System and How It is Operated,* Vancouver, British Columbia: Canadian Cataloguing in Publishing Data.

BC Hydro, 2013. *BC Hydro System: Quick Facts.* [Online]

Available at: https://www.bchydro.com/content/dam/BCHydro/customer-portal/documents/corporate/accountability-reports/financial-reports/annual-reports/quick-facts-for-year-ended-march-31-2013.pdf

[Accessed 13 November 2017].

BC Hydro, 2017. *BC Hydro System: Quick Facts.* [Online]

Available at: https://www.bchydro.com/content/dam/BCHydro/customer-portal/documents/corporate/accountability-reports/financial-reports/annual-reports/bchydro-quick-facts-june-2017.pdf

[Accessed 13 November 2017].

BC Transmission Corporation, 2010. *British Columbia Budget.* [Online]

Available at: www.bcbudget.gov.bc.ca/2010/sp/pdf/agency/bctc.pdf

[Accessed 19 October 2017].

Bellman, R., 1957. *Dynamic Programming.* Princeton(New Jersey): Princeton University Press.

Bertsekas, D., 1995. Neuro-Dynamic Programming: An Overview. , *Proceedings of the 34th Conference on Decision & Control New Orleans, Lon Angeles, USA.*

Blair, T., 2017. *Documentation: Application of SSDP within an International Treaty (MUReO),* s.l.: s.n.

Christensen, G. S. & Soliman, S. A., 1989. Long-term Optimal Operation of Series Parallel Reservoirs for Critical Period with Specified Monthly Generation and Average Monthly Storage. *Journal of Optimization Theory and Applications*, 63(3), pp. 333-353.

Dias, B. et al., 2010. Stochastic Dynamic Programming Applied to Hydrothermal Power Systems Operation Planning Based on the Convex Hull Algorithm. *Mathematical Problems in Engineering,* Volume 2010, pp. 1-20.

Druce, D., 1989. Decision Support for Short Term Export Sales From a Hydroelectric System. *In Computerized Decision Support Systems for Water Managers, J. W. Labadie, L.E. Brazil, I. Corbue, and L.E. Johanson, Eds., ASCE* , pp. 490-497.

Druce, D., 1990. Incorporating Daily Flood Control Objectives Into a Monthly Stochastic Dynamic Programing Model for a Hydroelectric Complex. *Water Resources Research,* 26(1), pp. 5-11.

Duranyildiz, I. & Bayazit, M., 1988. Optimal Operation of Reservoir Systems in Critical Periods. *Water Resources Management,* 2(2), pp. 141-148.

Faber, B. & Stedinger, J., 2001. Reservoir Optimization Using Sampling SDP with Ensemble Streamflow Prediction (ESP) Forecasts. *Journal of Hydrology,* 249(1-4).

Fane, L., 2003. *Generalized Optimization in the British Columbia Hydroelectric System.* s.l.:MA.Sc. Thesis, Department of Civil Engineering, University of British Columbia..

Guan, Z., Shawwash, Z. & Abdalla, A., 2017. Using SDDP to Develop Water-Value Functions for a Multireservoir System with International Treaties. *to appear in ASCE Journal of Water Resources Planning and Management.*

Hall, W. A., Askew, A. J. & Yeh, W. W.-G., 1969. Use of the Critical Period in Reservoir Analysis. *Water Resources Research,* 5(6), pp. 1205-1215.

Howson, H. R. & Sancho, N. G. F., 1975. A New Algorithm for the Solution of Multistate Dynamic Programming Problems. *Mathematical Programming,* Volume 8, pp. 104-116.

Jacobson, D. H. & Mayne, D., 1970. *Differential dynamic programming.* s.l.:New York, NY : North-Holland, 1970. - 216 p..

Kelman, J., Stedinger, J. & Cooper, L., 1990. Sampling Stochastic Dynamic Programming Applied to Reservoir Operation. *Water Resources Research,* 26(3), p. 447–454.

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

Lamond, B., 2003. Stochastic Optimization of a Hydroelectric Reservoir using Piecewise Polynomial Approximations. *INFOR,* 41(1), pp. 51-69.

Lamond, B. & Boukhtouta, A., 1996. Optimization Long-Term Hydropower Production using Markov Decision Processes. *International Transactions in Operational Research,* Volume 3, pp. 223-241.

Larson, R. E., 1968. *State Increment Dynamic Programming.* s.l.:American Elsevier Publishing Co., New York..

Lee, J.-H. & Labadie, J., 2007. Stochastic Optimization of Multireservoir Systems via Reinforcement Learning. *Water Resources Research,,* Volume Vol. 43.

Nadalal, K. & Bogardi, J. J., 2007. *Dynamic Programming Based Operation of Reservoirs.* International Hydrology Series, ed. Cambridge: Cambridge University Press.

Pereira, M. & Pinto, L., 1991. Multi-stage Stochastic Optimization Applied to Energy Planning. *Mathematical Programming,* 52(3), pp. 359-375.

Read, E. & George, J., 1990. Dual dynamic programming for linear production/inventory systems. *Computers and Mathematics with Applications,* 19(11), pp. 29-42.

Schaffer, J., 2015. *Performance of Sampling Stochastic Dynamic Programming Algorithm with Various Inflow Scenario Generation Methods.* s.l.:MA.Sc. Thesis, Civil Engineering Department, University of British Columbia.

Shabani, N., 2009. *Incorporating Flood Control Rule Curves of The Columbia River Hydroelectric System in a Multireservoir Reinforcement Learning Optimization Model.* s.l.:MA.Sc. Thesis, Department of Civil Engineering, University of British Columbia..

Stedinger, J., Sule, B. F. & Loucks, D. P., 1984. Stochastic Dynamic Programming Models for Reservoir Operation Optimization. *Water Resources Research*, 20(11), pp. 1499-1505.

Tejada-Guibert, J., Johnson, S. A. & Stedinger, J. R., 1993. Comparison of Two Approaches for Implementing Multireservoir Operating Policies Derived using Stochastic Dynamic Programming. *Water Resources Research,* 29(12), pp. 3969-3980.

Tilmant, A., Pinte, D. & Goor, Q., 2008. Assessing Marginal Water Values in Multipurpose Multireservoir Systems via Stochastic Programming. *Water Resources Research,* Volume 44.

Turgeon, A., 1980. Optimal Operation of Multireservoir Power Systems with stochastic Inflows. *Water Resources Research,* 16(2), pp. 275-283.

Turgeon, A., 1981a. Optimal Short-Term Hydro Scheduling From the Principle of Progressive Optimality. *Water Resources Research*, 17(3), pp. 481-486.

Turgeon, A., 1981b. A Decomposition Method for Long-term Scheduling of Reservoir in Series. *Water Resources Research,* pp. 1565-1570.

Turgeon, A., 2005. Solving a Stochastic Reservoir Management Problem with Multilag Autocorrelated Inflows. *Water Resources Research,* 41(12), p. W12414.

Turgeon, A., 2007. Stochastic Optimization of Multireservoir Operation: The Optimal Reservoir Trajectory Approach. *Water Resources Research*, Volume 43.

Wyatt, T., 1996. An Integrated Simulation and Dynamic Programming Approach for Evaluating the Performance of Complex Water Resources Systems and Optimising Operating Policies: Methodology and Applications. *Paper Presented at IAHR International Workshop.*

Yakowitz, S., 1982. Dynamic Programming Applications in Water Resources. *Water Resources Research*, 18(4), pp. 673-696.

Yeh, W.-G., 1985. Reservoir Management and Operations Model: A State-of-the-Art Review. *Water Resources Research*, 21(12), pp. 1797-1818.

Appendices

Appendix A: Running the SDPOM6RM Model

A.1. Code and Computation Details

The model is composed of three main modules, the Discretizer , the SDP Model and the Value Iteration Model. In addition to these main modules, there are several small modules that calculate specific variables. The code and the computation detauils in each one of these modules are listed below. All code is listed in the AMPL syntax.

A.1.1. Discretizer Model

A.1.1.1. Code

```
######### Discretizer Model###############
###Used to descritize the state-space and the deciosn-space############
######## Created by Amr Ayad########

set Plants ordered;
set Reservoirs ordered;
set ROTR_Plants ordered;
param Online_ROTR{ROTR_Plants};
param Account_Spill_Penalty;
param Penalty_Ratio;
param Start_Months;# the start time of the study
param End_Months; # the end time of the study
param Delta_Months;
param Mid_Months_1;
param Mid_Months_2;
set Months:=Start_Months..End_Months ordered;
set Study_Years ordered;
set Month_Name ordered;
set State{Reservoirs, Months} ordered ;
set Inflows {Reservoirs, Months} ordered;
set Stater{r in Reservoirs, t in Months, s in State[r,t]} ordered;
set Rel_Decision{Reservoirs,Months} ordered ;# this is the total release decision from each reservoir including both the turbine releases and the unforced
spills.
set Out_In_flow{Plants, Months} ordered;#
param Prob_Inflow {r in Reservoirs, t in Months, Inflows[r,t]};
param Days_Months {Months} default 30;
param Days_Months_Dflt default 30;
set No_Units{Plants, Months};#Numbers of units cosidered for each plant
param Max_QT_storage_Pt{r in Reservoirs,t in  Months, No_Units[r,tparam Max_Gen_storage_Pt{r in Reservoirs,t in  Months, No_Units[r,t]};
param Abs_Max_Gen_Cap{r in Reservoirs,t in  Months, No_Units[r,t]};
param Abs_QT_Max{r in Reservoirs,t in  Months, No_Units[r,t]};
param Exp_Imp_Margin default 9.11;
param Int_Rate;# Interest rate
param Load_Reward default 0;#
param u;
param o;
param N_States{Reservoirs, Months};# Number of states
param N_Staters{r in Reservoirs,t in Months, State[r,t], nu in No_Units[r,t]};# Number of end states
param N_Releases{r in Reservoirs,t in Months, nu in No_Units[r,t]};
param Max_States{Reservoirs};# Max value of the state/plant/month + buffer
param Min_States{Reservoirs};# Min value of the state/plant/month - buffer
param Max_Staters{Reservoirs, Months};# Max value of the ending state/plant/month + buffer
param Min_Staters{Reservoirs, Months};# Min value of the ending state/plant/month - buffer
param Abs_Max_States {Reservoirs}; #Maximum absolute storage value for the specified plant;
param Abs_Min_States {Reservoirs}; #Minimum absolute storage value for the specified plant;
set counter {r in Reservoirs,t in Months}=1..N_States[r,t]+1;
set counterr {r in Reservoirs, t in Months, s in State[r,t], nu in No_Units[r,t]}=1..N_Staters[r,t,s, nu]+1 ;
set counterel {r in Reservoirs, t in Months, nu in No_Units[r,t]}=1..N_Releases[r,t,nu]+1;
param count;
param Delta_States {Reservoirs, Months} default 15000;# Calculated state step size rounded to nearst 5
param Delta_Staters {Reservoirs, Months} default 15000;# Calculated state step size rounded to nearst 5
param Delta_Releases {Reservoirs, Months};# Calculated Releases step size rounded to nearst 5
param n;# number of steps of inflow taken as steps for the storage.
param Desctz_States {r in Reservoirs,t in Months, counter[r,t]};# the generated states upon the data provided ( Max, Min, No. of desired states)
param Desctz_Staters {r in Reservoirs, t in Months, s in State[r,t],nu in No_Units[r,t], counterr[r,t,s,nu]};# the generated states upon the data provided
( Max, Min, No. of desired states)
param Desctz_Releases {r in Reservoirs, t in Months, nu in No_Units[r,t], counterel[r,t,nu]};# the generated Releases upon the data provided ( Max, Min,
QP_Max, QP_Min)
```

```
param Inflow_Step {r in Reservoirs, Months};
param Max_States_Act{Reservoirs, Months};# Max value of the state/plant/month + buffer
param Min_States_Act{Reservoirs, Months};# Min value of the state/plant/month - buffer
param Max_Staters_Act{r in Reservoirs,t in Months, State[r,t]};# Max value of the state/plant/month + buffer
param Min_Staters_Act{r in Reservoirs,t in Months, State[r,t], nu in No_Units[r,t]};# Min value of the state/plant/month - buffer

param QP_Max_Act {r in Reservoirs,t in  Months, nu in No_Units[r,t]}; # limits on max. discharge by plants:
param QP_Min_Act {r in Reservoirs,t in  Months, nu in No_Units[r,t]};# Limits on min. discharge by plants: effective but not used
param Run_Single_Res{r in Reservoirs}; #adds the flexibility of running the model for one resrevoir for a fixed storage in the other one
param Single_Storage{r in Reservoirs}; # adds the flexibility of running the model for one resrevoir for a fixed storage in the other one
###########################################################
for { v in Version}
{
data ("Data_1_"&vv&".dat");
}
###########################################################
let Delta_Months:= round((End_Months-Start_Months+1)/3,0);
let Mid_Months_1:= Start_Months+ Delta_Months-1;
let Mid_Months_2:= Mid_Months_1+ Delta_Months;
###############################################
option display_width 250;
option display_round 3;
####### CREATING THE STATES; added the flexibility of running the model for one reservoir for a fixed storage in the other one
let {r in Reservoirs, t in Months} Inflow_Step[r,t]:= if  last(Inflows [r,t])- first( Inflows [r,t])= 0 and t=first (Months)  then first (Inflows [r,t])
else if last( Inflows [r,t])- first( Inflows [r,t])= 0 and first (Months)<t<last (Months)  then Inflow_Step[r,t-1] else if last( Inflows [r,t])- first(
Inflows [r,t])= 0 and t=last (Months)  then Inflow_Step[r,first(Months)]  else if card (Inflows[r,t])>2 then min (member(2, Inflows [r,t])-first( Inflows [
r,t]), member(3, Inflows [r,t])-member(2, Inflows [r,t])) else member(2, Inflows [r,t])-first( Inflows [r,t]);
print"param     "> Inflow_Step.out;
display Inflow_Step> Inflow_Step.out;

let {r in Reservoirs, t in Months} Delta_States[r,t] := (n*Inflow_Step[r,t]* Days_Months_Dflt);# can use also "floor" and "ceil" functions instead of round
let {r in Reservoirs, t in Months} Max_States_Act[r,t]:= if Run_Single_Res[r]=0 then round (Max_States[r]/(Delta_States[r,t]),0)*(Delta_States[r,t])else
round ((Single_Storage[r]/Delta_States[r,t]),0)*Delta_States[r,t] ;# Max value of the state/plant/month + buffer
let {r in Reservoirs, t in Months} Min_States_Act[r,t]:=if Run_Single_Res[r]=0 then round (Min_States[r]/(Delta_States[r,t]),0)*(Delta_States[r,t]) else
round ((Single_Storage[r]/Delta_States[r,t]),0)*Delta_States[r,t] ;# Min value of the state/plant/month - buffer
let {r in Reservoirs, t in Months} N_States[r,t]:= (Max_States_Act[r,t]- Min_States_Act[r,t])/ (Delta_States[r,t]);

for { r in Reservoirs, t in Months}
{
let count:=1;
for { c in counter [r,t]}
{
let Desctz_States[r,t, c] := Min_States_Act[r,t]+Delta_States[r,t]*(count-1) ;
let count:=count+1;
}
}
# Writing the created states in a data file in order to read it again for the optimizer
for { r in Reservoirs, t in Months}
{
print "set State ["&r&","&t&"]:=" >("States.dat");
for {c in counter [r,t]}
{
printf "%12.0f",Desctz_States[r,t, c]>("States.dat");
}
print ";" > ("States.dat");
printf "\n" >("States.dat");
}
```

```
####### Limits on plant discharge
param QP_Max {p in Plants,t in Months, No_Units[p,t]}; # limits on max. discharge by plants:
param QP_Min {p in Plants,t in Months, No_Units[p,t]}; # limits on max. discharge by plants:
####### Limits on Turbine Discharge
param QT_Min { Reservoirs, Months};# Limits on min. discharge by turbines: effective but not used
#####################data
for { v in Version}
{
data ("Data_2_"&v&".dat");
}
##########################################################

for{rr in 1..3}
{
if rr=1 then {
let u:= Start_Months;
let o:= Mid_Months_1;
}
else if rr=2 then {
let u:= Mid_Months_1+1;
let o:= Mid_Months_2;
}
else{
let u:= Mid_Months_2+1;
let o:= End_Months;
}
############ CREATING THE STATERS
let { r in Reservoirs, t in Months: u <=t<=o} Delta_Staters[r,t] := (n*Inflow_Step [r,t]*Days_Months_Dflt);
let {r in Reservoirs, t in Months, s in State[r,t]} Max_Staters_Act[r,t,s]:=if Run_Single_Res[r]=0 then round (max(min(s+last(Inflows [r,t])*
Days_Months_Dflt,Max_Staters[r,t]), Min_Staters[r,t])/(Delta_Staters[r,t]),0)*(Delta_Staters[r,t]) else round ((Single_Storage[r]/Delta_Staters[r,t]),0)*
Delta_Staters[r,t] ;# Max value of the state/plant/month + buffer
let {r in Reservoirs, t in Months, s in State[r,t],nu in No_Units[r,t]} Min_Staters_Act[r,t,s,nu]:=if Run_Single_Res[r]=0 then round(min(max(s+(first(
Inflows [r,t])-QP_Max[r,t,nu])*Days_Months_Dflt,Min_Staters[r,t]), Max_Staters[r,t])/(Delta_Staters[r,t]),0)*(Delta_Staters[r,t]) else round ((
Single_Storage[r]/Delta_Staters[r,t]),0)*Delta_Staters[r,t];# Min value of the state/plant/month - buffer
let {r in Reservoirs, t in Months, s in State[r,t],nu in No_Units[r,t]} N_Staters[r,t,s, nu]:= (Max_Staters_Act[r,t, s]- Min_Staters_Act[r,t,s, nu])/ (
Delta_Staters[r,t]);

for { r in Reservoirs,t in Months,s in State[r,t], nu in No_Units[r,t]: u <=t<=o}
{
let count:=1;
for { c in counterr [r,t,s,nu]}
{
let Desctz_Staters[r,t,s,nu, c] := if t< last (Months) then Min_Staters_Act[r,t,s, nu]+Delta_Staters[r,t]*(count-1) else Min_Staters_Act[r, first (Months),
s,nu]+Delta_Staters[r,first (Months)]*(count-1);
let count:=count+1;
}
}
###Writing the created staters in a data file in order to read it again for the optimizer
for { r in Reservoirs,t in Months,s in State[r,t], nu in No_Units[r,t]: u <=t<=o}
{
print "set Stater ["&r&","&t&","&s&"]:=" >("Staters_"&rr&".dat");
for {c in counterr [r,t,s,nu]}
{
printf "%12.0f",Desctz_Staters[r,t,s,nu, c]>("Staters_"&rr&".dat");
}
print ";" > ("Staters_"&rr&".dat");
printf "\n" >("Staters_"&rr&".dat");
}
```

```
############ Creating the Release Decisions:  plant releases
let {r in Reservoirs, t in Months, nu in No_Units[r,t]} QP_Min_Act[r,t,nu]:= round (QP_Min[r,t,nu]/Inflow_Step [r,t],0)*(Inflow_Step [r,t]) ;# Max value of
the state/plant/month + buffer
let {r in Reservoirs, t in Months, nu in No_Units[r,t]} QP_Max_Act[r,t,nu]:= round (QP_Max[r,t,nu]/Inflow_Step [r,t],0)*(Inflow_Step [r,t]) ;# Min value of
the state/plant/month - buffer
let {r in Reservoirs, t in Months} Delta_Releases[r,t] := n*Inflow_Step [r,t];# can use also "floor" and "ceil" functions instead of round
let {r in Reservoirs, t in Months, nu in No_Units[r,t]} N_Releases[r,t,nu]:= (QP_Max_Act[r,t,nu]- QP_Min_Act[r,t,nu])/ (Delta_Releases[r,t]);

for { r in Reservoirs}
{
for {t in Months, nu in No_Units[r,t]: u <=t<=o}
{
let count:=1;
for { c in counterel [r,t,nu]}
{
let Desctz_Releases[r,t, nu,c] :=QP_Min_Act [r,t,nu] +Delta_Releases [r,t]*(count-1);
let count:=count+1;
}
}
}

# Writing the created <<< Release Decions >>>> in a data file in order to read it again for the optimizer ####################
for { r in Reservoirs}
{
for {t in Months, nu in No_Units[r,t]: u <=t<=o}
{
print "set Rel_Decision ["&r&","&t&"]:=" >("Rel_Decision_"&rr&".dat");
for {c in counterel [r,t,nu]}
{
printf "%12.3f",Desctz_Releases[r,t,nu,c]>("Rel_Decision_"&rr&".dat");
}
print ";" > ("Rel_Decision_"&rr&".dat");
printf "\n" >("Rel_Decision_"&rr&".dat");
}
}
}
print"##################################################################";
print"##################################################################";
print "##Discretization of State-space and Decision Variable  is done##";
print"##################################################################";
print"##################################################################";
close;
```

