A GENETIC-ALGORITHM BASED AUTOMATIC MODEL CALIBRATOR FOR THE UBC WATERSHED MODEL

by

YAO-HUNG LAN

B.A.Sc. The University of British Columbia, 1998

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

DEPARTMENT OF CIVIL ENGINEERING

We accept this thesis as conforming

to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

JULY, 2001

© Yao-Hung Lan, 2001

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Civil Engineering

The University of British Columbia
Vancouver, Canada

Date Aug 1st, 2001

ABSTRACT

In this study, an optimization and search technique, based on the genetic algorithms (GAs) approach, is successfully used to develop an automatic model calibrator for the UBC Watershed Model. Unlike the existing random search calibration procedure, which limits the number of simultaneously calibrated modeling parameters to groups of about three to six at a time, the new GA-based calibrator allows all modeling parameters to be simultaneously evaluated. Because of the non-linear interactions between the modeling parameters, the simultaneous evaluation of all modeling parameters is demonstrated to achieve a good model calibration efficiently and quickly. The fundamental components of GAs as inspired by the Darwinian principle of natural selection are explained in detail in order to develop a complete GA-based model calibrator. A flow chart is used to illustrate the computational implementation of the GA procedures. Why GAs can work efficiently in finding an optimal set of modeling parameter values is explained by the schema theory with mathematical proofs provided. To test the soundness of the GA code developed for the automatic calibrator of the UBC Watershed Model, two well-studied watersheds in the Province of British Columbia, Campbell River and Illecillewaet River, are used. The effects of genetic operators (crossover, niching and elitism) on GA search efficiency are individually demonstrated. To objectively determine the performance of a calibrated watershed model, the difference between the observed and simulated streamflows is statistically measured. Four statistical measures are evaluated: coefficient of linear correlation (or coefficient of determination), Nash & Sutcliffe coefficient of efficiency *(e!)*, least squares objective function and least absolute difference objective function are introduced. GA computational experiments show that the Nash & Sutcliffe coefficient of efficiency *(e!)* exhibits the most consistently decreasing trend of streamflow volume error (*dV/V*) as the coefficient value increases. A fifth statistical measure, the modified Nash & Sutcliffe coefficient *(eopt!)*, is also used to quantify the difference between the observed and simulated streamflow data, and ensures the optimal or near-optimal set of model parameter values found at the end of a GA search achieves both high *e!* and low *dV/V* at the same time.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my thesis supervisor, Dr. M. Quick for providing the UBC Watershed Model as the first testing vehicle for my self-coded genetic-algorithms solver. Without Dr. Quick's guidance and patience, I would not have learned as much as I did in this research. Thanks to Dr. G. Lawrence for reviewing my work. I am also greatly indebted to Edmond Yu and Xiaosi Feng for helping me to resolve Visual Basic and Fortran coding problems I encountered. A note of special thanks should be given to Bryan Tolson for constantly encouraging me to dig deeper in the fields of Genetic Algorithms and Neural Networks. Lastly, I wish to mention that this research is funded by the scholarship from the Natural Sciences and Engineering Research Council of Canada.

1.0 INTRODUCTION

Watershed modeling is a key part of watershed management for the purposes of flood prevention, hydropower generation and preparation for possible water shortage in drought years. Detailed model calibration is often required to ascertain the usefulness and reliability of a model as a tool in forecasting streamflow. Traditional manual model calibration is laborious and non-systematic. It usually involves visual inspection and modelers have no means to see the whole picture of the model calibration process. Although random automatic model calibration frees modelers from repeatedly adjusting the model parameter values, it depends too much on "luck" and may not reach an optimal, or near optimal, set of model parameter values. For both manual calibration and automatic random calibration, limitations amplify as the number of modeling parameters and the complexity of watershed model structure increases.

Genetic algorithms (GAs) are heuristic search algorithms used as an optimization technique in single and multi-objective problems. The technique was inspired by the Darwinian theory of the survival of the fittest. The algorithm mechanisms mimic the natural selection and evolutionary process, improving the survivable characteristics of the species. Genetic algorithms differ from other conventional gradient-based optimization techniques, because no full or partial derivative of any order is required.

Since the 1990s, advances in genetic algorithms, coupled with high-speed personal computers, provide means to optimize the fitness (resemblance) between the simulated and the observed streamflow data of a watershed. Several researchers have reported their success in applying GAs in various model calibrations (MW Soft Inc. 1999 & Solomatine 1998). In the context of civil and water resource engineering, GAs have been applied in calibrating water distribution networks (H2ONET), sanitary sewer systems (SWMM in progress), river hydraulics (Mike 11) and watershed modeling.

1.1 Objectives

This thesis is concerned about the application and applicability of genetic algorithms in the systematic calibration of a computational model, particularly in the context of watershed modeling. The UBC Watershed Model was chosen as an interactive testing vehicle to first verify the soundness of the genetic algorithm code programmed by the author, and secondly, to demonstrate the strength and capability of genetic algorithms in model calibration.

The thesis is not a comparative study of the UBC Watershed Model and its applications. No attempts were made to investigate the model structure and thus no improvement was recommended.

1.2 Scope of Study

In order to meet the thesis objectives, the following tasks were performed:

- Review of conventional manual model calibration and of automatic random search model calibration.
- Review of common statistical measures used to evaluate model performance and soundness of calibration.
- Investigate and review the applicability of genetic algorithms as an optimizer, or a model calibrator, in engineering literature.
- Investigate the components of genetic algorithms and usage of major genetic operators.
- Outline the computational procedure for genetic algorithm implementation
- Research the theory of GAs and why GAs work.
- Implement the computation of the genetic algorithms in Fortran and customize the code in Visual Basic to interface directly with the UBC Watershed Model.

- Apply GAs for the calibration of two watersheds previously studied using the UBC Watershed Model, and report on the GA techniques used to improve search efficiency.
- Evaluate the GA calibration results and discuss their implications with respect to applicability of GA techniques.
- Summarize the findings of the study and recommend future work on GA model calibration for use with the UBC Watershed Model.

1.3 UBC Watershed Model

Development of the UBC Watershed Model was started in the late 1960s by Quick and Pipes (1977), and has undergone continuous development since that time by the UBC Civil Engineering Mountain Hydrology Group. It utilizes daily maximum and minimum temperatures and precipitation as input data and generates daily watershed outflow as the main output (UBC Mountain Hydrology Group 1995). Because the calculation of the watershed outflow requires and depends on the values of snowmelt, soil moisture budget, soil and groundwater, the model estimates these values individually in a given sequence before the watershed outflow is generated. Thus the model also provides the value of related hydrologic parameters. The watershed model was designed for short-term forecasting of the streamflow of the river that drains the watershed. It has been extensively used for flood forecasting and reservoir inflow prediction.

The input of the UBC Watershed Model consists of two major components. Besides the meteorological input data required, the user or modeler must also specify a set of modeling parameters, which quantitatively characterize the watershed and its response behavior. While some of these parameters are geographical data obtainable through surveying or gauging, some parameters must be estimated or back calculated. Although the unobtainable parameters can be estimated, back calculation is often required based on observed records to confirm the validity of the initial estimate values. Thus the UBC Watershed Model, similar to other models, must undergo the process of model calibration

for each specific watershed. This calibration process requires measured streamflow data, which are used to evaluate the model estimates of watershed outflow.

Under the existing structure, the UBC Watershed Model contains an optimization module specifically designed for model calibration. Within the module, the calibration process applies the concept of a constrained random search to find a set of model parameter values to characterize the watershed and maximize the resemblance of the watershed model to the real watershed. However, direct search of the values of all modeling parameters is not possible with the present model calibration framework. The simultaneous evaluation of all modeling parameters could be very useful because of the non-linear interactions between the various parameters. Until now, the number of modeling parameter values to be searched simultaneously is often limited to groups of about three to six at a time, and the users then proceed through further groups of parameters, moving from the more sensitive parameters to the less sensitive ones, and then repeating the process to refine the parameter values.

The intention is to use the genetic algorithm in parallel with this existing calibration module. This will provide the opportunity and the environment for genetic algorithms to be tested for their applicability as an optimization technique for model calibration. In particular, the genetic algorithm will be used to evaluate the values of all chosen model parameters concurrently.

1.4 Thesis Layout

This thesis consists of seven chapters in total. Chapter 2 offers a literature review on manual model calibration, objective calibration measure for model conformance, and automatic model random calibration. Chapter 3 offers a literature review on genetic algorithms as an optimization technique and its application in computational model calibration and many other fields. Chapter 4 discusses the basic genetic algorithm principles and fundamental genetic operators used to search for the optimal solution. It

also provides a summary of GAs' computational procedures. Chapter 5 explains and mathematically proves why genetic algorithms can be successfully used as an optimization technique. Chapter 6 investigates how well the existing GA code can automatically and systematically calibrate a watershed, based on certain statistical conformance measures. Two case studies are used to show which genetic technique, or combination of techniques, is most efficient in GA search. Three statistical measures are compared to determine which is the most consistent measure of model performance. The conclusion and the findings are summarized in Chapter 7.

2.0 LITERATURE REVIEW OF MODEL CALIBRATION METHODS

In this chapter, the need for model calibration is discussed. Literature reviews are presented for manual model calibration procedures, for objective calibration measures of model performance, and for automatic model random calibration.

2.1 Model Calibration

A model is a conceptual representation of the understanding of physical phenomena and processes. A watershed (hydrologic) model is a model that describes the governing processes of how precipitation in the form of snow and rain gradually flows over surface and/or through soil and leaves the watershed as streamflow. The reliability and the accuracy of the modeling results depend on the appropriateness of assumptions made on physical processes, quality of input data and estimated values of modeling parameters (Sorooshian and Gupta 1995). In watershed modeling, the input data include precipitation, both rain and snow, temperature, snowpack depth and many others. The data quality depends on how representative the field measurement data are for the entire watershed. The estimation of modeling parameter values, on the other hand, is less straightforward. In general, the modeling parameters can be classified into two types (Sorooshian and Gupta 1995).

1. Physical Parameters: parameters which represent physically measurable properties of the watershed.
2. Process Parameters: parameters which represent the implicit characteristics of the watershed. These are often not directly measurable.

Examples of physical parameters are the areas of different watershed elevation bands and the percentage of forested and vegetated area. Examples of process parameters include the rainfall fast runoff time constant and the snowmelt fast runoff time constant. Some parameters may be measurable, such as the impervious ratio of land surface area, or the

effective depth of maximum subsurface storage, or they may have to be estimated as part of the calibration process and deemed as process parameters rather than the physical parameters. Because the physical parameters are directly measurable, representative values can be derived from field data with reasonable accuracy, whereas the process parameters are difficult to estimate and their estimation relies heavily on the modeler's knowledge of the watershed and past modeling experience. Often the estimation of process parameter values is the part of computational modeling that requires the most attention in calibration. Therefore, model calibration can be defined as the process of selecting a set of process parameter values so that the resulting model closely simulates the physical process under scrutiny.

2.2 Manual Model Calibration

Traditionally, the calibration process is more an art than a science. A modeler, based on his/her experience, long-term observation and knowledge, makes a first estimate of the values of process modeling parameters. He or she then initiates the model with these estimated parameter values, and then compares the model output with the observed data. If the model output does not visually resemble the observed data, the modeling parameter values are then re-adjusted again and again until a higher degree of similarity is achieved. This trial and error iterative process of parameter value fine-tuning is called manual model calibration. It is often repetitive, time consuming and may be frustrating, yet it is the most common calibration process followed by modelers in the past three decades.

The manual calibration process is practiced in watershed modeling because certain parameters control streamflow volume and some control specific aspects of hydrograph shape, etc. These distinct characteristics determine the need for the various parameters and make calibration possible. Calibration can be carried out using sub-groups of parameters, and by constraining the range of values as the process proceeds – a constrained search. However, the most profound weakness of manual model calibration

is the fact that it is very subjective because a modeler simply eyeballs the simulated data graphed over the observed data and determines whether they are similar visually.

In the context of the calibration of watershed models (including UBC Watershed Model), a hydrograph of the streamflow leaving the watershed is typically used as the calibration measure. Despite the fact that a modeler can make reasonable estimate of the values of modeling parameters, some degree of calibration is still required. Theoretically, the manual model calibration process should persist until the modeler feels that the calculated streamflow hydrograph visually resembles the observed hydrograph. Due to the complexity involved in a watershed model, it is not unusual that tens of modeling parameters must be manually adjusted repeatedly. The multi-dimensionality of the watershed model may cause serious difficulty for the modeler in manual model calibration. This is why the manual calibration process is also humorously called the "Guess-Try-Swear" process. Nonetheless, knowledge about the characteristics and behavior of the watershed is definitely an asset to the modeler in knowing what parameters to adjust and what values to set.

2.3 Objective Model Calibration Measures

Because the manual model calibration involves a great deal of subjective judgment and eyeballing, two modelers may obtain two very different sets of modeling parameter values with no direct means available to measure and determine which one is superior to the other. Hence an objective and standardized way of measuring the degree of conformance between the observed and the simulated data should be developed for both the manual and automatic model calibration. Intuitively, modelers turn to the statistical usage of the coefficients of linear correlation (directly related to the coefficient of determination) and efficiency, which mathematically quantify the numerical discrepancies between the observed and the simulated data. In addition, drawing from regression and curve fitting theory, various forms of the least squares method can also be used (Sorooshian and Gupta 1995). The following discusses the application of statistical

principles in devising objective measures for fitness of model calibration. Their advantages and limitations will also be briefly mentioned.

2.3.1 Coefficients of Linear Correlation and Determination

The coefficient of linear correlation is a statistical strength measure of the relationship between two sets of interval scaled data, in this case, the observed and the simulated data. It can assume any value from –1 to +1 inclusive. If the two sets of interval scaled data are positively proportional to each other or exactly the same as each other, the coefficient is +1, which indicates a perfect positive correlation. If the two sets of data are inversely related, the coefficient is -1, which indicates a perfect negative correlation. A coefficient value of zero means that the two sets of data are not linearly related at all.. The coefficient of linear correlation can be expressed as

$$r = \frac{n(\sum_{i=1}^{n} Q_{obs_i} \cdot Q_{sim_i}) - (\sum_{i=1}^{n} Q_{obs_i})(\sum_{i=1}^{n} Q_{sim_i})}{\sqrt{\left[n(\sum_{i=1}^{n} Q_{obs_i}^{2}) - (\sum_{i=1}^{n} Q_{obs_i})^{2}\right]\left[n(\sum_{i=1}^{n} Q_{sim_i}^{2}) - (\sum_{i=1}^{n} Q_{sim_i})^{2}\right]}} \qquad \textbf{(Equation 2-1)}$$

where r is the coefficient of linear correlation. Q_{obs_i} and Q_{sim_i} are the values of the observed and the simulated data at a particular time interval. n is the total number of time intervals and i is the time interval index.

The coefficient of linear correlation offers a useful tool to measure the conformance of the shapes of the two plotted data curves. In the context of watershed model calibration, if the observed and the simulated hydrographs have very similar shapes, the correlation coefficient approaches positive unity. Therefore, one may say that a correlation coefficient of 0.9 or higher indicates very close fit of the shape of streamflow hydrographs. Figure 2-1 gives an example of two perfectly linear correlated hydrographs with a coefficient of +1. The figure, however, also clearly reveals the significant

discrepancy in flow magnitude and total streamflow volumes (areas under curves) despite a perfect positive linear correlation of +1. Therefore even though correlation coefficient measures the shape fitness of the two plotted data curves, its inability to measure the numerical discrepancy between two data shows that the use of correlation coefficient alone is not an adequate measure of good model calibration.

The coefficient of determination, r^2, by definition is simply the direct square of the coefficient of linear correlation. It can assume any value between 0 and 1. Similar to the coefficient of linear correlation, it only relates to the shape conformance of the observed and simulated hydrographs, and is independent of discrepancy in total streamflow volume (UBC Mountain Hydrology Group 1995). Thus, coefficient of determination alone is not an adequate measure of good model calibration either.

Hence, if the coefficients of linear correlation and determination are to be used as part of the model performance measure in the calibration process, they should be jointly used with other statistical measures in order to emphasize the streamflow volume.