A.1.1.2. Computation Details

Table 4: Calculations Details in the Discretizer Model

Step Number	Parameter Calculated	Notes
1	Delta_States, Delta_Staters. Delta_Releases	The increment of the starting, terminal states and plant release respectively. They are calculated as multiples of the inflows increment. When the multiple equals 1, they are exactly the same value as the inflow increment.
2	Max_States_Act, Min_States_Act	A rounded up/down number for the maximum/minimum starting state to the nearest "Delta_States"
3	N_States	The number of starting states between the" Max_States_Act" and the "Min_States_Act"
4	Max_Staters_Act	A rounded up number for the maximum terminal state to the nearest "Delta_Staters" capped by the maximum inflow in a given stage.
5	Min_Staters_Act	A rounded down number for the minimum terminal state to the nearest "Delta_Staters" capped by the minimum inflow and the maximum plant release in a given stage.
6	N_Staters	The number of terminal states between the" Max_Staters_Act" and the "Min_Staters_Act" in a given stage.
7	QP_Max_Act ,QP_Min_Act	A rounded up/down number for the maximum/minimum plant release in a given stage to the nearest "Delta_States".
8	N_Releases	The number of plant releases in a given stage.

A.1.2. SDP Model

A.1.2.1. Code

```
## This is the SDP for 6 Reseroirs
## Created by Amr Ayad
## Modified on May 20th 2011 to cope with Ziad recommendations, he required one unified release Rel_Decision['GMS',t], Rel_Decision['MCA',t] and
one unified load#####
## Modified Sunday July 10th 2011. AA
## Modified Wednesday August 10th 2011. AA suggestions by Guan Ziming(cleaning the code, added Rel_Decision['GMS',t], Rel_Decision['MCA',t] to the
Trans_Prob and Trans_Prob_S)
## Modified Friday August 19th. AA ( modifieying it to work for 3 plants)this model was converted again to 2 plants only on Oct 1st 2011
## Modified Wednesday August 24th Indexed the HK over plants, time, starting state and ending state. AA
## Modified Monday August 29th Added some parametrs and sets to automate the state generation ( Variable states Model).
## Modified Sept 2011 to add several parameters, they are tagged by "New Sept 2011"####################################
## Modified Oct 4th : Added the generation of IPPs and thermal as flat files. Generate the FB data from storage data
## Modified Oct 9th: Tried to add some parameters to conclude the optimization output but I think it is not working right ( something wrong with my
logic)
## Modifying the entire formulation after Ziad's meeting on Oct 12.( the modification was holistic and touched most of the indexing and
formulation)
## Modifying the model to index the terminal state over inflows so as to limit the visits from the initial states to the possible terminal states,
8th December 2011
## The model can now calculate the MVW and MV of Energy for each reservoir
## I am trying to implement penalty in the calculations ( it didn't work and I commented it out)
## Now using Inflow_500_Smoothed and Inflow_500_Smoothed_zero(better as it gives more realistic PV values) June 2012
## found out that the best descritization is 1000 cms for MCA and 500, 750 or 100 for GMS.
## Trying out version L9 which has the plant releases indexed over the storage states.(didn't work, back to L8 again)
##### Best working version so far is L8, July 3rd 2012...
## July 4th 2012, updates (L8--->L10) added functionality to retrieve the optimum policy, dates of the stages**needs enhancements**, using the
actual days in different months in generation calculation.
## Corrected the FB and the HK regression equations..July 5th
## Added FB equations for REV, PCN. July 6th.
##### Best working version so far is L10, July 6th 2012...
## the optimal monthly trade values are now produced as an output in the model July 12th 2012..
## Introduced the Availabilty parameter to the model July 12th 2012.
## July 13th 2012.....Migration from version (L10--->L11)..the difference is that I am trying to implement Max_Gen_Limits that is a function of the
storage (i.e. considering the loss of head in the turbine capacity)
## July 16th 2012.....Migrating from version (L11--->L12)..adopting the use of turbine release as a function of the storage which was first
introduced in version L9 before but the total plant releases were used back then not the turbine releases
##                   Added parameters  Max_QT_storage_Pt, Max_Gen_storage_Pt,  Abs_Max_Gen_Cap and set No_Units
## July 17th 2012.....HK equations were modified ( better accuracy)..
## July 24th 2012.....Added parameter Abs_QT_Max
## July 31st 2012.....The names of the data files that have been dramatically changed were named" xxxx_L12.dat"
## August 2nd 2012....Finally after the modifications from L10-->L12 have been implemented, now the results are quite acceptable.
##                   Added a data file " Units.dat" and modified the parameter No_Units As well as the parameters : Max_QT_storage_Pt,
Max_Gen_storage_Pt, Abs_Max_Gen_Cap, Abs_QT_Max,to be indexed over stages so that the study can be flexible to include adding a unit or number of
units on-line in the middle of the study
##                   Added a data file "Turbine_Gen_Spec.dat"
## August 3rd 2012....Created a new data file "OR.dat" for relibility of operation " Operating Reserve"
##                   Starting to add PCN and REV and may be ARD.
##                   Adding the new parameters "Inflow_ROTR, Tot_Inflow_ROTR" and modifying the parameter" Total_Sys_Inflow" and adding the data
file" Inflows_ROTR"
## August 7th 2012....the parameter  "OR" is noticed to cause the MVW to concide at one graph in some cases!.
##                   Added a new data files Corr_Matric, Inflow_ORTR   and the set ROTR_Plants
##                   Migrating from L12---> L13: calculating the PCN and REV generation compared to using fixed values in the previous version.
##                   Added severeal parameters to calculate the HK, Turbine release, spill, generation for both PCN and REV
## August 8th 2012....Extended the Outage.dat file to include REV and PCN.
##                   Added a new set " Plants" that contains all the storage and non-storage plants"ROTR plants" and accordingly changed the
indexing of several parameters to match that.
##                   Added a new data file ' Out_In_flow.dat'
##                   The code is stuck after t=1!!!!!!!!!!!!!THE REASON: FOUND OUT THAT INCREASING THE NUMBER OF SETS THAT ARE BEWEEN {} AND
FOLLOWS THE 'let' COMMAND CAUSES THE CODE TO STOP WITHOUT GIVING ANY WARNINGS!!!( bug)
##                   The names of the data files that have been dramatically changed were named" xxxx_L13.dat"
## August 9th 2012... Added the set'Version' and files 'Version.dat' and 'Version.mod' to control which veriosn the user likes to run and populated
it back from veriosn L13 to L8.
```

```
##                      Big change to the file SDP.run to match the changes after adding the set Version  which resulted in automating the version
selection.
## August 10th 2012.. Migrating from Version L13--> L14: Indexing the QP_Max parameter over the units.which led to changing the indexing of 11
parameters in the file Discretizer_QP.mod.
##                      Veriosn L14 is working fine now
## August 13th 2012   Migrating from Veriosn L14--> L15: Taking out most of the regression equations from the main code" SDP.mod" and distributig
them in (6) model files, each taking the same exact name of the parameter it calculates.
##
## Sept 12th 2012.... Migrating from Version L15--> L16: correlating the REV and ARD inflows to MCA & PCN to GMS which needed to change the
indexing of the parameter Inflow_ROTR and using a new model files Inflow_ROTR.mod instead of  Inflow_ROTR.dat. Also several parameters related to
generation and Trade had their indexing changed. Also the file HK_ROTR.mod was changed.
##
## Sept 19th 2012.... Migrating from Version L16--> L17: Adding and modeling Site_C (STC) as a ROTR plant
##                      Found and error in Veriosn L16 ( DON'T USE IT ANY MORE)
##                      Changed the equation that calculates the prices
##                      Now, the output folder name contains the version name, number of stages the model was run for and the current time and added
a time_log file called " Run_Times_Stats.amr"
## Oct 4th 2012...... Found a bug in generating the staters range in the Discretizer model and fixed it back to versions L17, L15
##                      As per Ziad request, I have added parameters to Calculate  the MVW ( previously it was the MVE only)
##                      Fixed some bugs in the discretizer model to be able to receive any set of Inflow data.
##
##############################################################################################################
###########################  VERSIONS AVAILABLE ARE:L18 L17 L15  L13  L12  L10  L08       ################################
##############################################################################################################
#
#
##################################################
##
##
## January 4th 2013...Migrating from Version L17--> L18: Basically adding flexability to the model to work with or without any of the ROTR
plants.Parameters Online_ROTR was added. Main files changed are: SDP_L18.mod, Discretizer_L18.mod, Horizon_l18.dat, Inflow_ROTR_L18.mod,
HK_RoTR_L17.mod; HK_rough_L18.mod
##                                                       Use the total HK for all the plants on one river to caculate the Marginal value of water/
energy in the storage plant instead of using the HK of this plant only. Parameter HK_rough was modified
##                                                       OR was commented out from the generation
## January 8th 2013...Added the ARD inflow to the total system inflow. It only affects the prices not the generation as ARD generation is not
optimized.
##                      Added three new parameters Gen_PCN_fixed, Gen_REV_fixed, Gen_ARD_fixed to used fixed generatrion for those ROTR plants when
not needed to be optimized.New data file was created for this purpose(Gen_ROTR_Fixed_L18.dat)
## Janaury 10th 2013..Added a new parameter Gen_STC_fixed to complete the ROTR group fixed generation.   Got the numbers from a recent study by
Joel Evans (Reference is at Gen_ROTR_Fixed_L18.dat))
##                      Added a new paarmeter to evaluate the domestic load (param Load_Reward)...didn't work as expected.
## January 18th 2013..Moved the price calculations to a new separate model file: (Prices_&v&.mod)
##                      Adding the flexibility to run the model for one storage reservoir only if needed. Main affected files are: Horizon.dat,
SDP.mod, Descretizer_QP.mod.
##                      As a result of the change above, the model can be run now for GMS only, MCA only or even for one point of storage for GMS and
MCA(produces one value for PV per stage)
##                      Modified the Spot_Buy and Spot_Sell Calaculations to account for contract Exp/Imp in the tie limits.
##                      Modified the HK_ROTR calculations,. Main file affcted is HK_ROTR_&V&.mod.
## January 22nd 2013..Added penalty on spilled water/energy which is taken into account in the reward function. New parameters were added:
Account_Spill_Penalty, param Penalty_Ratio, Tot_Spilled_Energy, Ex_Penalty_Spilled_Energy. Files affected are: Sdp_&V&.mod, Descretizer_Qp_&V&.mod
and Horizon_&V&.dat. Added the flexibility to set the penalty-on-spills calculations on/off.
##
## January 24th 2013..Removed the value iteration procedure to a new model file: Value_Iteration&V&.mod.
##                      Energy_Curt_Tie_Exp, Energy_Curt_Tie_Imp
## January 28th 2013..a new data file is created "Disc_Space_"&v&".dat" to include the output files created by the Discretizer_&V&.mod.
##                      Changed how the parameter "Tot_Spilled_Energy" is calculated
##                      Minor changes to two parameters: Max_Gen_Limits and QT_Max for better regression. That affected the files Max_Gen_&v&.mod and
QT_MAx_&V&.mod.
```

```
## February 5th 2013..New output file is created "Final_Results_Listed.out" so that the results are copied from this file to a VBA spreadsheet
called "PV_Grapher+3D.xlsm". Pushing the button in this spreadsheet will draw all the necessary graphs needed for presentation. Note that it is not
dependent on the number of stages or states. It needs the user to provide some simple information to start.
## February 27th 2013.the above output files is now created in another form ".csv" as well as in the old form ".out"
##
## May 10th 2013      Fixed a bug in renaming the output folder
##
##############################################################################################################

option log_file 'screen.out';
option eexit -100000000;### this option is to force AMPL to continue runnning even after 10 warning messages which usually are generated because
the data read are more than the horizon specified needs..

set Plants ordered; # set of all plants considered in the system
set Reservoirs ordered;## set of storage plants considered