Figure 2-1: An Example of Two Sets of Perfectly Linear Correlated Data

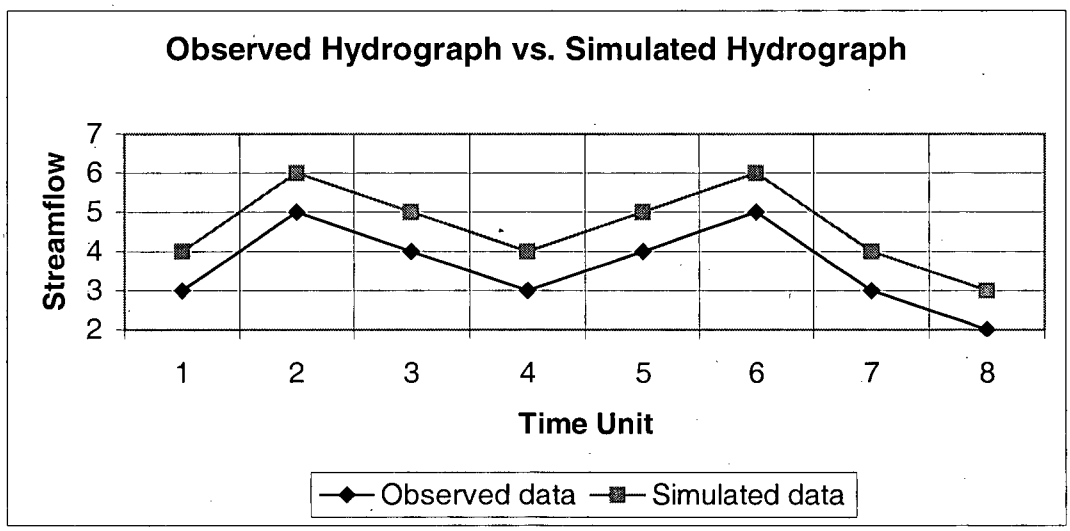
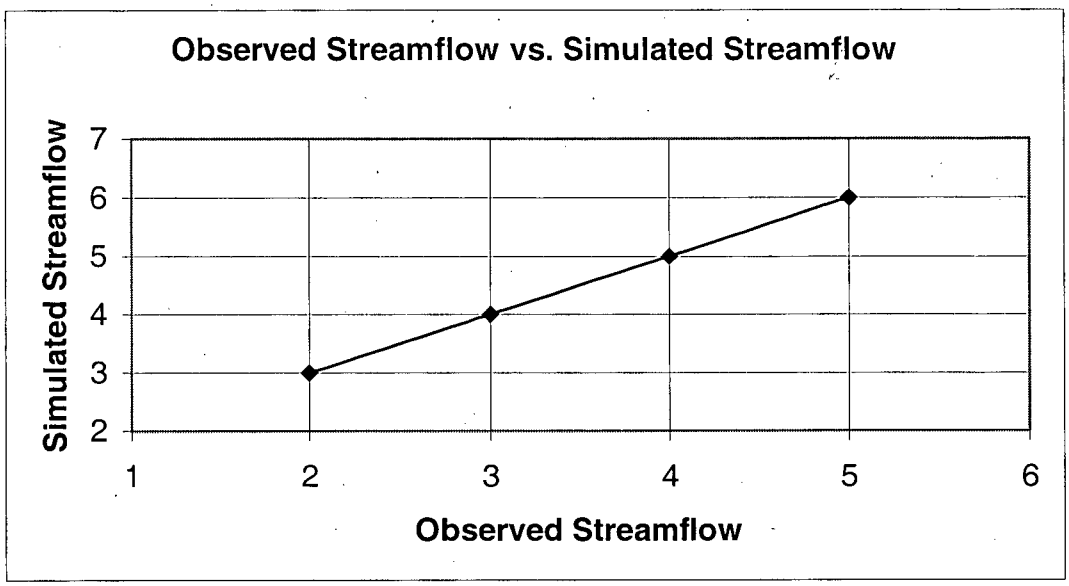


Figure 2-2: Perfect Linear Correlation between Two Data Sets Yields a Correlation Coefficient of +1



2.3.2 Nash & Sutcliffe Coefficient of Efficiency

Another model calibration measure is the coefficient of efficiency originally suggested by Nash & Sutcliffe (1970). The coefficient of efficiency is evaluated based on the shape and magnitude difference between two plotted data series. In the context of watershed model calibration, the coefficient of efficiency measures how well the simulated streamflow hydrograph is predicted, compared to the observed streamflow hydrograph in both hydrograph shape and streamflow magnitude. Thus, the Nash & Sutcliffe coefficient of efficiency is superior to correlation coefficient as an objective model calibration measure. Similar to the correlation coefficient, the efficiency coefficient is dimensionless but it can assume any value between negative infinity and +1. The coefficient of efficiency, *e!*, can be expressed as:

$$e! = 1 - \frac{\sum_{i=1}^{n}(Q_{obs_i} - Q_{sim_i})^2}{\sum_{i=1}^{n}(Q_{obs_i} - Q_{obs_{mean}})^2} = 1 - \frac{residual\ variance}{total\ variance} \qquad \textbf{(Equation 2-2)}$$

where *e!* is the coefficient of efficiency and $Q_{\underset{mean}{obs}}$ is the time average of the observed flow, Q_{obs}.

Because the coefficient of efficiency accounts for both the shape and total flow volume differences between the observed and the simulated hydrographs, an efficiency value of +1 indicates a perfect match in both shape and in flow magnitude and therefore the two hydrographs are identical (UBC Mountain Hydrology Group 1995). Although the coefficient of efficiency is a better model calibration measure than the coefficients of correlation and determination, it suffers a shortcoming that is often unnoticed. The shortcoming is that the coefficient of efficiency biases towards data sets with large total variances. Using the watershed model calibration as an example, if the observed hydrograph has many large flow peaks, which directly leads to a large $\sum_{i=1}^{n}(Q_{obs}_{i} - Q_{\underset{mean}{obs}})^2$ term, then the numerical discrepancy term, $\sum_{i=1}^{n}(Q_{obs}_{i} - Q_{sim}_{i})^2$ is relatively small and may easily be overlooked by the modelers.

Table 2-1 shows the computational procedures of the coefficient of efficiency for the example given in Figure 2-1. It has a negative value even though the two hydrographs have identical shapes. A negative value is already anticipated because the large difference in flow magnitude results in a total flow volume difference of 28%.

Table 2-1: Example of Efficiency Coefficient Calculation

Time	Q obs	Q sim	$(Q\ obs-Q\ sim)^2$	$(Q\ obs-Q\ obs\ mean)^2$
1	3	4	1	0.391
2	5	6	1	1.891
3	4	5	1	0.141
4	3	4	1	0.391
5	4	5	1	0.141
6	5	6	1	1.891
7	3	4	1	0.391
8	2	3	1	2.641
Total Volume	29	37		
Q obs mean =	3.625			
$\Sigma (Q\ obs-Q\ sim)^2 =$		8		
$\Sigma (Q\ obs-Q\ obs\ mean)^2 =$		7.875		
coefficient of efficiency =		-0.016		

Both the Watflood Model developed by Kouwen of the University of Waterloo (Kouwen 1997) and the UBC Watershed Model by Quick (UBC Mountain Hydrology Group 1995) apply the Nash-Sutcliffe model efficiency to evaluate the fitness of simulated results. To further emphasize the importance of the streamflow conservation principle, the total areas under observed and simulated hydrographs should be the same. Therefore, the UBC Watershed Model also adopts a slightly modified coefficient of efficiency, *eopt!*, as the objective model calibration measure to select the best of the 10 highest efficiency results. The modified coefficient of efficiency, *eopt!*, is expressed as:

$$eopt! = e! - abs\left(1 - \frac{\sum_{i=1}^{n}(Q_{sim})_i}{\sum_{i=1}^{n}(Q_{obs})_i}\right) = e! - \frac{abs\left(V_{total\ observed} - V_{total\ estimated}\right)}{V_{total\ observed}} \qquad \textbf{(Equation 2-3)}$$

where $V_{total\ observed}$ is the total observed streamflow volume integrated over the duration of model simulation period, and $V_{total\ estimated}$ is the total estimated streamflow volume integrated over the duration of the same model simulation period.

By using *eopt!* as a model calibration measure, the calibration process places equal weight on achieving high hydrograph shape conformance and minimizing streamflow volume discrepancy. It ensures that after calibration the simulated results are as similar to the observed data as possible.

2.3.3 Objective Calibration Measure Based on Least Squares Method

One other objective model calibration measure is the sum of the squares of the numerical differences between the observed and the simulated data sets. It is often used as a fitness measure in regression analysis such as curve fitting and derivation of an unknown relationship. Unlike the coefficients of correlation and efficiency, which are to be maximized, the sum of the squares of difference is to be minimized in order to achieve best possible conformance between the two data sets. Hence the technique is commonly referred as the least squares method.

In the context of watershed modeling, using the sum of the squares differences as a measure transforms model calibration into a minimization process. The dimensional form of the objective function can be expressed as:

Minimize $z = \sum_{i=1}^{n} (Q_{obs_i} - Q_{sim_i})^2$ **(Equation 2-4)**

where z is the objective function to be minimized. The summation term of Equation 2-4, in statistics, known as the residual variance, is used as part of the numerator in the Nash-Sutcliffe coefficient of efficiency. The dimensionless form of the least squares objective function can be expressed as:

Minimize $z = \dfrac{\sum_{i=1}^{n} (1 - \frac{Q_{sim_i}}{Q_{obs_i}})^2}{n}$ **(Equation 2-5)**

The dimensionless form is preferred over the dimensional form because it can assume any value between 0 and +1 for $Q_{i}^{sim} \leq 2Q_{i}^{obs}$. In addition, by using the dimensionless form, the sum of squares differences is averaged over the total number of intervals used in the observed data and is less affected by the interval size.

It is important to note that even though the sum of the squares differences bears high similarity to the coefficient of efficiency, it formulates the calibration process into a minimization problem instead of a maximization problem. However, through a minor modification, the objective function in Equation 2-5 can be readily transformed into a maximization problem for GA search, i.e.

Maximize $z = 1 - \frac{\sum_{i=1}^{n}(1 - \frac{Q_{i}^{sim}}{Q_{i}^{obs}})^2}{n}$ **(Equation 2-6)**

For simplicity, the objective function, z, in equation 2-6 from now on will be referred as the *least squares difference* objective function in this thesis.

2.3.4 Objective Calibration Measure Based on Least Absolute Difference

If the *least squares* objective function is used as a model calibration measure, larger flow differences can be over-emphasized and mislead the calibration process because the numerical difference is squared at every data interval. For example, if a set of observed streamflow data contains a portion of false readings (which is not identifiable) and the portion happens to be the only section that cannot be closely matched by the simulated data, then using the sum of the squares differences may easily deviate the modeler from obtaining the best set of modeling parameter values. Thus, for observed data with low reliability, the sum of the squares differences cannot be used as an effective calibration measure. Alternatively, the sum of simple absolute differences may be used as a model calibration measure. The *least absolute difference* objective function can be expressed as

Minimize $z = \sum_{i=1}^{n} abs(Q_{obs_i} - Q_{sim_i})$ **(Equation 2-7)**

or

Minimize $z = \frac{\sum_{i=1}^{n} abs(1 - \frac{Q_{sim_i}}{Q_{obs_i}})}{n}$ **(Equation 2-8)**

To apply the genetic algorithms, which requires a minimization to be transformed into a maximization problem, equation 2-8 can be modified as follows:

Maximize $z = 1 - \frac{\sum_{i=1}^{n} abs(1 - \frac{Q_{sim_i}}{Q_{obs_i}})}{n}$ **(Equation 2-9)**

It is important to note the choice of turning the calibration process into either a minimization or a maximization problem depends on the type of optimization technique to be used and the convenience of computer programming. For genetic algorithms which are the optimization techniques proposed to be used in UBC Watershed Model calibration, a maximization of the objective function is preferred and considered to be much easier to deal with. More details will be discussed in later chapters.

2.4 Automatic Random Search Model Calibration

Because calibration is important for the reliability and the applicability of a model, modelers' quests for faster and more efficient calibration process continue. Early work led to the creation of automatic calibration which randomly searches within a specified range of reasonable values for the best set of modeling parameter values that would produce the best fit to the observed data (i.e. optimize the pertinent calibration measure). This is a constrained random search, which ensures that parameters are within physically reasonable limits.

These calibration methods depend on the following:

1. Advances in computer technology so that programs can be written to instruct the computer to run the model repeatedly without human intervention until a better agreement with the observed data is achieved.
2. Random number generation that is used to produce parameter values at random within the upper and lower limits. A well-designed random generator should produce different values uniformly.
3. Statistical measures such as the coefficient of efficiency, or the sum of squares differences are used as the objective function to be optimized. The objective function value indicates the degree of conformance between the observed and the simulated data, and can also be used as a stopping criterion to terminate the calibration process.

These steps, when integrated, form an automatic model calibrator with random search capability.

Although the automatic random search model calibration is designed to free modelers from strenuous manual calibration process, the random search technique it implements is not systematic and efficient.

In a search space of only three dimensions, for example, the random search may have the "luck" to approach the areas (*schemata*) of high objective function values. But if the search space is in the order of tens of dimensions, say 20 for example, random search for the best set of modeling parameter values can be as inefficient as the enumerative search scheme in which all possible combinations of parameter values are to be listed. Conservatively speaking, if a parameter dimension has 20 possible values, a 20-dimensional search space would require an enumeration of $20^{20} = 1.05*10^{26}$ combinations. This implies that the random search scheme must test at least a significant portion of the $1.05*10^{26}$ combinations before an optimal or near optimal candidate solution can be confidently declared.

The limitations of random search in automatic model calibration leads to the application of genetic algorithms, which dramatically reduces the number of search points (candidate solutions with different combinations of parameter values) required to be tested before an optimal or near optimal candidate solution is found and a winner declared.

3.0 LITERATURE REVIEW OF GENETIC ALGORITHMS

Genetic algorithms are heuristic search algorithms used as an optimization technique. Because the development of genetic algorithms was inspired by the Darwinian theory of the survival of the fittest, its mechanisms bear great similarity to the natural selection and evolutionary process of a survivable species. In addition, many terms used in genetic algorithms are directly borrowed from the field of biology.

The genetic algorithms used in practice in solving single or multi-objective optimization problems are indeed a simplified version of the natural history of a particular species, in which the candidate solutions are the various members of the species with different genetic features. Thanks to modern advances in computing technology, the evolutionary process of a species, which could have taken thousands to millions of years or generations to achieve, can now be virtually simulated in the matter of minutes or hours. In every generation of the evolutionary process, only the candidate solutions achieving better objective function values will survive and be used to create new sets of candidate solutions. The best-of-generation candidate solution found at the end of the evolutionary process is then deemed as the optimal solution to the optimization problem.

In the context of watershed model calibration, candidate solutions are sets of model parameter values tested in the model calibration process. Only the sets of model parameter values with high calibration performance measure, can be used combinatorially to create the next improved set of model parameter values and ultimately achieve the best possible agreement between the observed and the model-simulated data. Thus the core feature of genetic algorithms as an optimization tool is built on its evolutionary process, which favorably biases towards high-performance candidate solutions and purges the poor-performance solutions.

It is generally recognized in the genetic algorithms community that genetic algorithms were invented by John Holland in the 1970s, marked by the publication of his book "Adaptation in Natural and Artificial Systems" in 1975 (Obitko 1998). His students and

colleagues further expanded his ideas and applied them in various fields of study. They are discussed in the next two sections. Goldberg, a former Ph.D. student of Holland conducted extensive research in GAs and summarized his findings and notable advances of GAs in his book "Genetic Algorithms in Search, Optimization and Machine Learning". As indicated in his book, genetic algorithms are finding applications in business, sciences, and engineering. They are gaining popularity and wide acceptance because the algorithms are "computationally simple, yet powerful in search for improvement". (Goldberg 1989)

Presently, the study of genetic algorithms is considered as part of evolutionary computing which is a growing area of artificial intelligence (Obitko 1998)

3.1 Advantages of Genetic Algorithms

Because of genetic algorithms' simple formulation and flexibility in stipulating parameter constraints, the technique can be applied in almost all optimization problems. It is extremely useful in searching the optimal or at least the near-optimal solution, in single objective optimization problems. Unlike the traditional gradient-based search techniques, which usually fail to find an optimal solution due to the non-continuity, non-linearity and multi-modality of the objective function, the genetic algorithms are affected by none of these difficulties. The genetic algorithms require neither the linearization of the objective function nor calculation of partial derivatives.

In Holland's own words, "GAs offer robust procedures that can exploit massively parallel architectures and ..... provide a new route toward an understanding of intelligence and adaptation." The parallel architectures, which enable GAs to search a multi-dimensional solution space efficiently, will be discussed in Chapter 5.

3.2 Genetic Algorithms' General Applications and Usage in Civil and Water Resources Engineering

The name *genetic* usually causes the first-time reader to link GAs with biology. However, genetic algorithms have been extensively used in economics, psychology, linguistics, image enhancing, computer science, and all aspects of engineering.