set ROTR_Plants ordered;# set of plants that have small/ no storage capacity and considered as Run-of-the-river plants"ROTR".

param Online_ROTR{ROTR_Plants};

param Start_Months;# the start time of the study, controlled from the ' Horizon.dat' file
param End_Months; # the end time of the study
set Months:=Start_Months..End_Months ordered;## the span of the time horizon
set Month_Name ordered;
set Study_Years ordered;## future study years
set Stage_Name ordered:= setof { j in Study_Years, i in Month_Name} if i='Oct' or i= 'Nov' or i='Dec' then i & "_" & j else i & "_" & j+1;
set Stages ordered; #used in renaming the output folder
set State{Reservoirs, Months} ordered;## starting state at each time step
set Rel_Decision{Reservoirs,Months} ordered ;# total release decision from each reservoir including both the turbine releases and the forced
spills.
set Inflows {Reservoirs, Months} circular; # natural inflows to each reservoir, the only stochastic variable in the problem so far

set Stater{r in Reservoirs, t in Months, s in State[r,t]} ordered;  ## teraminal state at each time step
param Prob_Inflow {r in Reservoirs, t in Months, Inflows[r,t]}; # Discrete probability distribution of the inflows
set Load {Months} ;# load = local demand

param Flow_Corr{Reservoirs,ROTR_Plants};# flow correlation between the storage plants and the run-of-the-river plants
param Inflow_ROTR{ROTR_Plants,r in  Reservoirs ,t in Months, Inflows[r,t]} default 0;# deterministic inflows to PCN  and REV# added August 2012
param Tot_Inflow_ROTR{rp in ROTR_Plants,r in Reservoirs, t in Months, Inflows[r,t],d in Rel_Decision[r,t]}; # total inflow to the PCN "Dinosaur"
reservoir and REV reservoir
param Upper_Bound_State{Reservoirs, Months};
param Lower_Bound_State{Reservoirs, Months};
param Upper_Bound_Stater{Reservoirs, Months};
param Lower_Bound_Stater{Reservoirs, Months};

param Total_Sys_Inflow{ t in Months ,i1 in Inflows['GMS',t],i2 in Inflows['MCA',t]};# summation of the total inflow to the systems which is used to
generate the price set.
#param Probability_Mtrx_Size{r in Reservoirs, t in Months,s in State[r,t],i in Inflows[r,t],Stater[r,t,s], Rel_Decision['GMS',t],
Rel_Decision['MCA',t]}; # I didn't activate it yet but I have a fitted form from Excel  = [( number of states* number of staters * Number of Inflow
scenarios)^( number of reservoirs)]* number of decisions .i.e. (s*w*I)^r*d  !!
param Int_Rate;# Interest rate
param Gama;# Discount rate
param iter default 0;
param Exp_Imp_Margin default 9.11; ## difference between the import and export prices, fixed
param Load_Reward default 0;#

param Run_Single_Res{r in Reservoirs}; # add the flexibility of running the model for one resrevoir for a fixed storage in the other one
param Single_Storage{r in Reservoirs}; # add the flexibility of running the model for one resrevoir for a fixed storage in the other one
```

```
param n;# number of inflow increments taken as storage increment.
param Days_Months {Months} default 30;## number of days in each month
param Days_Months_Dflt default 30;
#param Nill:=1122002211;#"NON" symbolic;## dummy parameter
###########################################################
param Max_States{Reservoirs};# Max value of the state/plant/month + buffer
param Min_States{Reservoirs};# Min value of the state/plant/month - buffer
param Max_Staters{Reservoirs, Months};# Max value of the ending state/plant/month + buffer
param Min_Staters{Reservoirs, Months};# Min value of the ending state/plant/month - buffer
param Abs_Max_States {Reservoirs}; #Maximum absolute storage value for the specified plant;
param Abs_Min_States {Reservoirs}; #Minimum absolute storage value for the specified plant;
param Delta_States {Reservoirs, Months} default 15000;# Calculated state step size rounded to nearst 5
param Inflow_Step {r in Reservoirs, Months} default 500;
##############################################################################
####### Plant's turbines outflow outflows upper bounds  Logic: not all the resulting +ve    ####################################
####### Trade (to sell in the spot market) could be used as it should be compared to the max. that could be ####################################
####### generated by the plant and then the min of them will be considered in the PV function        ####################################
param Max_Gen_Limits {r in Reservoirs,t in Months, s in State[r,t],Stater[r,t,s]}; #limits on maximum generation of plant
param Min_Gen_Limits {r in Reservoirs, t in Months}; #limits on minimum generation of plant:
param Outage{p in Plants, t in Months} default 1; # Factors <=100% representing the maximum generation capacity ratio which is reduced by  the
outage schedule
param Availability{p in Plants, t in Months} default 1; # Factors <=100% representing the experience-based estimated online time of the plants
during the month
param OR{r in Reservoirs, t in Months} default 0.05; #Operating Reserve percentage# could be used as a constraint on generation..
##parameters"G_ORO, G_RM_BUFFER, DepCap" and file ORO.dat#
param G_OR{r in Reservoirs, t in Months, s in State[r,t], w in Stater[r,t,s], Rel_Decision[r,t]} default 0;#Operating Reserve amount in GWh# see
the above parameter "OR"#
set No_Units{p in Plants, t in Months};#Numbers of units cosidered for each plant.
####### IPP and thermal
param IPP_Therm{ Months};# generation of the IPP and thermal. Added to the plants generation to get the total generation of the system
##############################################################################
param Term_W {r in Reservoirs, t in Months,s in State[r,t],i in  Inflows[r,t], Stater[r,t,s], Rel_Decision[r,t]};
param Generation{r in Reservoirs, t in Months, s in State[r,t], w in Stater[r,t,s], Rel_Decision[r,t]} default 0;
param Gen_ROTR{rp in ROTR_Plants,r in Reservoirs, t in Months,i in Inflows[r,t],Rel_Decision[r,t] };
param Gen_PCN_fixed{t in Months};
param Gen_REV_fixed{t in Months};
param Gen_ARD_fixed{t in Months};
param Gen_STC_fixed{t in Months};
param Tot_Gen{t in Months,s1 in State['GMS',t],s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in
Inflows['MCA',t],Stater['GMS',t,s1],Stater['MCA',t,s2], Rel_Decision['GMS',t], Rel_Decision['MCA',t]} ;
#param Monthly_Gen {t in Months,s1 in State['GMS',t],s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in
Inflows['MCA',t],Stater['GMS',t,s1],Stater['MCA',t,s2], Rel_Decision['GMS',t], Rel_Decision['MCA',t]} ;

param Trade {t in Months, s1 in State['GMS',t], s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],Stater['GMS',t,s1],Stater['MCA',
t,s2], Rel_Decision['GMS',t], Rel_Decision['MCA',t]};

param State_Prob{r in Reservoirs,t in Months, s in State[r,t], i in Inflows[r,t], Stater[r,t,s], Rel_Decision[r,t]};    #Calculated
param Trans_Prob {t in Months,s1 in State['GMS',t], s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in
Inflows['MCA',t],Stater['GMS',t,s1],Stater['MCA',t,s2], Rel_Decision['GMS',t],Rel_Decision['MCA',t]};   #Calculated
param Trans_Prob_S {t in Months, s1 in State['GMS',t], s2 in State['MCA',t],Stater['GMS',t,s1],Stater['MCA',t,s2],Rel_Decision['GMS',t],
Rel_Decision['MCA',t]};         #Calculated
param Trans_Prob_S_S {t in Months, s1 in State['GMS',t], s2 in State['MCA',t],Rel_Decision['GMS',t], Rel_Decision['MCA',t]};

param Contr_Exp {t in Months};
param Contr_Imp {t in Months};
param Spot_Buy {t in Months, s1 in State['GMS',t], s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in
Inflows['MCA',t],Stater['GMS',t,s1],Stater['MCA',t,s2], Rel_Decision['GMS',t], Rel_Decision['MCA',t]};  #Calculated
param Spot_Sell {t in Months, s1 in State['GMS',t], s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in
Inflows['MCA',t],Stater['GMS',t,s1],Stater['MCA',t,s2], Rel_Decision['GMS',t], Rel_Decision['MCA',t]};  #Calculated
```

```
param Energy_Curt_Tie_Imp{t in Months, s1 in State['GMS',t], s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],Stater['GMS',t,s1],
Stater['MCA',t,s2], Rel_Decision['GMS',t], Rel_Decision['MCA',t]};      #Calculated
param Energy_Curt_Tie_Exp{t in Months, s1 in State['GMS',t], s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],Stater['GMS',t,s1],
Stater['MCA',t,s2], Rel_Decision['GMS',t], Rel_Decision['MCA',t]};      #Calculated


param Max_Mthly_Spot_Buy_GW {t in Months};      #Calculated
param Min_Mthly_Spot_Buy_GW {t in Months};      #Calculated
param Max_Mthly_Spot_Sell_GW {t in Months};     #Calculated
param Min_Mthly_Spot_Sell_GW {t in Months};     #Calculated
param Optimum_Spot_Export_GW{t in Months, s1 in State['GMS',t], s2 in State['MCA',t]};
param Optimum_Spot_Import_GW{t in Months, s1 in State['GMS',t], s2 in State['MCA',t]};

#param Infeasible_Spot_Buy {t in Months, s1 in State['GMS',t], s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in
Inflows['MCA',t],Stater['GMS',t,s1],Stater['MCA',t,s2], Rel_Decision['GMS',t], Rel_Decision['MCA',t]};  #Calculated
#param Infeasible_Spot_Sell {t in Months, s1 in State['GMS',t], s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in
Inflows['MCA',t],Stater['GMS',t,s1],Stater['MCA',t,s2], Rel_Decision['GMS',t], Rel_Decision['MCA',t]};  #Calculated

param EX_Imp_Cost{t in Months,s1 in State['GMS',t],s2 in State['MCA',t],Rel_Decision['GMS',t], Rel_Decision['MCA',t]} ;
#Calculated
param EX_Exp_Rev {t in Months,s1 in State['GMS',t],s2 in State['MCA',t],Rel_Decision['GMS',t], Rel_Decision['MCA',t]} ;
#Calculated
param Cont_Rev{t in Months,s1 in State['GMS',t],s2 in State['MCA',t],Rel_Decision['GMS',t],Rel_Decision['MCA',t]} default 0 ;
#Calculated
param Policy_Income {t in Months, s1 in State['GMS',t],s2 in State['MCA',t], Rel_Decision['GMS',t], Rel_Decision['MCA',t]} ;
#Calculated

param PV{t in Months,s1 in State['GMS',t],s2 in State['MCA',t], Rel_Decision['GMS',t], Rel_Decision['MCA',t]} default 1;
param PV_Max {t in Months, s1 in State['GMS',t],s2 in State['MCA',t]} default 100;                          #Calculated
param PV_Fin {t in Months, s1 in State['GMS',t], s2 in State['MCA',t]} default 100;                         #Calculated: future cost to
go function
param PV_diff{t in Months,s1 in State['GMS',t], s2 in State['MCA',t]};                                      #Calculated
param PV_diff_total default 10;
param Optimum_Policy_GMS{t in Months, s1 in State['GMS',t],s2 in State['MCA',t]};                     #Calculated
param Optimum_Policy_MCA{t in Months, s1 in State['GMS',t],s2 in State['MCA',t]};
param HK{ r in Reservoirs, t in Months,s in State [r,t], w in Stater[r,t,s]} ;
param HK_ROTR {rp in ROTR_Plants,r in Reservoirs, t in Months,i in Inflows[r,t],Rel_Decision[r,t]};
############ Declaring Spill Parameters #################
param Turbine_Release{r in Reservoirs,t in Months,s in State [r,t],w in Stater[r,t,s], Rel_Decision[r,t]};
param Plant_Release{r in Reservoirs,t in Months, Rel_Decision[r,t]};
param Spill{r in Reservoirs,t in Months,s in State [r,t],w in Stater[r,t,s], Rel_Decision[r,t]};
param Turbine_Release_ROTR{rp in ROTR_Plants,r in Reservoirs, t in Months, i in Inflows[r,t], Rel_Decision[r,t]}; #  PCN and REV turbine release
param Spill_ROTR{rp in ROTR_Plants, r in Reservoirs, t in Months, i in Inflows[r,t], Rel_Decision[r,t]}; # PCN and  REV spills


param Tot_Spilled_Energy{t in Months,s1 in State['GMS',t],s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],Stater['GMS',t,s1],
Stater['MCA',t,s2], Rel_Decision['GMS',t], Rel_Decision['MCA',t]};##
param Ex_Penalty_Spilled_Energy{t in Months, s1 in State['GMS',t],s2 in State['MCA',t], Rel_Decision['GMS',t], Rel_Decision['MCA',t]};#
param Account_Spill_Penalty; #
param Penalty_Ratio;#

#param Penalty{r in Reservoirs,t in Months,s in State [r,t],i in Inflows[r,t],w in Stater[r,t,s], Rel_Decision[r,t]};
#param Tot_Penalty{t in Months,s1 in State['GMS',t],s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in
Inflows['MCA',t],Stater['GMS',t,s1],Stater['MCA',t,s2], Rel_Decision['GMS',t], Rel_Decision['MCA',t]} ;
#param EX_Penalty{t in Months,s1 in State['GMS',t],s2 in State['MCA',t],Rel_Decision['GMS',t],Rel_Decision['MCA',t]} ;          #Calculated
#\ check of Hydraulic Balnace and Load-Resource Balance and Feasibilty
param Hydraulic_Balance {r in Reservoirs,t in Months, s in State[r,t], i in Inflows[r,t], w in Stater[r,t,s], d in Rel_Decision[r,t]} ;# Units are
in cms.day
param Load_Reseource_Balance {t in Months,s1 in State['GMS',t],s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in
Inflows['MCA',t],Stater['GMS',t,s1],Stater['MCA',t,s2], Rel_Decision['GMS',t], Rel_Decision['MCA',t]} ;### Units are in MWH
```

```
#param Feasibility_Check {t in Months, s1 in State['GMS',t],s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in
Stater['GMS',t,s1],w2 in Stater['MCA',t,s2], d1 in Rel_Decision['GMS',t], d2 in Rel_Decision['MCA',t]};### Feasibility Check
# Declaring parameters for Trade limits of Transmission################
param Trade_Exp_Limit{Months};# Export trade limit##
param Trade_Imp_Limit{Months};# Import trade limit##
####### Limits on plant discharge
#old#param QP_Max {Reservoirs, Months}; # limits on max. discharge by plants:
param QP_Max {p in Plants, t in Months, No_Units[p,t]}; # limits on max. discharge by plants:
#param QP_Min { Reservoirs, Months};# Limits on min. discharge by plants: effective but not used
param QP_Min {p in Plants, t in Months, No_Units[p,t]}; # limits on max. discharge by plants:
####### Limits on Turbine Discharge
param QT_Max {r in Reservoirs,t in Months,s in  State[r,t],Stater[r,t,s]}; # Modified to be indexed over the storage ....limits on max. discharge
by turbines: effective but not used
param QT_Min { Reservoirs, Months};# Limits on min. discharge by turbines: effective but not used
param Max_QT_storage_Pt{r in Reservoirs, t in Months, No_Units[r,t]};# the storage point at which the turbine release peaks then goes down beacuse
of reaching the generator capacity.
param Max_Gen_storage_Pt{r in Reservoirs, t in Months, No_Units[r,t]};# the storage point at which the generator capacity peaks then levels beacuse
of reaching the generator capacity.
param Abs_Max_Gen_Cap{p in Plants, t in Months, No_Units[p,t]};# the generator capacity corresponsing to the number of units used
param Abs_QT_Max{p in Plants, t in Months, No_Units[p,t]};# Absolute maximum value of turbine release
param Start_Time:= ctime()symbolic;
########################################### THE DATA FILES 1&2
for { v in Version}
{
data ("Data_1_"&v&".dat");
data ("Data_2_"&v&".dat");
data ("Disc_Space_"&v&".dat");
}
##############################################################################
printf "\n" >Run_Times_Stats.amr;
printf "#######################################################################" >Run_Times_Stats.amr;
printf "\n" >Run_Times_Stats.amr;
for { v in Version}
{
print"This run is done by Version "&v&" of the SDPOM2R model for "&End_Months&" Stages..." > Run_Times_Stats.amr;
}
printf "#######################################################################" >Run_Times_Stats.amr;
printf "\n" >Run_Times_Stats.amr;
print"This run started at "&Start_Time&""> Run_Times_Stats.amr;
printf "\n" >Run_Times_Stats.amr;
####### HK calculations
for { v in Version}
{
commands ("HK_"&v&".mod");
}
##############################################################################display HK>HK.out;
####### parameters for Prices Calculations
param a {t in Months};
param b {t in Months};
param c {t in Months};
param dd {t in Months};
param Exp_Price  {t in Months,i1 in Inflows['GMS',t],i2 in Inflows['MCA',t]};
param Imp_Price  {t in Months,i1 in Inflows['GMS',t],i2 in Inflows['MCA',t]};
param Cont_Price {t in Months,i1 in Inflows['GMS',t],i2 in Inflows['MCA',t]};
########################################### THE Prices data data Price_Coef.dat;#### Coefficients of the 2nd degree polynomial equation used to
claculate the Exp, Imp and contracted prices.
#include Price_Coef_Flat.dat;#### Coefficients of the 2nd degree polynomial equation used to claculate the Exp, Imp and contracted prices.
########################################### THE DATA for { v in Version}
{
```

```
data ("Data_3_"&v&".dat");
}
# Inflows from the ROTR Plants#############################################
for { v in Version}
{
commands ("Inflow_ROTR_"&v&".mod");
}


for { v in Version}
{
commands ("Prices_"&v&".mod");
}

let Gama:= 1/(1+Int_Rate);

option display_width 250;
option display_round 4;

#####LOOPING FOR STAGES
print "set Stages:=" > ("Set_Stages.use");
printf "\n" >("Set_Stages.use");

for {t in Months }#### looping for Months#####
{
display t;
### Calculating the turbine limits
for { v in Version}
{
commands ("QT_Max_"&v&".mod");
}
######Calculating the core parameters
let { r in Reservoirs, s in State[r,t], i in Inflows[r,t],w in Stater[r,t,s], d in Rel_Decision[r,t]} Term_W [r,t,s,i,w,d] :=
((s)/Days_Months_Dflt)+i-d; #((s-w)/Days_Months[t])+i; # potential of free water (to be released-discharged =) to use for generation and/or forced
spills for each plant ..Units cms
let {r in Reservoirs, d in Rel_Decision[r,t]} Plant_Release[r,t,d]:=d;
let {r in Reservoirs, s in State[r,t], w in Stater[r,t,s], d in Rel_Decision[r,t]} Turbine_Release [r,t,s,w,d]:=  min (QT_Max[r,t,s,w],
Plant_Release[r,t,d]); #### Hourly rate of energy produced by plant in MWh
let {r in Reservoirs, s in State[r,t], w in Stater[r,t,s], d in Rel_Decision[r,t]} Spill[r,t,s,w,d]:=(d) -Turbine_Release [r,t,s,w,d];
let {r in Reservoirs, s in State[r,t], i in Inflows[r,t], w in Stater[r,t,s], d in Rel_Decision[r,t]} Hydraulic_Balance [r, t,s,i,w,d]:=Term_W
[r,t,s,i,w,d]-(w/Days_Months_Dflt);# SiW [r,t,s,i,w,d]-(d);

display Spill>spill.out;
### Calculating the total inflows, turbine releases, spills and  HK to PCN, REV and STC
let {i2 in Inflows['MCA',t],d2 in Rel_Decision['MCA',t]} Tot_Inflow_ROTR['REV','MCA',t,i2,d2]:= if Online_ROTR['REV']=1 then
Inflow_ROTR['REV','MCA',t,i2]+d2 else 0;
let {i1 in Inflows['GMS',t],d1 in Rel_Decision['GMS',t]} Tot_Inflow_ROTR['PCN','GMS',t,i1,d1]:= if Online_ROTR['PCN']=1 then
Inflow_ROTR['PCN','GMS',t,i1]+d1 else 0;
let {i1 in Inflows['GMS',t],d1 in Rel_Decision['GMS',t]} Tot_Inflow_ROTR['STC','GMS',t,i1,d1]:= if Online_ROTR['STC']=1 then
Inflow_ROTR['STC','GMS',t,i1]+Inflow_ROTR['PCN','GMS',t,i1]+d1 else 0;#+ Tot_Inflow_ROTR['PCN','GMS',t,i1,oi]  ;

let { i2 in Inflows['MCA',t],  nu in No_Units['REV',t],d2 in Rel_Decision['MCA',t]} Turbine_Release_ROTR['REV','MCA',t,i2,d2]:= if
Online_ROTR['REV']=1 then min (Abs_QT_Max['REV',t,nu],Tot_Inflow_ROTR['REV','MCA',t,i2,d2] ) else 0;
let { i1 in Inflows['GMS',t],  nu in No_Units['PCN',t],d1 in Rel_Decision['GMS',t]} Turbine_Release_ROTR['PCN','GMS',t,i1,d1]:= if
Online_ROTR['PCN']=1 then min (Abs_QT_Max['PCN',t,nu],Tot_Inflow_ROTR['PCN','GMS',t,i1,d1] ) else 0;
let { i1 in Inflows['GMS',t],  nu in No_Units['STC',t],d1 in Rel_Decision['GMS',t]} Turbine_Release_ROTR['STC','GMS',t,i1,d1]:= if
Online_ROTR['STC']=1 then min (Abs_QT_Max['STC',t,nu],Tot_Inflow_ROTR['STC','GMS',t,i1,d1] ) else 0;
```

```
let {i2 in Inflows['MCA',t], d2 in Rel_Decision['MCA',t]} Spill_ROTR['REV','MCA',t,i2,d2]:= if Online_ROTR['REV']=1 and
Tot_Inflow_ROTR['REV','MCA',t,i2,d2]> Turbine_Release_ROTR['REV','MCA',t,i2,d2]then
Tot_Inflow_ROTR['REV','MCA',t,i2,d2]-Turbine_Release_ROTR['REV','MCA',t,i2,d2] else 0;
let {i1 in Inflows['GMS',t], d1 in Rel_Decision['GMS',t]} Spill_ROTR['PCN','GMS',t,i1,d1]:= if Online_ROTR['PCN']=1 and
Tot_Inflow_ROTR['PCN','GMS',t,i1,d1]> Turbine_Release_ROTR['PCN','GMS',t,i1,d1]then
Tot_Inflow_ROTR['PCN','GMS',t,i1,d1]-Turbine_Release_ROTR['PCN','GMS',t,i1,d1] else 0;
let {i1 in Inflows['GMS',t], d1 in Rel_Decision['GMS',t]} Spill_ROTR['STC','GMS',t,i1,d1]:= if Online_ROTR['STC']=1 and
Tot_Inflow_ROTR['STC','GMS',t,i1,d1]> Turbine_Release_ROTR['STC','GMS',t,i1,d1]then
Tot_Inflow_ROTR['STC','GMS',t,i1,d1]-Turbine_Release_ROTR['STC','GMS',t,i1,d1] else 0;

display Turbine_Release, Turbine_Release_ROTR>TSI.out;

#### Calculating HK for the ROTR plants
for { v in Version}
{
commands ("HK_ROTR_"&v&".mod");
}
display HK_ROTR>HK_ROTR.out;
##### Calculating the Generation limits
for { v in Version}
{
commands ("Max_Gen_Limits_"&v&".mod");
}
#### Calculating Generation and Trade" Generation:affecting the PV function
let {p in Plants, r in Reservoirs, s in State[r,t], w in Stater[r,t,s], d in Rel_Decision[r,t], nu in No_Units[r,t]} Generation[r,t,s,w,d]:= min(
Turbine_Release [r,t,s,w,d]*HK [r,t, s,w],Max_Gen_Limits[r,t,s,w])*Days_Months[t]*24*Outage[p,t]*Availability[p,t];

let {r in Reservoirs, s in State[r,t], w in Stater[r,t,s], d in Rel_Decision[r,t], nu in No_Units[r,t]} G_OR[r,t,s,w,d]:= OR[r,t]*
Generation[r,t,s,w,d];# units are in MWh

let {p in Plants, i2 in Inflows['MCA',t],d2 in Rel_Decision['MCA',t], nu in No_Units['REV',t]} Gen_ROTR['REV','MCA',t,i2,d2]:=if
Online_ROTR['REV']=1 then min( Turbine_Release_ROTR['REV','MCA',t,i2,d2]*HK_ROTR['REV','MCA',t,i2,d2], Abs_Max_Gen_Cap['REV',t,nu])*Days_Months[t]*
24*Outage[p,t]*Availability[p,t] else if Online_ROTR['REV']=2 then Gen_REV_fixed[t] else 0;
let {p in Plants, i1 in Inflows['GMS',t],d1 in Rel_Decision['GMS',t], nu in No_Units['PCN',t]} Gen_ROTR['PCN','GMS',t,i1,d1]:=if
Online_ROTR['PCN']=1 then min( Turbine_Release_ROTR['PCN','GMS',t,i1,d1]*HK_ROTR['PCN','GMS',t,i1,d1], Abs_Max_Gen_Cap['PCN',t,nu])*Days_Months[t]*
24*Outage[p,t]*Availability[p,t] else if Online_ROTR['PCN']=2 then Gen_PCN_fixed[t] else 0;
let {p in Plants, i1 in Inflows['GMS',t],d1 in Rel_Decision['GMS',t], nu in No_Units['STC',t]} Gen_ROTR['STC','GMS',t,i1,d1]:=if
Online_ROTR['STC']=1 then min( Turbine_Release_ROTR['STC','GMS',t,i1,d1]*HK_ROTR['STC','GMS',t,i1,d1], Abs_Max_Gen_Cap['STC',t,nu])*Days_Months[t]*
24*Outage[p,t]*Availability[p,t] else if Online_ROTR['STC']=2 then Gen_STC_fixed[t] else 0;
let {p in Plants, i2 in Inflows['MCA',t],d2 in Rel_Decision['MCA',t], nu in No_Units['REV',t]} Gen_ROTR['ARD','MCA',t,i2,d2]:=if
Online_ROTR['ARD']=2 then Gen_ARD_fixed [t] else 0;

let {rp in ROTR_Plants, s1 in State['GMS',t],s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in Inflows['MCA',t], w1 in Stater['GMS',t,s1],w2 in
Stater['MCA',t,s2] , d1 in Rel_Decision['GMS',t], d2 in  Rel_Decision['MCA',t]} Tot_Gen[t,s1,s2,i1,i2,w1,w2,d1,d2]:= Generation['GMS',t,s1,w1,d1]+
Generation['MCA',t,s2,w2,d2]+IPP_Therm[t]+ Gen_ROTR['PCN','GMS',t,i1,d1]+ Gen_ROTR['STC','GMS',t,i1,d1]+ Gen_ROTR['REV','MCA',t,i2,
d2]+Gen_ROTR['ARD','MCA',t,i2, d2]; ####Units are in MWh###if Generation['GMS',t,s1,i1,w1,d1]=0 or Generation['MCA',t,s2,i2,w2,d2]=0 then 0 else
let {l in Load[t],s1 in State['GMS',t],s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in Inflows['MCA',t], w1 in Stater['GMS',t,s1],w2 in
Stater['MCA',t,s2], d1 in Rel_Decision['GMS',t], d2 in  Rel_Decision['MCA',t]} Trade[t,s1,s2,i1,i2,w1,w2,d1,d2] :=  Tot_Gen[t,s1,s2,i1,i2,w1,w2,d1,
d2]-l-Contr_Exp [t]+Contr_Imp [t] ;#if Tot_Gen[t,s1,s2,i1,i2,w1,w2,d1,d2]=0 then 0 else  Tot_Gen[t,s1,s2,i1,i2,w1,w2,d1,d2]-l-Contr_Exp [t] ;# This
is the load-Resource Balance Equation to get the Trade ( Export/ Import) values #Units are in MWh##if Tot_Gen[t,s1,s2,i1,i2,w1,w2,d1,d2]= 0 then 0
else
display Generation, Gen_ROTR,Tot_Gen> Gen.out;
##### Transition Probability Calculations#####################################
let {r in Reservoirs}Upper_Bound_State[r,t]:= last (State [r,t]);
let {r in Reservoirs}Lower_Bound_State[r,t]:= first (State [r,t]);
let {r in Reservoirs}Upper_Bound_Stater[r,t]:= max{s in State [r,t]} last (Stater[r,t,s]);
let {r in Reservoirs}Lower_Bound_Stater[r,t]:= min{s in State [r,t]} first (Stater [r,t,s]);
#display Upper_Bound_State,Lower_Bound_State,Upper_Bound_Stater,Lower_Bound_Stater>Bounds.out;
```

```
let {r in Reservoirs,s in State[r,t], i in Inflows[r,t], w in Stater[r,t,s], d in Rel_Decision[r,t]:Lower_Bound_Stater[r,t]<
w<Upper_Bound_Stater[r,t]} State_Prob[r,t,s,i,w,d] := if Hydraulic_Balance [r, t,s,i,w,d]=0  then Prob_Inflow[r,t,i]else 0;
#**##
let {r in Reservoirs,s in State[r,t], i in Inflows[r,t], w in Stater[r,t,s], d in Rel_Decision[r,t]:Lower_Bound_Stater[r,t]= w} State_Prob[r,t,s,i,
w,d] := if Hydraulic_Balance [r, t,s,i,w,d] <=0 then Prob_Inflow[r,t,i]else 0;
#**##
let {r in Reservoirs,s in State[r,t], i in Inflows[r,t], w in Stater[r,t,s], d in Rel_Decision[r,t]:w=Upper_Bound_Stater[r,t]}
State_Prob[r,t,s,i,w,d] := if Hydraulic_Balance [r, t,s,i,w,d]>=0 then Prob_Inflow[r,t,i]else 0;
# those two are constrained with Trade limits and other limits
let {l in Load[t],s1 in State['GMS',t],s2 in State['MCA',t] ,i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in Stater['GMS',t,s1],w2 in
Stater['MCA',t,s2], d1 in Rel_Decision['GMS',t], d2 in  Rel_Decision['MCA',t]} Spot_Buy[t,s1,s2,i1,i2,w1,w2,d1,d2] := if Trade[t,s1,s2,i1,i2,w1,w2,
d1,d2] < 0  then max ((-Trade_Imp_Limit[t]*Days_Months[t]*24+Contr_Imp [t]),Trade[t,s1,s2,i1,i2,w1,w2,d1,d2]) else 0;# accounts for contract
Exp/Imp in the tie limits.
let {l in Load[t],s1 in State['GMS',t],s2 in State['MCA',t] ,i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in Stater['GMS',t,s1],w2 in
Stater['MCA',t,s2], d1 in Rel_Decision['GMS',t], d2 in  Rel_Decision['MCA',t]} Spot_Sell[t,s1,s2,i1,i2,w1,w2,d1,d2] := if Trade[t,s1,s2,i1,i2, w1,
w2,d1,d2] >0  then min ((Trade_Exp_Limit[t]*Days_Months[t]*24-Contr_Exp [t]),Trade[t,s1,s2,i1,i2,w1,w2,d1,d2]) else 0;#  accounts for contract
Exp/Imp in the tie limits.
let {l in Load[t],s1 in State['GMS',t],s2 in State['MCA',t] ,i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in Stater['GMS',t,s1],w2 in
Stater['MCA',t,s2], d1 in Rel_Decision['GMS',t], d2 in  Rel_Decision['MCA',t]} Energy_Curt_Tie_Imp[t,s1,s2,i1,i2,w1,w2,d1,d2] := if Trade[t,s1,s2,
i1,i2, w1,w2,d1,d2] < Spot_Buy [t,s1,s2,i1,i2,w1,w2,d1,d2]< 0 then   Trade[t,s1,s2,i1,i2,w1,w2,d1,d2]-Spot_Buy[t,s1,s2,i1,i2,w1,w2,d1,d2]  else 0;#
calculates energy curtialed from Exp due to tie limits.
let {l in Load[t],s1 in State['GMS',t],s2 in State['MCA',t] ,i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in Stater['GMS',t,s1],w2 in
Stater['MCA',t,s2], d1 in Rel_Decision['GMS',t], d2 in  Rel_Decision['MCA',t]} Energy_Curt_Tie_Exp[t,s1,s2,i1,i2,w1,w2,d1,d2] := if Trade[t,s1,s2,
i1,i2, w1,w2,d1,d2] > Spot_Sell[t,s1,s2,i1,i2,w1,w2,d1,d2]> 0 then   Trade[t,s1,s2,i1,i2,w1,w2,d1,d2]-Spot_Sell[t,s1,s2,i1,i2,w1,w2,d1,d2] else 0;#
calculates energy curtialed from Exp due to tie limits. this amount should be backed off by the plant or spilled

let { s1 in State['GMS',t],s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in Stater['GMS',t,s1],w2 in Stater['MCA',t,s2], d1
in Rel_Decision['GMS',t], d2 in Rel_Decision['MCA',t]} Tot_Spilled_Energy [t,s1,s2,i1,i2,w1,w2,d1,d2]:= (Spill['GMS',t,s1,w1,d1]*HK ['GMS',t,
s1,w1]+Spill['MCA',t,s2,w2,d2]*HK ['MCA',t, s2,w2]+ Spill_ROTR['REV','MCA',t,i2,d2]*HK_ROTR['REV','MCA',t,i2,d2]+ Spill_ROTR['PCN','GMS',t,i1,d1]*
HK_ROTR['PCN','GMS',t,i1,d1]+Spill_ROTR['STC','GMS',t,i1,d1]*HK_ROTR['STC','GMS',t,i1,d1])*Days_Months[t]*24-Energy_Curt_Tie_Imp[t,s1,s2,i1,i2,w1,
w2,d1,d2]+ Energy_Curt_Tie_Exp[t,s1,s2,i1,i2,w1,w2,d1,d2] ;

let Max_Mthly_Spot_Buy_GW[t]:= -min{s1 in State['GMS',t], s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in Stater['GMS',t,
s1],w2 in Stater['MCA',t,s2], d1 in Rel_Decision['GMS',t], d2 in Rel_Decision['MCA',t]}Spot_Buy[t,s1,s2,i1,i2,w1,w2,d1,d2]/1000;
let Min_Mthly_Spot_Buy_GW[t]:= max{s1 in State['GMS',t], s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in
Stater['GMS',t,s1],w2 in Stater['MCA',t,s2], d1 in Rel_Decision['GMS',t], d2 in Rel_Decision['MCA',t]}Spot_Buy[t,s1,s2,i1,i2,w1,w2,d1,d2]/1000;
let Max_Mthly_Spot_Sell_GW[t]:= max{s1 in State['GMS',t], s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in Stater['GMS',t,
s1],w2 in Stater['MCA',t,s2], d1 in Rel_Decision['GMS',t], d2 in Rel_Decision['MCA',t]}Spot_Sell[t,s1,s2,i1,i2,w1,w2,d1,d2]/1000;
let Min_Mthly_Spot_Sell_GW[t]:= min{s1 in State['GMS',t], s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in Stater['GMS',t,
s1],w2 in Stater['MCA',t,s2], d1 in Rel_Decision['GMS',t], d2 in Rel_Decision['MCA',t]}Spot_Sell[t,s1,s2,i1,i2,w1,w2,d1,d2]/1000;

let {l in Load[t],s1 in State['GMS',t],s2 in State['MCA',t] , i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in Stater['GMS',t,s1],w2 in
Stater['MCA',t,s2], d1 in Rel_Decision['GMS',t], d2 in  Rel_Decision['MCA',t]} Trans_Prob[t,s1,s2,i1,i2,w1,w2,d1,d2] :=  State_Prob['GMS',t,s1,i1,
w1,d1]*State_Prob['MCA',t,s2,i2,w2,d2] ;
let {l in Load[t],s1 in State['GMS',t],s2 in State['MCA',t],w1 in Stater['GMS',t,s1],w2 in Stater['MCA',t,s2], d1 in Rel_Decision['GMS',t], d2 in
Rel_Decision['MCA',t]} Trans_Prob_S[t,s1,s2,w1,w2,d1,d2] := sum { i1 in Inflows['GMS',t],i2 in Inflows['MCA',t]} Trans_Prob[t,s1,s2,i1,i2,w1,w2,d1,
d2];
let {l in Load[t],s1 in State['GMS',t],s2 in State['MCA',t], d1 in Rel_Decision['GMS',t], d2 in  Rel_Decision['MCA',t]} Trans_Prob_S_S[t,s1,s2,d1,
d2] := sum { w1 in Stater['GMS',t,s1],w2 in Stater['MCA',t,s2]}  Trans_Prob_S[t,s1,s2,w1,w2,d1,d2];
#check of Hydraulic Balnace and Load-Resource Balance and Feasibility
let {l in Load[t],s1 in State['GMS',t],s2 in State['MCA',t],i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in Stater['GMS',t,s1],w2 in
Stater['MCA',t,s2], d1 in Rel_Decision['GMS',t], d2 in Rel_Decision['MCA',t]}Load_Reseource_Balance [t, s1,s2,i1,i2,w1,w2,d1,d2]:=
Generation['GMS',t,s1,w1,d1]+Generation['MCA',t,s2,w2,d2]+IPP_Therm[t]+ Gen_ROTR['PCN','GMS',t,i1,d1]+
Gen_ROTR['STC','GMS',t,i1,d1]+Gen_ROTR['REV','MCA',t,i2,d2]+Gen_ROTR['ARD','MCA',t,i2,d2]-Contr_Exp [t]+ Contr_Imp [t]- Spot_Buy [t,s1,s2,i1,i2,w1,
w2,d1,d2]- Spot_Sell [t,s1,s2,i1,i2,w1,w2,d1,d2]-l;
```

```
let {l in Load[t],s1 in State['GMS',t],s2 in State['MCA',t] ,  d1 in Rel_Decision['GMS',t], d2 in  Rel_Decision['MCA',t]} EX_Exp_Rev[t,s1,s2,d1,d2]
:=  sum { i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in Stater['GMS',t,s1],w2 in Stater['MCA',t,s2]} Spot_Sell[t,s1,s2,i1,i2,w1,w2,d1,d2] *
Trans_Prob[t,s1,s2,i1,i2,w1,w2,d1,d2] *Exp_Price[t,i1,i2]/1000000 ; # Expected spot market sell revenue
let {r in Reservoirs, l in Load[t] , s1 in State['GMS',t],s2 in State['MCA',t] , d1 in Rel_Decision['GMS',t], d2 in  Rel_Decision['MCA',t]}
Cont_Rev[t,s1,s2,d1,d2] :=  sum { i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in Stater['GMS',t,s1],w2 in Stater['MCA',t,s2]} (Contr_Exp [t]-
Contr_Imp [t]) *Trans_Prob[t,s1,s2,i1,i2,w1,w2,d1,d2] * Cont_Price[t,i1,i2]/1000000  ; # Contracts Revenue
#let {l in Load[t],s1 in State['GMS',t],s2 in State['MCA',t] , d1 in Rel_Decision['GMS',t], d2 in  Rel_Decision['MCA',t]} EX_Penalty[t,s1,s2,d1,d2]
:=  sum { i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in Stater['GMS',t,s1],w2 in Stater['MCA',t,s2]} Tot_Penalty[t,s1,s2,i1,i2,w1,w2,d1,d2] *
Trans_Prob[t,s1,s2,i1,i2,w1,w2,d1,d2] *Imp_Price[t,i1,i2]*10/1000000  ; # Expected spot market sell revenue

let { s1 in State['GMS',t],s2 in State['MCA',t], d1 in Rel_Decision['GMS',t], d2 in Rel_Decision['MCA',t]}
Ex_Penalty_Spilled_Energy[t,s1,s2,d1,d2]:= sum { i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in Stater['GMS',t,s1],w2 in Stater['MCA',t,s2]}
Tot_Spilled_Energy [t,s1,s2,i1,i2,w1,w2,d1,d2]*Trans_Prob[t,s1,s2,i1,i2,w1,w2,d1,d2] *(-Imp_Price[t,i1,i2]/1000000);

let {l in Load[t] , s1 in State['GMS',t],s2 in State['MCA',t] , d1 in Rel_Decision['GMS',t], d2 in  Rel_Decision['MCA',t]} Policy_Income [t,s1,s2,
d1,d2] := Cont_Rev[t,s1,s2,d1,d2] + EX_Imp_Cost[t,s1,s2,d1,d2]+ EX_Exp_Rev[t,s1,s2,d1,d2]+ l* Load_Reward/1000000 + if Account_Spill_Penalty=1 then
Ex_Penalty_Spilled_Energy[t,s1,s2,d1,d2]*Penalty_Ratio else 0;
########################################################
reset data               Term_W    Plant_Release  Spill    Hydraulic_Balance      Tot_Gen Trade    Trade   State_Prob      EX_Imp_Cost      EX_Exp_Rev
Cont_Rev                 Plant_Release    Spill; #Trans_Prob, Spot_Buy    Spot_Sell Generation
print "Finished Basics_Month: "&t&"";
printf "%12.0f",t>> Set_Stages.use;
}# looping over the Months
print ";" > ("Set_Stages.use");
#######################Declaring the Forebay parameters and calculating them
param FBi{r in Reservoirs, t in Months,  State[r,t]}; # Initial forebay
param FBf{r in Reservoirs, t in Months,s in State[r,t],Stater[r,t,s]};# Final forebay
###############################################################################
for { v in Version}
{
commands ("FB_"&v&".mod");
}
###############################################################################
purge   HK               Tot_Gen Trade           State_Prob      EX_Imp_Cost      EX_Exp_Rev      Cont_Rev               Plant_Release    Spill ;
#Trans_Prob, Spot_Sell, Spot_Buy         Term_W  Generation
purge Trade_Exp_Limit    Trade_Imp_Limit Prob_Inflow         Min_Gen_Limits          QT_Min  QP_Max  QP_Min  Outage  IPP_Therm
Contr_Trans;#Inflows , Max_Gen_Limits, QT_Max
################ ITERATION ########################### Iterating over the PV Total Difference, to get the optimum policy PROCEDURE
for { v in Version}
{
commands ("Value_Iteration_"&v&".mod");
}
###############################################################################
shell 'md Outputdata';
cd Outputdata;

####Declaring the stages' dates
print "set Months:=" >("Stage_Name.out");
for { t in Months}
{
print member(t,Stage_Name)>("Stage_Name.out");
}
print ";" > ("Stage_Name.out");
printf "\n" >("Stage_Name.out");
display Months>("Stage_Name.out");
display Stage_Name >("Stage_Name.out");
##########################################################
```

```
display PV_Max > ("PV_Final.out");
display PV>PV.out;
display Optimum_Policy_GMS, Optimum_Policy_MCA, PV_Max, Optimum_Spot_Export_GW,Optimum_Spot_Import_GW >Optimum_Policy.out;
############################################### End of the Iterations
##CALCULATING THE MARGINAL VALUE OF WATER/ENERGY FOR EACH RESERVOIR
param MVW_GMS{ t in Months, s1 in State['GMS',t], s2 in State['MCA',t]} ;#Marginal value of water..units are in $/cms
param MVW_MCA{ t in Months, s1 in State['GMS',t],s2 in State['MCA',t]} ;#Marginal value of water..units are in $/cms
param MVE_GMS{ t in Months, s1 in State['GMS',t], s2 in State['MCA',t]} ;#Marginal value of water..units are in $/MWh
param MVE_MCA{ t in Months, s1 in State['GMS',t],s2 in State['MCA',t]} ;#Marginal value of water..units are in $/MWh

param d1{ t in Months};
param d2{ t in Months};
param HK_rough{ r in Reservoirs, t in Months,s in State [r,t]};# a rough estimate for HK to use in MVW calculations;

cd..;
#### Calculating HK_rough that is used to calculate the MV of energy
for { v in Version}
{
commands ("HK_rough_"&v&".mod");
}

cd Outputdata;

let { r in Reservoirs, t in Months} Delta_States[r,t] := (n*Inflow_Step[r,t]* Days_Months_Dflt);# units are in cms.day, can use also "floor" and
"ceil" functions instead of round
let{t in Months} d1[t]:=Delta_States['GMS',t];
let{t in Months}d2[t]:=Delta_States['MCA',t];

for{  t in Months,s2 in State['MCA',t]}
{
let { s1 in State['GMS',t]}MVW_GMS[t,s1,s2]:= if Run_Single_Res['GMS']=0 and s1< last (State['GMS',t]) then (PV_Fin[t,s1+d1[t],s2]-
PV_Fin[t,s1,s2])*1e6/ (Delta_States['GMS',t]*24) else if Run_Single_Res['GMS']=0 then MVW_GMS[t,s1-d1[t],s2] else 0;
let { s1 in State['GMS',t]}MVE_GMS[t,s1,s2]:= if Run_Single_Res['GMS']=0 and s1< last (State['GMS',t]) then (PV_Fin[t,s1+d1[t],s2]-
PV_Fin[t,s1,s2])*1e6/ (Delta_States['GMS',t]*HK_rough ['GMS',t, s1]*24) else if Run_Single_Res['GMS']=0 then MVE_GMS[t,s1-d1[t],s2] else 0;

}

for{ t in Months, s1 in State['GMS',t]}
{
let { s2 in State['MCA',t]}MVW_MCA[t,s1,s2]:= if Run_Single_Res['MCA']=0 and s2< last (State['MCA',t]) then (PV_Fin[t,s1,s2+d2[t]]-
PV_Fin[t,s1,s2])*1e6/ (Delta_States['MCA',t]*24) else if Run_Single_Res['MCA']=0 then MVW_MCA[t,s1,s2-d2[t]];
let { s2 in State['MCA',t]}MVE_MCA[t,s1,s2]:= if Run_Single_Res['MCA']=0 and s2< last (State['MCA',t]) then (PV_Fin[t,s1,s2+d2[t]]-
PV_Fin[t,s1,s2])*1e6/ (Delta_States['MCA',t]*HK_rough ['MCA',t, s2]*24) else if Run_Single_Res['MCA']=0 then MVE_MCA[t,s1,s2-d2[t]] else 0;
}

display MVW_GMS > MVW_GMS.out;
display MVW_MCA > MVW_MCA.out;
display MVE_GMS > MVE_GMS.out;
display MVE_MCA > MVE_MCA.out;
display Max_Mthly_Spot_Buy_GW, Min_Mthly_Spot_Buy_GW, Max_Mthly_Spot_Sell_GW, Min_Mthly_Spot_Sell_GW> Trade_Stats.out;
display FBi,FBf>FB.out;
display Max_Gen_Limits, QT_Max, Turbine_Release> Turbine_Limits.out;
display HK_rough>HK_rough.out;

print" Month     GMS_Storage     MCA_Storage     Value_Function,M$       MVE_GMS,$/MWh   MVE_MCA,$/MWh" > Final_Results_Listed.out;
print" Month     GMS_Storage     MCA_Storage     Value_Function,M$       MVE_GMS,$/MWh   MVE_MCA,$/MWh" > Final_Results_Listed.csv;
```

```
print" Month    GMS_Storage     MCA_Storage     Value_Function,M$       MVE_GMS,$/MWh   MVE_MCA,$/MWh" > Final_Results_Listed.out;
print" Month    GMS_Storage     MCA_Storage     Value_Function,M$       MVE_GMS,$/MWh   MVE_MCA,$/MWh" > Final_Results_Listed.csv;

for {t in Months,s1 in State['GMS',t],s2 in State['MCA',t]}
{
printf "%4.0f   %6.0f   %6.0f   %6.0f   %6.2f   %6.2f",t,s1,s2,PV_Max[t,s1,s2], MVE_GMS[t,s1,s2], MVE_MCA[t,s1,s2] > Final_Results_Listed.out;
printf "\n" >Final_Results_Listed.out;
printf "%4.0f   %6.0f   %6.0f   %6.0f   %6.2f   %6.2f",t,s1,s2,PV_Max[t,s1,s2], MVE_GMS[t,s1,s2], MVE_MCA[t,s1,s2] > Final_Results_Listed.csv;
printf "\n" >Final_Results_Listed.csv;

}

cd..;
print"###########################################################################################";
print"###########################################################################################";
for { v in Version}
{
print"##############  SDPOM6R Model, Version "&v&" is finished , Thank You  #################";
}
print"###########################################################################################";
print"###########################################################################################";

######### Renaming the output folder to include Version name, Number of time steps, Current date and time######################
param Time_hr:=if floor((time()-1356998400)/3600) <=12 then floor((time()-1356998400)/3600) else floor((time()-1356998400)/3600) -12;## returns the
current time (in seconds since 00:00:00 1 Jan. 1970 GMT)
param Time_min:=if floor((time()-1356998400)/3600) <=12 then round(((time()-1356998400)/3600-Time_hr)*60,0) else round(((time()-1356998400)/3600-(
Time_hr+12))*60,0);
param AM_PM:=if floor((time()-1356998400)/3600) <12 then "AM" else "PM";
param CTIME= ctime()symbolic;
data;
include Set_Stages.use;
for { v in Version}
{
if floor((time()-1349334001)/3600) <12
then
print"shell' ren Outputdata Ouputdata_Vr."&v&"_Stagesfrom"&Start_Months&"to"&End_Months&"_"&Inflow_Step['GMS',first(Stages)]&"cms.forGMS_"&
Inflow_Step['MCA',first(Stages)]&"cms.forMCA@(Hr."&Time_hr&"~Min."&Time_min&".AM)';" > RenameFolder.mod;
else
print"shell' ren Outputdata Ouputdata_Vr."&v&"_Stagesfrom"&Start_Months&"to"&End_Months&"_"&Inflow_Step['GMS',first(Stages)]&"cms.forGMS_"&
Inflow_Step['MCA',first(Stages)]&"cms.forMCA@(Hr."&Time_hr&"~Min."&Time_min&".PM)';" > RenameFolder.mod;
}
display ctime();
commands ("RenameFolder.mod");
print"This run ended at "&CTIME&""> Run_Times_Stats.amr;
###################################################

reset;
############################### END OF THE CODE#####################################################
####################################################################################################
close;
```