For example, John Deere has been using genetic algorithms to optimize its production scheduling of 90 seed planter models to select from about 1.5 million possibilities. The application of genetic algorithms reduces the planning time for a weekly production scheduling from one day to "literally a zap of screen". To design a computer chip on the smallest piece of silicon possible, Texas Instruments used genetic algorithms to reduce the computer chip size by 18%. General Electric has also used genetic algorithms to increase efficiency for gas turbine design which later became the engine for the Boeing 777. US West uses GAs to design fiber-optic cable network and reduce the design time from two months to two days. Genetic algorithms have also been used for positioning cellular telephone towers that provide the maximum coverage with minimal or no overlap (Begley 1995; Frey 1995).

Genetic algorithms have also been found extremely applicable in civil and water resources engineering. Goldberg and Kuo (1987) applied the concept of genetic algorithms in designing a gas transmission pipe at minimum capital cost while meeting all the demand and operational requirements. Tolson (2000) reported that GAs have been applied to almost every type of optimization problem encountered in water resources. In the field of water distribution, genetic algorithms are applied for optimal pipe design, rehabilitation and least-cost pump scheduling. Savic and Watler (1997) developed a GA integrated computer model, GANET to design a least cost pipe network with a complete fulfillment of pressure and demand requirements. Wu et al (2000) applied a competent variation of genetic algorithms, called fast messy genetic algorithms (fmGA) in a decision support system to identify least cost design, rehabilitation, and expansion options for water distribution systems. Karney (2000) used GAs to back-calculate the

frictional factor and elasticity modulus of pressured pipes in water hammering analysis. He called his approach "The Inverse Model Calibration Method".

In water resources management, King, Fahmy and Wentzel (1997) applied a genetic algorithm to the problem of optimizing the operation of a river/reservoir system for maximum economic returns. Tolson (2000) applied GAs to minimize the cost of wastewater treatment for known point source of pollution along the Williamette River in Oregon while maximizing the water quality performance indicators.

3.3 Genetic Algorithms as an Optimizer for Watershed Model Calibration

Currently, the author is not aware of any well-established watershed (hydrological) models using GA techniques in the model calibrating process. Nevertheless, genetic algorithms can be suitably used as the optimization technique in search of the best set of modeling parameters so that the model yields accurate prediction. Thus it is the goal of this thesis to use genetic-algorithms as the basis for an automatic calibrator for the UBC Watershed Model.

James (personal communication) has indicated that a genetic-algorithms based model calibrator for the Storm Water Management Model (SWMM) is being developed. This demonstrates that more watershed (hydrologic) modelers are beginning to take advantage of the efficient GA search and implement the techniques for their model calibration.

4.0 FUNDAMENTALS OF GENETIC ALGORITHMS AND THEIR OPERATIONS

The basic genetic algorithm principles were inspired by the mechanism of natural selection in a competing environment where the stronger or the "fitter" individuals of a species are more likely to survive and produce offspring which survive generation after generation. The weaker or the "less fit" individuals, on the contrary, either die off and become extinct or are forced to evolve and adapt to the non-stationary, competing environment. As observed in nature, by continuous evolution, the weaker individuals do stand a chance to become stronger and therefore may be granted the rights to continuously exist in the population. Usually, as commonly seen in the animal kingdom, a weaker individual can become more adaptive, resilient and stronger through mating and mutation. These genetic operations are the fundamental key components that successful genetic algorithms possess, except in a much-simplified fashion. To develop a genetic-algorithm based optimizer or calibrator, one must understand some of the not-so-simple genetic-algorithm terminology and use the procedures discussed in the following sections. A glossary of common genetic-algorithm terminology is provided in section 1.

4.1 Brief Glossary of Genetic Algorithms Terminology

Coding: A system used to compactly store a set of model input parameter values. In most cases, coding implies the binary coding which is commonly used for genetic algorithms.

Crossover: One of the three major types of genetic operation in which a selected parent string mates with another selected parent string and they exchange genetic information at certain gene locations to form one or two offspring.

Elitism: One type of genetic operation in which the best one or two of the candidate strings of a generation are by default automatically reproduced for the next generation.

Fitness: The evaluation of a complete set of parameter values in the objective function of an optimization problem.

Gene: A coding bit which is the smallest unit of a substring or a string. If binary coding is used, a bit is either 0 or 1.

Generation: This term can be used in two ways. Firstly, as a noun, it collectively means a population of sets of model input parameter values in the evolutionary process. Secondly, as a verb, it is a time step in the evolutionary cycle in which every complete set of model input parameter values, coded as a string, may be genetically altered in order to improve fitness.

Genetic Algorithms: Search procedure based on the mechanics of natural selection and natural genetics (Goldberg 1989).

Mutation: One of the three major types of genetic operation in which every gene of a selected parent string is assigned a uniform probability to randomly alter the bit value. The parent string with randomly altered genes becomes the new offspring.

Niching: One type of genetic operation used to prevent a large number of strings from simultaneously searching a high-fitness region of the search space. By preventing the clustering of GA search points, it forces other unexplored regions to be searched.

Population: A user-assigned (fixed) number of sets of model input parameter values in every cycle of the evolution process.

Reproduction: One of the three major types of genetic operation in which the offspring produced is simply the duplicate of the parent string with identical genetic information.

Scaling: One type of genetic operation in which the fitness of high performance strings is under-emphasized to prevent rapid loss of gene diversity in the string population. It is

also used to over-emphasize the difference between strings of similar fitness so that an optimal string can be identified quickly.

String: A complete coded set of input parameter values for a model, also known as a candidate solution. It is called a string because the set of parameters is (binary) coded and resembles a DNA string.

Substring: A coded input parameter value, which forms the basic building block of a string.

4.2 Brief Outline of the Genetic Algorithm Search Process

The following provides a rudimentary overview of how a modeler can apply GAs in model calibration to achieve high model agreement.

1. Randomly generate sets of model parameters, called strings, which become the initial *population*. The string may contain *genes* and *substrings* that are potentially useful in creating the optimal *string* (best candidate solution).
2. Calibrate a model by optimizing its agreement with the observed data, run the computational model, in this case the UBC Watershed Model, and the statistical measure module to determine the model performance for each set of parameters. The statistical measure values are the *fitness* values.
3. Based on the fitness values, the fitter (better) sets of parameter values (*strings*) are retained via several possible selection schemes for creation of other sets of parameter values - *population of the next generation*. The fitter strings selected are called the *parent string*.
4. A new generation of strings is then created from *parent strings* using various mating and mutation techniques in genetics. Pairs of selected parent strings exchange genes that form new substrings and create new strings with some inherent genetic characteristics and diversity.

5. Now return to Step 2, and the process continues until one of the prescribed termination criteria is met. The iterative process allows genetic algorithms to search for an optimal or a near optimal string (set of model parameters).

4.3 Choosing A Coding Representation

Because an optimization problem, such as a computational model, can possesses descriptive model parameters in the order of tens or even hundreds, there must be an efficient and compact way to represent the real values of all these parameters. This is especially true when using genetic algorithms for optimal solution search and model calibration, because the iterative process requires the values of these parameters to be tracked for generation after generation. The difficulty and complexity of tracking unique genetic information for each string in a large population for many generations requires a convenient coding system to be designed and implemented.

The first step in the development of genetic algorithms is to choose a coding system to represent each possible set of model parameters, often referred to as a *string* or a *candidate solution*. Depending on the coding system used, sets of parameter values may be conveniently and efficiently transformed into strings of bits of particular lengths, where the string length is measured in numbers of bits. In GAs, a string often consists of many substrings, which individually represent values of their corresponding parameters. For example, a string coded to represent a set of 10 model parameters will have 10 substrings to individually represent each model parameter. The position of a substring within a finite-length string determines which parameter it is assigned to represent. Mathematically, a GA coding string is exactly the same as a multi-dimensional vector that can possess components of various magnitudes in different directions of a search space. Hence, a substring's relative position in a string is best viewed as a component of a vector in a specific direction.

For example, a coded string consisting of three substrings can represent a 3-dimensional vector where substring 1, substring 2, and substring 3 respectively represent the vector's magnitudes in i, j, and k axes of the conventional Cartesian coordinate system. By using a coding representation, one allows GAs to treat a set of parameter values as a mapped point in a multi-dimensional search space and store it concisely as a string of finite length.

The most commonly used coding system in GAs is binary coding because of its simplicity and tracibility. As suggested by its name, binary coding only allows two genotypes in a bit which is either 0 or 1. Therefore, a binary string of l bits in length offers 2^l possible 0/1 combinations and can be used to represent a total of 2^l real values. For example, if $l = 3$, then there exist a maximum of eight combinatory ways to arrange the 3-bit string. The eight possible combinations and the real values they represent are listed in Table 4-1.