A.1.2.2. Computation Details

Table 5: Calculations Details in the SDP Model

Step Number	Parameter Calculated	Function
1	Total_Sys_Inflow	Sums up the inflows to the plants included in the model including the storage and non-storage plants.
2	Exp_Price, Imp_Price, Cont_Price	Calculate the prices for import, export and contract prices using regression equations that relate the energy prices to the total inflow of the system. For a given inflows combination, the difference between import and export prices is fixed while the contract prices are the average of them.
3	Term_W	For each point of starting storage state, release decision and inflow, this parameter calculates the actual terminal storage that corresponding to those values.
4	Turbine_Release	Calculates the turbine flow for the storage plants (i.e. GM Shrum and Mica) which is capped by the release decision and the maximum turbine flow (QT_Max).
5	Spill	The difference between the release decision and the turbine release.
6	Hydraulic_Balance	The difference between the actual terminal storage and the discretized terminal storage. It is mainly used to assign the state probability and the transition probability.
7	Tot_Inflow_ROTR	Sums up the natural inflow to a given run-of-the-river (ROTR) plant to the flows coming from all the plants at the upstream of this plant.

Step Number	Parameter Calculated	Function
8	Turbine_Release_ROTR	Calculates the turbine flow for the ROTR plants (i.e. PCN, REV and STC) from the total flow to each plant capped by absolute maximum turbine flow (Abs_QT_Max).
9	Spill_ROTR	The difference between the ROTR total flow and the ROTR turbine release.
10	Generation	The storage plants generation calculated as the turbine release times the HK capped by the Max_Gen_Limits and the Abs_Max_Gen_Capacity considering both of the outage and the availability factors.
11	Gen_ROTR	The ROTR plants generation calculated as the turbine release times the HK capped by the Abs_Max_Gen_Capacity considering both of the outage and the availability factors.
12	Tot_Gen	Total energy generated from the system in MWhr including the storage plants, ROTR plants, IPPS, thermal plants and fixed generation from the non-optimized plants such as ARD.
13	Trade	Is the amount of energy surplus or deficit after satisfying the long-term contracts and meeting the domestic load.
14	State_Prob	The state transition probability from a given starting state to a given terminal states for each storage reservoir using the logic discussed earlier in the document. The size of the resulting matrix equals the number of stages times number of starting states times number of inflows times the number of ending states times the number of release decisions.

Step Number	Parameter Calculated	Function
15	Spot_Buy, Spot_Sell	Is the trade but capped by the export/import transmission limits
16	Trans_Prob	Is the State_Prob of a storage reservoir times the State_Prob of the other reservoir. The size of the matrix equals the size of State_Prob matrix of GM Shrum times the size of the State_Prob matrix of Mica.
17	Trans_Prob_S	Is the summation of the Trans_Prob matrix over different combination of inflows to the storage plants
18	Load_Reseource_Balance	Is the energy balance of the system calculated as the total generation from the system minus the domestic load and the trade.
19	EX_Imp_Cost	Expected import cost calculated as Spot_Buy times the Trans_Prob times the Imp_Price and summing the outcome over different combinations of inflows and terminal states for the storage reservoirs.
20	EX_Exp_Rev	Expected export revenue calculated as Spot_Sell times the Trans_Prob times the Exp_Price and summing the outcome over different combinations of inflows and terminal states for the storage reservoirs.
21	Cont_Rev	Expected revenue/cost related to the long-term contracts calculated as contract trade times the Trans_Prob times the Cont_Price and summing the outcome over different combinations of inflows and terminal states for the storage reservoirs.
22	Policy_Income	Is the current total expected revenue/cost for a given time step for different combinations of release decisions and starting states of the storage reservoirs,

Step Number	Parameter Calculated	Function
		calculated as the summation of EX_Imp_Cost, EX_Exp_Rev and Cont_Rev.
23	Value iteration calculations	Calculates the value of water in storage along a pre-determined time horizon considering a pre-defined tolerance for convergence. The details of the procedure are discussed earlier in the document.
24	Optimum_Policy_GMS, Optimum_Policy_MCA	Parameters that pick the release decisions for the storage plants that gives the maximum value of water.
25	MVW_GMS, MVW_MCA	Is the marginal value of water which is the slope of the water value function for the storage reservoir.
26	MVE_GMS, MVE_MCA	Is the marginal value of energy in $/MWhr calculated as the marginal value of water for GMS/ MCA divided by the total HK of the plants on the Peace River system (GMS, PCN, STC)/ the Columbia River system (MCA, REV).

A.1.3. Value Iteration Model

A.1.3.1. Code

```
repeat while ( PV_diff_total > .25)
{
let iter := iter + 1;
display iter;
for {t in last (Months)..first (Months) by -1 }
{
update data;
let { s1 in State['GMS',t],s2 in State['MCA',t]} PV_Fin[t,s1,s2] := if t=End_Months then PV_Max[Start_Months,s1,s2] else PV_Max[t+1,s1,s2] ;
let { s1 in State['GMS',t],s2 in State['MCA',t], x1 in Rel_Decision['GMS',t],x2 in  Rel_Decision['MCA',t]} PV[t,s1,s2,x1, x2] := Policy_Income[t,s1,s2,x1,
x2]+(Gama)*Trans_Prob_S_S[t,s1,s2,x1,x2] * PV_Fin[t,s1,s2];#sum {w1 in Stater['GMS',t,s1],w2 in Stater['MCA',t,s2]} Trans_Prob_S[t, s1,s2,w1,w2,x1,
x2]*PV_Fin[t,s1,s2];    ### Trans_Prob_S_S[t,s1,s2,d1,d2] *PV_Fin[t,s1,s2]

update data;
let { s1 in State['GMS',t],s2 in State['MCA',t]}PV_Max [t,s1,s2] := max { x1 in Rel_Decision['GMS',t], x2 in  Rel_Decision['MCA',t]} PV[t,s1,s2,x1, x2] ;
update data;
let { s1 in State['GMS',t],s2 in State['MCA',t]}PV_diff[t,s1,s2] := PV_Max[t,s1,s2]-PV_Fin[t,s1,s2];
update data;
}
let PV_diff_total:= abs(sum {t in Months,s1 in State['GMS',t], s2 in State['MCA',t]} PV_diff[t,s1,s2]);
}
#### Retrieving the optimum policy ########################
for{ t in Months, s1 in State['GMS',t],s2 in State['MCA',t], x1 in Rel_Decision['GMS',t], x2 in  Rel_Decision['MCA',t]}
{
if PV[t,s1,s2,x1, x2]= PV_Max [t,s1,s2]
then{
let Optimum_Policy_GMS[t,s1,s2]:= x1;
let Optimum_Policy_MCA [t,s1,s2]:= x2;# union x2 else 0;
let Optimum_Spot_Export_GW[t,s1,s2]:=sum { i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in Stater['GMS',t,s1],w2 in Stater['MCA',t,s2]}Spot_Sell[t,s1,
s2,i1,i2,w1,w2,x1,x2]* Trans_Prob[t,s1,s2,i1,i2,w1,w2,x1,x2] /1000;
let Optimum_Spot_Import_GW[t,s1,s2]:=sum { i1 in Inflows['GMS',t],i2 in Inflows['MCA',t],w1 in Stater['GMS',t,s1],w2 in
Stater['MCA',t,s2]}Spot_Buy[t,s1,s2,i1,i2,w1,w2,x1,x2]* Trans_Prob[t,s1,s2,i1,i2,w1,w2,x1,x2]/1000;
}
}
```

A.1.4. Other Modules

A.1.4.1. Code

1. HK Model

```
## This is the model that calculates HK for the plants as a fucntion of starting and ending storages.

let{ t in Months,s1 in State ['GMS',t], w1 in Stater['GMS',t,s1]} HK ['GMS',t, s1,w1] := -0.0000000000001*((s1+w1)/2)^2 + 0.000000645*((s1+w1)/2)+
1.2083; #  equation modified on July 17th 2012
let{ t in Months,s2 in State ['MCA',t], w2 in Stater['MCA',t,s2]} HK ['MCA',t, s2,w2] :=  -0.0000000000038*((s2+w2)/2)^2 + 0.00000395*((s2+w2)/2) +
0.7155;# equation modified on July 17th 2012 (generally they are lower values now for lower storages and almost the same for high storages)#MCA
average 1.405   max 1.635       min 1.153
let{ t in Months,s3 in State ['ARD',t], w3 in Stater['ARD',t,s2]} HK ['ARD',t, s2,w2] := 0.0641*log((s3+w3)/2) - 0.565; # ARD    average 0.113   max
0.171   min 0.042## needs revision
#let{ t in Months,s4 in State ['PCN',t], w4 in Stater['PCN',t,s2]} HK ['PCN',t, s2,w2] := min(.3612,max(0.3335,0.00009*((s4+w4)/2) + 0.115));#PCN
average 0.346   max 0.361       min 0.334
#let{ t in Months,s5 in State ['REV',t], w5 in Stater['REV',t,s2]} HK ['REV',t, s2,w2] := 0.0000079*((s5+w5)/2)+ 0.7168;#REV    average 1.189   max
1.208   min 1.170
```

2. HK ROTR Model

```
## This file caculates the HK for the Run-of-the-river projects.
## As per the discussion with Ziad, HK_ROTR equation were modified so as to get higher values for low releases. January 18th, 2013

let{i1 in Inflows['GMS',t], d1 in Rel_Decision['GMS',t]}  HK_ROTR['PCN','GMS',t,i1,d1]:= min(0.343,max(0.343/2, 0.00000000247*(Turbine_Release_ROTR
['PCN','GMS',t,i1,d1])^2.4580));
let{i2 in Inflows['MCA',t], d2 in Rel_Decision['MCA',t]}  HK_ROTR['REV','MCA',t,i2,d2]:= min(1.140,max(1.140/2, 0.00000708000*(Turbine_Release_ROTR
['REV','MCA',t,i2,d2])^1.5227));
let{i1 in Inflows['GMS',t], d1 in Rel_Decision['GMS',t]}  HK_ROTR['STC','GMS',t,i1,d1]:= min(0.430,max(0.430/2, 0.00001050000*(Turbine_Release_ROTR
['STC','GMS',t,i1,d1])^1.3550));

display HK_ROTR>HK.out;
```

3. HK Rough Model

```
## this file is last modified on January 4th 2013
## Calculates the HK as a functuion of the starting state only.
let{t in Months,s1 in State ['GMS',t]} HK_rough ['GMS',t, s1] :=
     if Online_ROTR['PCN']=1 and Online_ROTR['STC']=1 then -0.0000000000001*(s1)^2 + 0.000000645*(s1)+ 1.2083 + 0.343+ 0.430
else if Online_ROTR['PCN']=1 and Online_ROTR['STC']=0 then -0.0000000000001*(s1)^2 + 0.000000645*(s1)+ 1.2083 + 0.343
else if Online_ROTR['PCN']=0 and Online_ROTR['STC']=1 then -0.0000000000001*(s1)^2 + 0.000000645*(s1)+ 1.2083 + 0.430
else                                                       -0.0000000000001*(s1)^2 + 0.000000645*(s1)+ 1.2083 ;## the last two constants are the
max HKs for PCN and STC respectively# modified July17th 2012 -0.000000000000000002*(s1)^3 + 0.00000000000116*(s1)^2 + 0.0000003*(s1) + 1.2728;

let{t in Months,s2 in State ['MCA',t]} HK_rough ['MCA',t, s2] :=
     if Online_ROTR['REV']=1 then -0.0000000000038*(s2)^2 + 0.00000395*(s2) + 0.7155+1.14
else                              -0.0000000000038*(s2)^2 + 0.00000395*(s2) + 0.7155; # the last constant is the max HK for REV# modified August
7th, modified July17th 2012 3.43E-18*(s2)^3 - 0.0000000000082*(s2)^2 + 0.000004*(s2) + 0.9573;

#let{t in Months,s3 in State ['ARD',t]} HK_rough ['ARD',t, s3] :=0.0641*log(s3) - 0.565;
#let{t in Months,s4 in State ['PCN',t]} HK_rough ['PCN',t, s4] := min(.3612,max(0.3335,0.00009*(s4) + 0.115));
#let{t in Months,s5 in State ['REV',t]} HK_rough ['REV',t, s5] := 0.0000079*(s5)+ 0.7168;
#display HK_rough> HK_rough.out;
```

4. Inflow ROTR Model

```
## Calculates the inflow to the ROTR plants as a function of the inflows to Williston and Kinbasket
## The source is: J:\PS\Power Planning\00_CRT 2014 Studies\Phase 2\OperatingStudy\TTYcontinues135C\HYSIM\HYSIM Data File20120517.xls
## Note that the flow numbers in this file is the total flows to a given plant i.e PCN flows are from GMS plus all the other natural
inflows...etc. all of this had to be compiled to come up with approximate numbers for the natural inflow only.
## modified December 2012 to eliminate any -ve inflow generated from the regression equations
## this file is last modified on January 4th 2013

#### PCN:

for {t in Months,i1 in Inflows['GMS',t]}
{
if Online_ROTR['PCN']=1
then

let Inflow_ROTR['PCN','GMS',t, i1]:=

     if i1=0                                    then        0
else if t= 1    or t =13 or t = 25 or t =37 then        max(0, 0.0137*i1 - 0.2997)
else if t= 2    or t =14 or t = 26 or t =38 then        max(0, 0.0102*i1 + 0.8020)
else if t= 3    or t =15 or t = 27 or t =39 then        max(0, 0.0090*i1 + 0.3041)
else if t= 4    or t =16 or t = 28 or t =40 then        max(0, 0.0010*i1 + 2.1384)
else if t= 5    or t =17 or t = 29 or t =41 then        max(0, 0.0049*i1 + 0.9278)
else if t= 6    or t =18 or t = 30 or t =42 then        max(0, 0.0098*i1 - 0.1398)
else if t= 7    or t =19 or t = 31 or t =43 then        max(0, 0.0124*i1 - 0.0037)
else if t= 8    or t =20 or t = 32 or t =44 then        max(0, 0.0098*i1 + 15.299)
else if t= 9    or t =21 or t = 33 or t =45 then        max(0, 0.0125*i1 + 1.5336)
else if t= 10   or t =22 or t = 34 or t =46 then        max(0, 0.0080*i1 + 6.1454)
else if t= 11   or t =23 or t = 35 or t =47 then        max(0, 0.0120*i1 - 2.1790)
else                                                    max(0, 0.0110*i1 - 0.4526);

else let Inflow_ROTR['PCN','GMS',t, i1]:=0;

}

#### Site_C:

for {t in Months,i1 in Inflows['GMS',t]}
{
if Online_ROTR['STC']=1

then
let Inflow_ROTR['STC','GMS',t, i1]:=

     if i1=0                                    then        0
else if t= 1    or t =13 or t = 25 or t =37 then        max(0, 0.03840*i1 + 41.536)
else if t= 2    or t =14 or t = 26 or t =38 then        max(0, 0.07970*i1 + 7.9809)
else if t= 3    or t =15 or t = 27 or t =39 then        max(0, 0.12820*i1 - 16.412)
else if t= 4    or t =16 or t = 28 or t =40 then        max(0, 0.00370*i1 + 33.269) #this one is created and tweaked manually;  original is :
-0.0370*i1 + 33.269 but it generates -ve inflows
else if t= 5    or t =17 or t = 29 or t =41 then        max(0, 0.00290*i1 + 16.158)
else if t= 6    or t =18 or t = 30 or t =42 then        max(0, 0.0089*i1 + 18.5430) #this one is created and tweaked manually;  original is :
-0.0089*i1 + 18.543 but it generates -ve inflows
else if t= 7    or t =19 or t = 31 or t =43 then        max(0, 0.03690*i1 + 34.550)
else if t= 8    or t =20 or t = 32 or t =44 then        max(0, 0.04710*i1 + 134.79)
else if t= 9    or t =21 or t = 33 or t =45 then        max(0, 0.04710*i1 + 134.79)
else if t= 10   or t =22 or t = 34 or t =46 then        max(0, 0.09790*i1 + 86.725)
else if t= 11   or t =23 or t = 35 or t =47 then        max(0, 0.33333*i1 - 179.29)
else                                                    max(0, 0.07430*i1 + 32.576);
```

5. Maximum Generation Limits Model

```
#Calculates the genration limits as a function of number of units and starting and ending storages

## Modified Janaury 28th , 2013...
## GMS...
let {s in State['GMS',t], w in Stater['GMS',t,s], nu in No_Units['GMS',t]} Max_Gen_Limits ['GMS',t,s,w]:=
     if s <= Max_Gen_storage_Pt['GMS',t, nu] and w <= Max_Gen_storage_Pt['GMS',t, nu] then     min(-0.000000003*((s+w)/2)^2+0.00306*((s+w)/2)+
     2281.8,Abs_Max_Gen_Cap['GMS',t,nu])
else if s >  Max_Gen_storage_Pt['GMS',t, nu] and w >  Max_Gen_storage_Pt['GMS',t, nu] then     Abs_Max_Gen_Cap['GMS',t,nu]
else if s >  Max_Gen_storage_Pt['GMS',t, nu] and w <= Max_Gen_storage_Pt['GMS',t, nu] then     (min(-0.000000003*w^2+0.00306*w+2281.8,
Abs_Max_Gen_Cap['GMS',t,nu]) +Abs_Max_Gen_Cap['GMS',t,nu])/2
else                                                                                            (min(-0.000000003*s^2+0.00306*s+2281.8,
Abs_Max_Gen_Cap['GMS',t,nu]) +Abs_Max_Gen_Cap['GMS',t,nu])/2;# GMS 10 units## updated January 28th, 2013

## MCA...
let {s in State['MCA',t], w in Stater['MCA',t,s], nu in No_Units['MCA',t]} Max_Gen_Limits ['MCA',t,s,w]:=

     if nu =4

then if (s+w)/2 <= Max_Gen_storage_Pt['MCA',t, nu] then min(-0.000000012*((s+w)/2)^2 + 0.0092*((s+w)/2) + 236.74, Abs_Max_Gen_Cap['MCA',t,nu]) else
Abs_Max_Gen_Cap['MCA',t,nu]


else if nu=5

then if (s+w)/2 <= Max_Gen_storage_Pt['MCA', t,nu] then min(-0.000000019*((s+w)/2)^2+0.0126*((s+w)/2)+246.85,Abs_Max_Gen_Cap['MCA',t,nu])  else
Abs_Max_Gen_Cap['MCA',t,nu]

else if nu=6

then if (s+w)/2 <= Max_Gen_storage_Pt['MCA',t, nu] then min (-0.000000022*((s+w)/2)^2 + 0.0153*((s+w)/2)+ 265.88,Abs_Max_Gen_Cap['MCA',t,nu]) else
Abs_Max_Gen_Cap['MCA',t,nu];# MCA 4,5,6 units# updated Janaury 28th, 2013
```

6. Maximum Turbine Limits Model

```
# Calculates turbine limits as a function of number of units and the starting and ending states.
#July16th  2012####

## january 28th, 2013... trying this new formulation...

# GMS...
let {s in State['GMS',t],w in Stater['GMS',t,s], nu in No_Units['GMS',t]} QT_Max ['GMS',t,s,w]:=
                                                                                   min (Abs_QT_Max['GMS',t,nu],
      if s <=Max_QT_storage_Pt['GMS',t,nu] and w <=Max_QT_storage_Pt['GMS',t,nu] then  -0.00000000102*((s+w)/2)^2 + 0.001*(s+w)/2 + 1922.6
else  if s > Max_QT_storage_Pt['GMS',t,nu] and w > Max_QT_storage_Pt['GMS',t,nu] then   0.000000000008*((s+w)/2)^2- 0.00067*(s+w)/2 + 2361.4
else  if s > Max_QT_storage_Pt['GMS',t,nu] and w <=Max_QT_storage_Pt['GMS',t,nu] then  (0.000000000008*s^2- 0.00067*s + 2361.4-0.00000000102*w^2 +
0.001*w + 1922.6)/2
else                                                                                   (0.000000000008*w^2- 0.00067*w + 2361.4-0.00000000102*s^2 +
0.001*s + 1922.6)/2 ) ;### New: July16th  2012#### modified January 28th, 2013

## MCA...
let {s in State['MCA',t],w in Stater['MCA',t,s], nu in No_Units['MCA',t]} QT_Max ['MCA',t,s,w]:=
                                                                  min (Abs_QT_Max['MCA',t,nu],
     if nu=4 and (s+w)/2<=Max_QT_storage_Pt['MCA',t,nu] then -0.0000000045*((s+w)/2)^2+ 0.003*(s+w)/2+ 753.96
else if nu=4 and (s+w)/2 >Max_QT_storage_Pt['MCA',t,nu] then  0.000000009*((s+w)/2)^2 - 0.00703*(s+w)/2 + 2480.2
else if nu=5 and (s+w)/2<=Max_QT_storage_Pt['MCA',t,nu] then -0.000000007*((s+w)/2)^2 + 0.00418*(s+w)/2+ 917.96
else if nu=5 and (s+w)/2> Max_QT_storage_Pt['MCA',t,nu] then  0.00000001*((s+w)/2)^2- 0.00815*(s+w)/2 + 3007.7
else if nu=6 and (s+w)/2<=Max_QT_storage_Pt['MCA',t,nu] then -0.000000009*((s+w)/2)^2+ 0.0052*((s+w)/2)+ 1089.5
else if nu=6 and (s+w)/2> Max_QT_storage_Pt['MCA',t,nu] then  0.0000000122*((s+w)/2)^2 - 0.0097*((s+w)/2)+ 3562.6);## MCA 4, 5, 6 units### New:
July16th  2012#### updated Janaury 28th 2013.
```

7. FB Model

```
# Calculates the FB as the a function of storage
let { t in Months,  s in State['GMS',t]} FBi['GMS',t,s]:=  0.000000000000000045*s^3 - 0.000000000073*s^2 + 0.000092*s + 640.53;
let {t in Months,  s in State['MCA',t]} FBi['MCA',t,s]:=  0.0000000000000025*s^3 - 0.00000000215*s^2 + 0.00082*s + 637;### It is giving higher
numbers than the right numbers!!!
#let {t in Months,  s in State['ARD',t]} FBi['ARD',t,s]:= 0.00053*s + 419;
#let {t in Months,  s in State['PCN',t]} FBi['PCN',t,s]:=0.00000004*s^2 + 0.0103*s + 475.19;
#let {t in Months,  s in State['REV',t]} FBi['REV',t,s]:=-0.000000004*s^2 + 0.0013*s + 510.16;
let {t in Months,s in State['GMS',t],w in Stater['GMS',t,s]} FBf['GMS',t,s,w]:= 0.000000000000000045*w^3 - 0.0000000000*w^2 + 0.000092*w + 640.53;
let {t in Months,s in State['MCA',t],w in Stater['MCA',t,s]} FBf['MCA',t,s,w]:= 0.0000000000000025*w^3 - 0.00000000215*w^2 + 0.00082*w + 637;
#let {t in Months,s in State['ARD',t],w in Stater['ARD',t,s]} FBf['ARD',t,s,w]:= 0.00053*w + 419;
#let {t in Months,s in State['PCN',t],w in Stater['PCN',t,s]} FBf['PCN',t,s,w]:=0.00000004*w^2 + 0.0103*w + 475.19;
#let {t in Months,s in State['REV',t],w in Stater['REV',t,s]} FBf['REV',t,s,w]:=-0.000000004*w^2 + 0.0013*w + 510.16;
```

8. Prices Model

```
#prices modeling: caluclates the export/import prices as a function of the total flow to the system
let {t in Months, i1 in Inflows['GMS',t], i2 in Inflows['MCA',t]}Total_Sys_Inflow[t,i1,i2]:= (i1+i2)+ (Inflow_ROTR ['PCN','GMS',t,i1]+Inflow_ROTR
['REV','MCA',t,i2]+Inflow_ROTR ['STC','GMS',t,i1]+Inflow_ROTR ['ARD','MCA',t,i2]);#units are in cms# modified August 2012
let {t in Months, i1 in Inflows['GMS',t], i2 in Inflows['MCA',t]:i1>0 or i2>0}  Exp_Price[t,i1,i2]:= max (0,a[t]* c[t]*(Total_Sys_Inflow[t,i1,i2])^
b[t]-dd[t]) ;#this is the original>>>>>###max (0, a[t]* (Total_Sys_Inflow[t,i1,i2])^ b[t]); # power (x^y) equation
let {t in Months, i1 in Inflows['GMS',t], i2 in Inflows['MCA',t]:i1=0 and i2=0}  Exp_Price[t,i1,i2]:= max (0,a[t]* c[t]*(0.50*Inflow_Step['GMS',t])
^ b[t]-dd[t]) ;#this is the original>>>>>###max (0, a[t]* (Total_Sys_Inflow[t,i1,i2])^ b[t]); # power (x^y) equation
let {t in Months, i1 in Inflows['GMS',t], i2 in Inflows['MCA',t]}  Imp_Price[t,i1,i2]:= Exp_Price[t,i1,i2]+ Exp_Imp_Margin;#Exp_Price[t,i1,i2]+
Exp_Imp_Margin*10;#
let {t in Months, i1 in Inflows['GMS',t], i2 in Inflows['MCA',t]}  Cont_Price[t,i1,i2]:= (Exp_Price[t,i1,i2]+ Exp_Imp_Margin/2);
#(Exp_Price[t,i1,i2]/4);#
delete a,b,c;
display  Exp_Price,  Imp_Price, Cont_Price> Prices.out;
```

A.1.4.2. Computation Details

Table 6: List of the Other Modules and Their Functions

Module	Parameter Calculated	Function
HK_"&v&".mod"	HK	Calculates the HK values for the storage plants as a function of each pair of starting and terminal storage states through regression equations capped by the maximum absolute value of the HK for the storage plants
HK_ROTR_"&v&".mod	HK_ROTR	Calculated the HK values of the run-of-the-river plants as a function of the total flow that passes by those plants.
HK_rough_"&v&".mod"	HK_rough	Calculates the total HK values on a river system for a given starting storage state of the storage plants. For instance, HK_rough for peace river would be the summation of the GMS's HK, PCN's HK and STC's HK for a given GMS starting state.
Inflow_ROTR_"&v&".mod	Inflow_ROTR	Calculated the natural inflows to the run-of-the-river plants as a function of the upstream storage plants through regression equations deduced from historical inflows.
Max_Gen_Limits_"&v&".mod"	Max_Gen_Limits	Calculates the maximum generation for the storage plants as a function of the starting and ending storage capped by the absolute maximum generation capacity that dictates the generation curve after a certain point. The calculation is made through regression equations that relate the storage to generation.