Table 4-1: Eight Possible Combinations of a 3-Bit Binary String and Corresponding Decoded Real Values

Combination ID	Binary Code	Decoded Real Value
1	000	0
2	001	1
3	010	2
4	011	3
5	100	4
6	101	5
7	110	6
8	111	7

But how is a string used to represent a parameter value and what value does it represent? As shown in Table 4-1, a 3-bit substring can have 8 possible combinations and each of them can be conveniently utilized to represent a real number between 0 and 7. To obtain the implicitly assigned real number, a binary decoding equation is used to convert a string into a real value. For a string S_{l-1}, S_{l-2}, S_{l-3}, S_{l-4}, ..., and S_0, where S_{l-1} and S_{l-2}, are the binary values of bit 1 and bit 2 of the string, the decoded real value can be written as:

$$Decoded\ Real\ Value = \sum_{i=0}^{l-i} 2^i S_i \qquad \textbf{(Equation 4-1)}$$

Table 4-1 also lists the decoded real values represented by the 8 combinations of a 3-bit string.

Mathematically, a binary decoding equation is a series carrying l terms, where l is again the length of a sub-string. The design length of a binary code string is simply derived based on the number of the substrings (parameters) it has to represent and the desirable accuracy of the represented parameter values. Hence individually, the length of a substring required to represent a parameter depends only upon the individual degree of accuracy desired. The accuracy of a parameter is calculated as:

$$Coding\ Accuracy = \frac{(X_{\max} - X_{\min})}{2^l - 1} \qquad \textbf{(Equation 4-2)}$$

where X_{max} = the upper bound of the feasible range of parameter X,

X_{min} = the lower bound of the feasible range of parameter X,

l = sub-string length in number of binary bits

For a substring length of 3-bits, the accuracy of the specific parameter is equal to (X_{max} - X_{min}) / 7. Equation 4-2 can also be rearranged to calculate the substring length, $l_{substring}$, needed to achieve a specified accuracy.

$$Log(l_{substring}) = \frac{\left(\frac{(X_{\max} - X_{\min})}{CodingAccuracy}\right) + 1}{Log(2)} \qquad \textbf{(Equation 4-3)}$$

For example, if a modeling parameter is feasible between 0 and 1000 and one would like to know what substring length is required to achieve a parameter accuracy of 1 or less, then equation 4-3 can be used. It can be easily shown that the substring length should be

at least of 10 bits in order to obtain 1,024 possible combination of a 10-bit string and achieve a corresponding accuracy of 0.976, which is within the accuracy desired.

Once a string is decoded into a real value and the related accuracy calculated, the true parameter value can then be calculated as:

$$X = X_{min} + Decoded\ Real\ Value \times Coding\ Accuracy \qquad \textbf{(Equation 4-4)}$$

Reusing the 3-bit string as an example, one can see that string 000 (or any arbitrary-length binary string containing genotype 0 only) would always yield a decoded value of zero and thus is equal to X_{min} by default. In contrast, the string 111 (or any arbitrary-length binary string containing genotype 1 only) would always yield a decoded value of $2^{l} - 1$ (which can be proven with some mathematical manipulation). Thus, the string 111 or other entirely genotype-1 strings are automatically equal to X_{max} values. The remaining six 3-bit strings vary their true parameter values linearly based on their decoded real values as shown in Table 4-1. This rule applies to strings of any length (any number of bits).

To illustrate the above discussion, assume that the eight 3-bit strings in Table 4-1 are used to represent eight possible evenly spaced parameter values between 1 and 10. Clearly, string 000 has a true parameter value of 1 while string 111 has a true parameter value of 10. From equation 4-2, the accuracy is calculated as 9/7 and the true parameter values of remaining six strings may be computed from equation 4-4. They are summarized in Table 4-2.

Table 4-2: Eight Possible Combinations of a 3-Bit Binary String and Corresponding Parameter Values

Combination ID	Binary Code	Decoded Real Value	True Parameter Values
1	000	0	1.0
2	001	1	2.3
3	010	2	3.6
4	011	3	4.9
5	100	4	6.1
6	101	5	7.4
7	110	6	8.7
8	111	7	10.0

The inherent limitation of the binary coding and any other form of coding system is that mathematically their coding strings can only represent a family of evenly spaced discrete values between the lower and the upper bounds of each parameter in the search space. Therefore, the coding strings at their best only form a discrete function rather than a continuous function over the domain of the feasible search space. Hence, the size of the even spacing between the discrete values is apparently the coding accuracy calculated in equation 4-2.

This limitation of a coding system creates difficulty when one would like to code for a random value in between two discrete values, say 9.35 using a simple 3-bit binary string. From Table 4-2, one can see that 9.35 is the midpoint between the discrete values 8.7 (string 110) and 10 (string 111), and it is not possible to generate another 3-bit string to accurately represent the value of 9.35. To resolve this difficulty, one will commonly increase the string length to reduce the even spacing between the discrete values represented by a family of coding strings in order to reduce the inevitable round off error. In this case, if a family of 4-bit string is used, then the coding accuracy will increase by a factor of 2.2 from 1.3 to 0.6 and consequently the value of 9.35 can be more closely represented by a 4-bit string 1110 which gives a parameter value of 9.4. Therefore, for the degree of accuracy required by the modeling parameters needed in an optimization problem and model calibration, it is possible to satisfy the coding accuracy and error requirement by increasing string length. Nonetheless, for optimization problems that involve only decision variables, the usage of coding string is appropriate and incurs no errors because the variables are exactly represented through coding.

Table 4-3 lists the 16 possible combinations of a 4-bit binary coded string. The value of 9.35 can be represented by string 1110 which indeed represents an exact value of 9.4. However, if the discrepancy of 0.05 between 9.35 and 9.4 cannot be tolerated, the length of coding strings can be further increased to 5 to minimize the error.

Table 4-3: 16 Possible Combinations of a 4-Bit Binary String and Corresponding Parameter Values

Combination ID	Binary Code	Decoded Real Value	True Parameter Values
1	0000	0	1.0
2	0001	1	1.6
3	0010	2	2.2
4	0011	3	2.8
5	0100	4	3.4
6	0101	5	4.0
7	0110	6	4.6
8	0111	7	5.2
9	1000	8	5.8
10	1001	9	6.4
11	1010	10	7.0
12	1011	11	7.6
13	1100	12	8.2
14	1101	13	8.8
15	1110	14	9.4
16	1111	15	10.0

Other coding systems such as tertiary and Gary coding may be used to achieve high coding accuracy without carrying long string length. But they are more complicated and add unnecessary complication to the computer implementation of genetic algorithms. Hollstein (1971) investigated the use of Gary code in GAs and found that it works slightly more efficiently than the conventional binary coding. In the present thesis, only the binary code representation will be discussed because of its wide acceptance among practitioners of GAs and the established theory on GA operations: Schema Theory is developed based on binary code structure.

It is worth noting that binary coding is extremely efficient at representing decision variables in the optimization of a decision model. A decision variable can be simply

represented by one bit with 1 for "yes" and 0 for "no" in the decision making process. In fact, this is exactly the way GAs are used in designing optimal low-cost natural gas transmission or water distribution networks as mentioned in Chapter 3. In GAs, a binary string of 100 bits can represent the decision variable of 100 water supply pipes in the network. If a decision variable is represented by a 0 in the optimal solution, then the pipe is redundant. However, if a decision variable is represented by a 1, then the pipe must be constructed to meet the conveyance criteria.

4.4 Initialization of Genetic Algorithms

The initialization of genetic algorithms involves the random creation of the initial string population, setting of population size, number of generations to evolve, and termination criteria for the GA search to terminate. The following sections provide the details.

4.4.1 Random Creation of Initial String Population

Once a coding system is chosen to represent sets of model parameter values in strings, an initial population of strings must be randomly generated in coded form before any genetic evolution can take place. The reason why the strings are randomly generated is two-fold. Firstly, unless specific information exists, strings (candidate solutions) freely created by a random generator are no worse than any other guesses. Secondly, by randomly creating each string's genetic information for the entire initial population, maximum genetic diversity is achieved in the gene pool. A diverse gene pool is more likely to contain the key genetic information blocks needed to form some high fitness strings and, ultimately, the optimal string.