Module	**Parameter Calculated**	**Function**
QT_Max__"&v&" .mod"	QT_Max	Calculates the maximum turbine flow for the storage plants as a function of the starting and terminal storage capped by the absolute turbine flow capacity that is when reached the flow must be reduced to account for the generator capacity. The calculation is made through regression equations that relate the turbine flow to generation.
"FB_"&v&".mod"	FBi, FBf	Calculates the forebay corresponding to staring states “FBi” and the forebay corresponding to terminal states “FBf”. The calculation is done through regression equations that relate the storage to the forebay.
“Prices__"&v&". mod"	Imp_Price, Exp_Price, Cont_Price	Calculates the prices as a function of total system inflows

A.2. Main Sets and Parameters of the Model

In the tables below, **Table 7** and **Table 8**, the details of the main sets used in the SDPOM6R are presented.

Table 7: Main Sets and Parameters of the Model in an Alphabetical Order

Set	Definition	Indexed Over	Files[21]	Notes
counter	A counter for the number of starting storage states at each time step	Reservoirs		The range is 1 to number of starting storage states
counterel	A counter for the number of water release decisions	Reservoirs & Months		The range is 1 to number of release decisions
Counter	A counter for the number of terminal storage states at each time step	Reservoirs, Months, State & Inflows		The range is 1 to number of terminal storage states
Inflows	Set of Inflow values for each time step	Reservoirs & Months	Inflows.dat	Units in cms. The range is The starting month to the end month (controllable)

[21] Other than the main modules (Discretizer, Value Iteration and SDP)

Set	Definition	Indexed Over	Files[21]	Notes
Load	Electricity local demand for each time step	Months	Load.dat	Units in MWh
Months	Time step (monthly)			
Rel_Decision	Set of total water release decisions for each time step	Reservoirs & Months	Rel_Decision.dat	Total here means the sum of spills and turbine releases, units in cms
Reservoirs	Set of reservoirs involved in the optimization process		Horizon.dat	
State	Set of starting storage	Reservoirs	States.dat	Units in cms.day
Stater	Set of terminal storage states for each reservoir for each time step	Reservoirs, Months, State & Inflows	Staters.dat	Units in cms.day
Study_Years	Set of the future years used in the study		YMD.dat	

Table 8: The Main Parameters Used in the SDP Model in an Alphabetical Order

Parameter	Definition	Indexed Over	Files	Notes
a	Coefficient provided in a flat file to calculate	Months	Price_Coef.dat	

Parameter	Definition	Indexed Over	Files	Notes
	Import/Export/ contracted prices			
B	Coefficient provided in a flat file to calculate import/export/ contracted prices	Months	Price_Coef.dat	
Cont_Price	Prices of contracted energy	Months& Inflows		
Cont_Rev	Revenue from contracted energy	Months, Load, StateRel_Decision		
Contr_Trans	Contracted amount of energy (imports/exports)	Months	Contr_Trans.dat	Units in MWh
Days_Months	Number of days in each month for the entire planning horizon	Study_Years & Months	YMD.dat	
Delta_Releases	Increment of total releases	Reservoirs & Months		Units in cms
Delta_Staters	Increment of terminal storage states	Reservoirs & Months		Units in cms.day
Delta_States	Increment of starting	Reservoirs		Units in cms.day

Parameter	Definition	Indexed Over	Files	Notes
	storage states			
Desctz_Releases	Generated values of releases according to the pre-calculated release increment and maximum/minimum plant releases provided	Reservoirs, Months & counterel		Units in cms.day
Desctz_Staters	Generated values of terminal storage states according to the pre-calculated state increment, maximum/ minimum storage values, starting states and inflow values	Reservoirs, Months, State, Inflows & counter		Units in cms.day
Desctz_States	Generated values of starting storage states according to the pre-calculated state increment and maximum/ minimum storage values provided	Reservoirs & counter		Units in cms.day
End_Months	The last monthly time step		Horizon.d	

Parameter	Definition	Indexed Over	Files	Notes
	in the planning horizon		at	
EX_Exp_Rev	Expected revenue from the spot market exports	Months, Load, State & Rel_Decision		Units are in million $
EX_Imp_Cost	Expected cost from the spot market imports	Months, Load, State & Rel_Decision		Units are in million $
EX_Income	Expected total income from export, import and contracts	Months, Load, State & Rel_Decision		Units are in million $
Exp_Imp_Margin	Difference between export and import prices			default 9.11, units are in $
Exp_Price	Prices of exported energy	Months & Inflows		Units are in $
FBf	Final forebay at each time step corresponding to a certain storage	Reservoirs, Months, State, Inflows & Stater		Units are in m
FBi	Starting forebay at each time step corresponding to a certain storage	Reservoirs, Months & State		Units are in m
Generation	Calculated generation of each plant	Reservoirs, Months, State,		Units in MWh

Parameter	Definition	Indexed Over	Files	Notes
		Inflows, Stater& Rel_Decision		
HK_rough	a rough estimate for HK to use in MVW	Reservoirs, Months & State		
HK		Reservoirs, Months, State, Inflows & Stater		
Imp_Price	Prices of imported energy			Units in $
Inflow_Step		Reservoirs & Months		Units in cms
IPP_Therm	Generation of the independent power producers (IPP) and thermal plants. Added to the plants generation to get the total generation of the system involved	Months		Units in MWh
Iter	Counter for number of iterations used to convergence in the value iteration procedure			
Max_Gen_Limits	Limits on maximum	Reservoirs &	Gen_Limit	Units in MWh

Parameter	Definition	Indexed Over	Files	Notes
	generation of each plant	Months	s.dat	
Max_Staters_Act	Rounded number for maximum terminal storage states	Reservoirs, Months, State &Inflows		Units in cms.day
Max_Staters	Provided value for maximum terminal storage states for each reservoir in each time step	Reservoirs & Months	State_Space.dat	Units in cms.day
Max_States_Act	Rounded number for maximum starting storage states	Reservoirs		Units in cms.day
Max_States	Provided value for maximum starting storage states for each reservoir	Reservoirs	State_Space.dat	Units in cms.day
Min_Gen_Limits	Limits on minimum generation of each plant	Reservoirs & Months	Gen_Limits.dat	Units in MWh
Min_Staters_Act	Rounded number for minimum terminal storage states	Reservoirs, Months & State		Units in cms.day
Min_Staters	Provided value for minimum terminal storage	Reservoirs & Months	State_Space.dat	Units in cms.day

Parameter	Definition	Indexed Over	Files	Notes
	states for each reservoir in each time step			
Min_States_Act		Reservoirs		Units in cms.day
Min_States	Provided value for minimum starting storage states for each reservoir	Reservoirs	State_Space.dat	Units in cms.day
MVW_GMS	Marginal value of water at GMS	Months, State & State		Units are in $/MWh
MVW_MCA	Marginal value of water at GMS	Months, State & State		Units are in $/MWh
N_Releases	Number of releases generated according to the release increment and maximum/minimum releases for each plant	Reservoirs & Months		
N_Staters	Number of terminal storage states generated according to the storage increment, maximum/minimum storage states , starting	Reservoirs, Months, State & Inflows		

Parameter	Definition	Indexed Over	Files	Notes
	storage states and inflows for each plant in each time step			
N_States	Number of starting storage states generated according to the storage increment and maximum/minimum storage states ,for each plant	Reservoirs		
Outage	Factors <=1 representing the maximum generation capacity ratio taken the outage schedule into consideration	Reservoirs & Months	Outage.dat	
Plant_Release	Plant release for each time step depending on the hydraulic balance and plant release limits	Reservoirs, Months, State, Inflows, Stater & Rel_Decision		Units in cms
Policy_Income	The total income of the policy which is the sum of the revenue from import,	Months, Load, State & Rel_Decision		Units in million $

Parameter	Definition	Indexed Over	Files	Notes
	export and contract transactions.			
Prob_Inflow	The probability of each inflow value provided for each plant in a certain time step	Reservoirs, Months & Inflows	Inflows.dat	
PV_diff_total, PV_diff	The total and sequential difference of the present value of water; used in the value iteration procedure	Months & State		Units in million $
PV_Fin , PV_Max, PV_Temp	Maximum present value of water at a certain time step over the terminal storage states and inflow scenarios	Months & State		Units in million $
PV	Present water value	Months, Load, State &Rel_Decision		Units in million $
QP_Max	Maximum plant release	Reservoirs & Months	QP_Limits .dat	Units in cms
QP_Max_Act	Rounded value for maximum plant release	Reservoirs & Months		Units in cms

Parameter	Definition	Indexed Over	Files	Notes
QP_Min	Minimum plant release	Reservoirs & Months	QP_Limits .dat	Units in cmc
QP_Min_Act	Rounded value for minimum plant release	Reservoirs & Months		Units in cms
QT_Max	Maximum turbine release	Reservoirs & Months	QT_Limit s.dat	Units in cmc
QT_Min	Minimum turbine release	Reservoirs & Months	QT_Limit s.dat	Units in cms
Rate	Interest rate			Default 0.05858 (monthly rate)
SiW	The surplus/deficit of water in each reservoir due to a transition from a storage state to another including the inflows and excluding the releases (Starting storage+Inflow-terminal storage)	Reservoirs, Months, State, Inflows, Stater & Rel_Decision		Units in cms.day
Spill	Forced spills (Non-power spills)	Reservoirs, Months, State, Inflows, Stater		Units in cmc

Parameter	Definition	Indexed Over	Files	Notes
		& Rel_Decision		
Spot_Buy		Months, Load, State, Inflows, Stater& Rel_Decision		Units in million $
Spot_Sell		Months, Load, State, Inflows, Stater& Rel_Decision		Units in million $
Start_Months	The first monthly time step in the planning horizon		Horizon.dat	
State_Prob	The probability attached to each storage transition state according to the values of the inflow and releases used	Reservoirs, Months, State, Inflows, Stater & Rel_Decision		
Total_Sys_Inflow	Summation of the system inflows for each time step	Months & Inflows		Units in cms
Tot_Gen	Total system generation including thermal plants and the IPPs	Months, State, Inflows, Stater & Rel_Decision		Units in MWh

Parameter	Definition	Indexed Over	Files	Notes
Trade	Export or Import spot energy transactions	Months, Load, State, Inflows, Stater & Rel_Decision		Units in MWh
Trade_Exp_Limit	Export limits according to the capacity of transmission lines and other considerations	Months	TradeLimits.dat	Units in MWh
Trade_Imp_Limit	Import limits according to the capacity of transmission lines and other considerations	Months	TradeLimits.dat	Units in MWh
Trans_Prob	The joint probability of the transition from a combination storage state of the reservoir considered to another combination of terminal storage states	Months, Load , State, Inflows, Stater & Rel_Decision		
Trans_Prob_S	Sum of transition probability over the terminal storage states and	Months, Load, State & Rel_Decision		

Parameter	Definition	Indexed Over	Files	Notes
	inflows			
Turbine_Release	Outflows coming through turbines to generate electricity (power spills)	Reservoirs, Months, State, Inflows, Stater & Rel_Decision		Units are in cms

A.3. How to Run the Model

The main steps that the user has to follow are illustrated in the following workflow diagram.

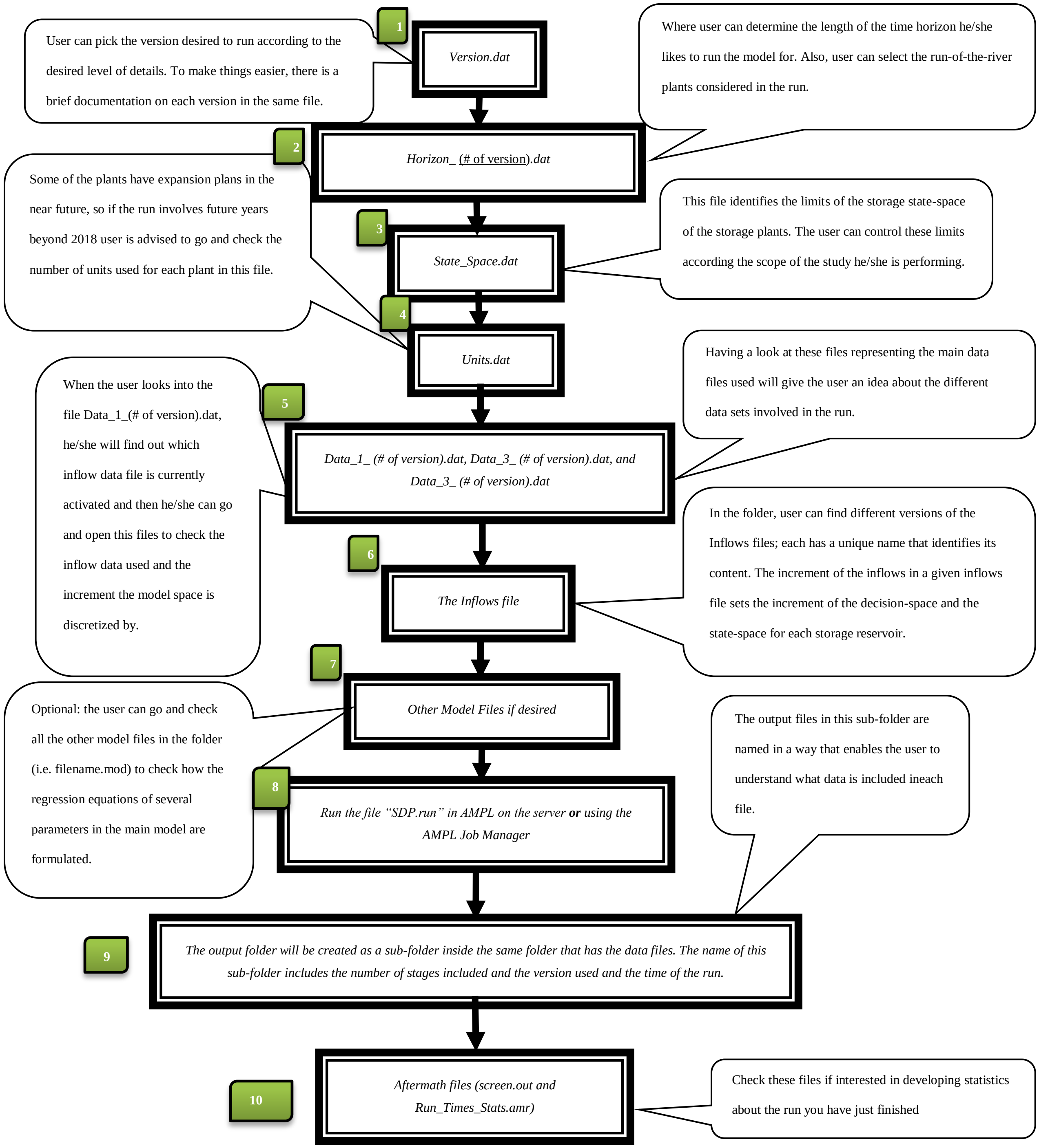


Figure 41: Schematic of the Main Steps Needed to Run the SDPOM6RM Model