4.4.2 Choosing GA Population Size and Number of Generations

The concept of genetic diversity leads to the two most frequently asked questions about genetic algorithms:

1. What is the population size needed to maintain sufficient genetic diversity among candidate solutions in order to achieve an optimal, or at least near optimal, solution efficiently, without carrying the burden of long computational time?
2. How many generations of genetic evolution are needed to allow strings (candidate solutions) to mate, mutate and subsequently produce an optimal or at least near optimal solution in relatively short computational time?

There is no absolute rule about choosing the right population size and number of generations to run in GAs. However, some guidelines can be found in the literature. In most cases, the GA user must determine what to use through a trial and error process.

Goldberg (1989) suggests that population size is commonly between 30 and 100, depending on the complexity of the objective function in terms of the degree of non-linearity. However, he also points out that the population required is case specific, and he make no recommendations on the number of generations to run. Carrol (1999) suggests that using a population which is an exponential of two, i.e. 2^N (where N is any integer greater than 1), can enhance GAs' efficiency in finding the optimal solution.

In most GAs reported, the number of strings in the population is almost always fixed for the simplicity of coding and programming. However, there is no written rule that prohibits the use of varying population size in different generations. In fact, the population of a species in nature would vary in every generation, based on the species' performance in a hostile environment. Thus, it is logical to think that if a candidate solution's performance is poor and below a filtering criterion, then it could die without a chance to produce offspring and the population declines. But to the best knowledge of the author, there is no conclusive research reported that shows varying population size

after each generation will help the GA solver to approach an optimal solution more efficiently.

The second dilemma faced in GA initialization is the setting of the number of generations to run. The number of generations in simple terms is the number of evolution cycles required to improve string fitness (optimality) and eventually reach the optimal solution. If the number of generations assigned is small, then a sub-optimal solution near the global optimal solution is more likely to be found instead of the global optimal solution itself. On the contrary if the number of the generations is too large, then the following two problems may be encountered:

1. The computational time may be so consuming that GAs become less practical against other optimization techniques.
2. Genetic diversity of the candidate solutions may be lost and the search zeros in on a single false or non-optimal solution. Fortunately, this problem can be avoided by using techniques that will be explained later.

Hence, in order to avoid unnecessary searching and reduce computational time, a set of termination (stopping) criteria should be used to end the GA search.

4.4.3 Termination (Stopping) Criteria of GA Search

Typically, the first termination criterion is to stop the iterative GA search after the maximum number of generations prescribed in advance is reached. The prescribed value is usually weighted by the upper limit of the acceptable computation time desired. The limitation of this type of termination criterion is that it takes no consideration of whether a near-optimal solution is found or not.

The second termination criterion is to stop the GA search if the optimal solution has apparently converged. But how does one know whether a convergence has been reached and an optimal solution found?

Under some special circumstances, the upper limit of the objective function of an optimization problem is known. For example, in the UBC Watershed Model, the theoretical upper limit of a possible objective function for model performance, *e!*, is +1. Thus, the second termination criterion can be stipulated to stop the GA search if the best performing candidate solution of the continuously evolved population achieves a fitness value close to +1, say 0.95, for example. If an *e!* value of 0.95 is not obtainable in the model calibration, then the second criterion is not satisfied. The GA search will go on until the first criterion is satisfied.

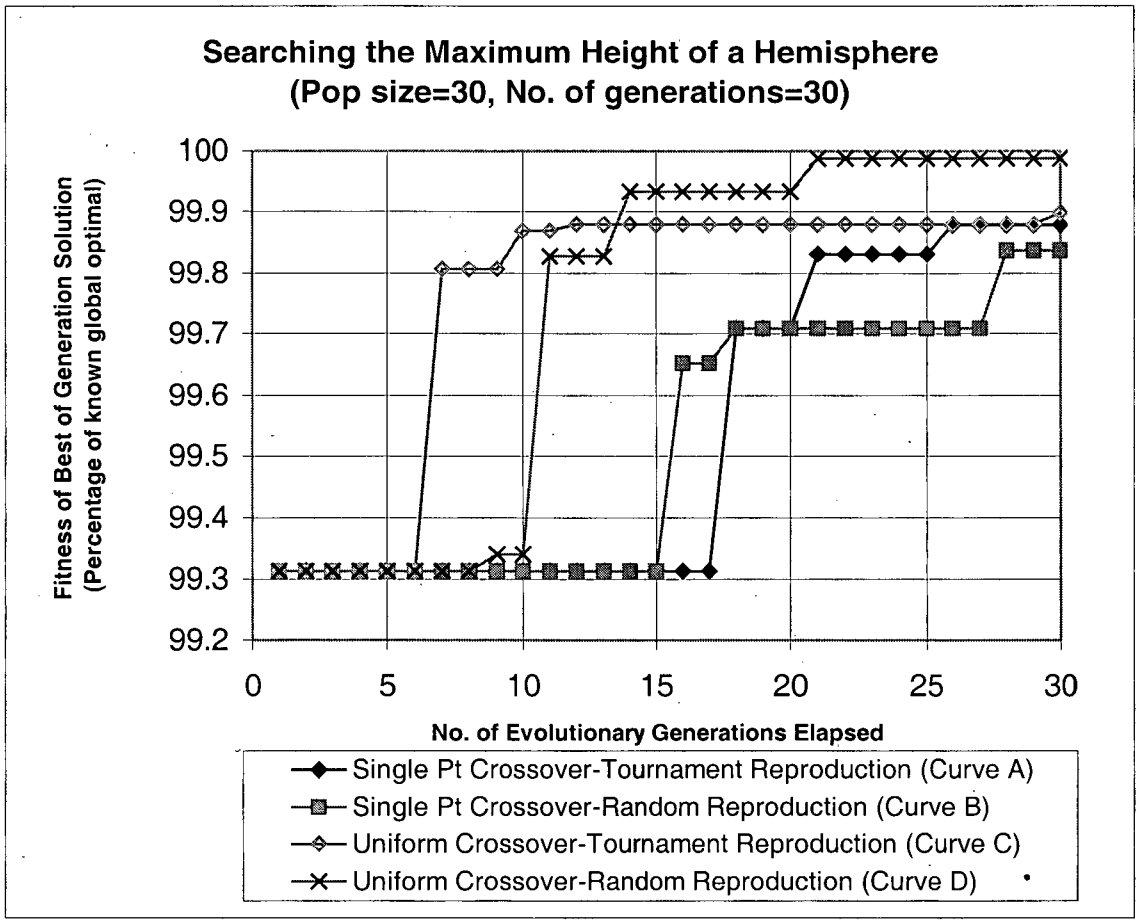
However, under most circumstances the optimal objective function value is not known and needs to be explored. Therefore, the GA search may be set to stop if temporal convergence is reached. The term "temporal convergence" means that the best objective function value achieved by a continuously evolved population of candidate solutions ceases to improve significantly generation after generation. For example, the GA search can be stopped if the best fitness value achieved by a population does not improve significantly for ten consecutive generations.

It is important to emphasize that when a temporal convergence is reached; the candidate solution found may not necessarily be the optimal. The cruel reality GA developers and users have to face is the fact that in a non-continuous, discrete, multi-modal search space of multi-dimensionality, no solution can be analytically proven as the global optimal solution.

Figure 4-1 is a simple example of GAs performance in finding the maximum height (the objective function value) of a hemisphere. One can see that the curve D achieves the ultimate convergence at 99.98% of the known maximum height while curve C reaches a temporal convergence at 99.88% and consequently outputs a non-optimal solution (if the

number of generations simulated is limited to 29 or less). This example will be discussed in the next section.

Figure 4-1: An Example of Temporal Convergence, Curve C in Searching the Maximum Height of a Hemisphere



In summary, two sets of termination criteria should be used in the initialization process of GAs. Whenever one of the termination criteria is satisfied, the GA search will be ended and the candidate solution with the highest objective function (fitness) is deemed to be the optimal solution.

Due to the sheer number of strings used in the population of GAs and the number of generations that are required, it is not difficult to understand why GAs were only implemented since the 1970s after the computer became the main computational tool in

research. Without the high speed of current state of art personal computer, the research boom in genetic algorithms would not have been possible.

4.5 Designation and Evaluation of Fitness (Objective Function)

Once an initial population of candidate solutions in the form of genetic strings is randomly generated, each string is then decoded to real parameter values and its fitness value evaluated against the designated objective function. The formulation of the objective function can be direct or indirect depending on how easily it can be calculated. In a simple single-objective optimization problem, the objective function can be an explicit and concise formula; thus it can be embedded into the GA program code as a subroutine. However, in cases in which GAs are used to calibrate models, the objective function used to statistically measure the model performance can be complicated and should be treated as a separate module, independent from the GA program code. In the GA calibration process of the UBC Watershed Model, the GA program code repeatedly calls the UBC Watershed Model to generate a set of simulated streamflow and calculates the respective statistics such as *e!* for every string to find out its corresponding fitness. The resulting fitness value of a genetic string is important because it determines whether this candidate solution can be selected as a parent to produce offspring in the immediate next generation. Typically, the string with a higher fitness will have a higher probability of being selected and thus more likely to produce offspring while the lower fitness strings will have low probability.

4.6 Selection of Parent Strings (to Produce Offspring)

The two most commonly used selection methods to choose parent strings for the creation of the offspring in the next generation are the weighted roulette wheel and tournament selection methods.

4.6.1 Weighted Roulette Wheel (Fitness Proportionate) Selection

In this method, a biased roulette wheel is virtually created using a computer algorithm to artificially implement the concept of survival of the fittest. In mimicking what is observed in the natural realm, high performance strings accordingly have high probability of producing one or more offspring with their genetic traits.

Mathematically, the circumference (or the total area) of the roulette wheel represents the sum of fitness measures of the entire string population whereas the arc-length (or the sector area) of each slot in the wheel represents the individual string fitness. In other words, the size of each slot of the wheel is allocated proportionately to individual string fitness, and the number of slots is equal to the population of the strings. Therefore, the higher fitness value a string has, the bigger slot it occupies on the roulette wheel, and thus the more likely it is to be selected. Statistically, the most likely number (N) of offspring a parent string would contribute to create is:

$$N = n \cdot \frac{f_{individual}}{\sum_{i=1}^{Pop.} f_i} = \frac{f_{individual}}{f_{average}} \qquad \textbf{(Equation 4-5)}$$

where n is the number of strings in a population of candidate solutions

$f_{\text{individual}}$ is the fitness of a parent string

$f_{i=1 \text{ to } Pop.}$ is the fitness of every string in the population

f_{average} is the average fitness of the entire string population

Note that $\frac{f_{individual}}{\sum_{i=1}^{Pop.} f_i}$ is the probability of a string to be selected for mating; it is therefore always less than 1.

4.6.2 Tournament Selection

This method requires two parent strings to be selected at random regardless of their respective fitness. Once selected, the two strings are then compared on their fitness and the higher performance string becomes the parent string for the production of offspring in next generation. Thus, to select two parent strings for mating, four random selections must be conducted. Although this method is biased towards high fitness parent strings, it does not dictate the production of offspring based on the ratio of individual string fitness to total fitness of string population.

4.7 Genetic Operations

Upon the selection of the parent strings, they are used to produce offspring via genetic operation. The conventional genetic algorithms consist of three operators mimicking the evolutionary processes observed in nature. They are reproduction, crossover and mutation. More advanced operators have also been used with success, but the three main operators are efficient for searching for near optimal solutions in a multi-dimensional space. Because these genetic operators are probabilistically selected for use, each of them is assigned a respective probability. The probabilities of reproduction (P_r), crossover (P_c) and mutation (P_m) are usually assigned based on empirical results. Again, there are no absolute rules, but only guidelines in setting the probabilities of genetic operators.

4.7.1 Reproduction

The reproduction operator allows a parent string to produce a duplicate offspring with identical genetic information. Although this operator appears to be too simple and redundant, the key importance of this operator in GAs is to preserve the incumbent high performance strings for mating in future generations rather than in the immediate next generation. Without this operator, the incumbent high performance string may be

completely destroyed by other genetic operators such as crossover (to be discussed) and denied the chance to mate with strings of the future generations, yet to be born.

The probability of a selected parent string to reproduce itself for the immediate next generation, P_r, is often decided by the GA user. Depending on the complexity and nature of the optimization problem or calibration process involved, the importance of the reproduction operator's role in creating new offspring may vary. P_r is often between about 0.1 and 1/3. While higher P_r values will slow the GA search efficiency because the offspring are identically the same as the parents, low P_r values will also prolong the required search time because of the destruction of high fitness strings. In common GA program code, P_r is usually not set explicitly because P_r is equal to $1\text{-}P_c\text{-}P_m$ once the P_c and P_m values are agreed upon.

Based on the chosen P_r value, a portion of the selected parent string (using the roulette wheel scheme) will be given the reproduction operation. Thus, probabilistically the approximate number of genetic strings to remain totally unchanged in a generation of genetic operation is

$$N_{reproduction} = Population * P_r \qquad \textbf{(Equation 4-6)}$$

Since reproduction does not change a string's fitness at all, tournament selection can be imposed to further increase the average fitness of strings reproduced for next generation. This is achieved by selecting two strings using the roulette wheel selection method first and then only admitting the string of higher fitness to the population of the next generation. Because the fitness of every string has been already evaluated and stored in a numeric array, the fitness of the two strings selected for tournament can be directly compared without calling the objective function subroutine or rerunning the computational model for fitness.

Figure 4-1 compares the use of the reproduction operator with and without tournament selection in terms of search efficiency. A reproduction operation without tournament

selection (Curve B) performs better than a reproduction scheme with additional tournament selection (Curve A) when a single crossover operator is also used. However, the opposite is true when a uniform crossover operator is used. The definition of a crossover operator and its various types are the subject of the next section.

4.7.2 Crossover

The crossover operator allows a selected parent string to mate with another selected parent string and exchange genetic information at one or more gene location(s) of the binary strings to form one or two offspring. Regardless of the number of the offspring formed, the offspring strings should inherit a portion of genetic information from each parent. The intent of using a crossover operator in genetic algorithms is to divide genetic information into building blocks or even as small as genes (bits) and hope a systematic recombination of the building blocks will lead to higher fitness strings or candidate solutions.

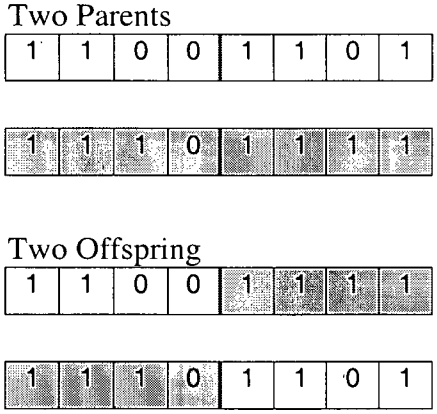
Typically, crossover operation can be categorized as one of three types:

1. Single point crossover
2. Multi-point crossover
3. Uniform crossover

A gap in between any two gene-positions about which the genetic information of two parent strings can exchange is called a cutoff point. The number of cutoff points used in the crossover operation determines the crossover type. In a single point crossover, a cutoff point is usually randomly chosen along the full string length and the genetic information of two parent strings is exchanged about this cutoff point. An example of single point crossover is provided in Figure 4-2. For a string length with L bits, there are $L-1$ bit-linkages that can become cutoff points. Thus the probability of a binary linkage to be selected as a cutoff point is $1/(L-1)$. Depending on the GA user's preference, the

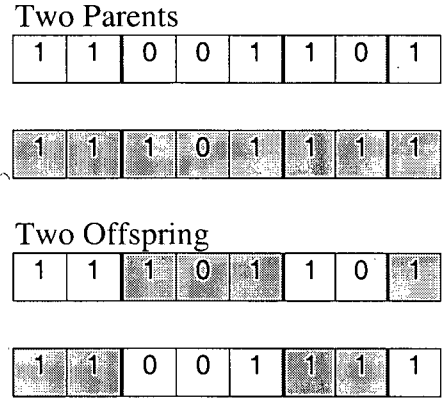
random cutoff locations may also be limited only to the linkages of substrings, with each substring representing the value of each parameter in the optimization problem or model calibration.

Figure 4-2: Single-Point Crossover of Two Selected Parent Strings



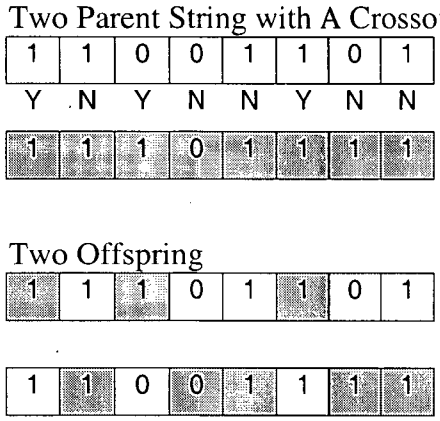
A multi-point crossover or uniform crossover often replaces the single-point crossover when the string length becomes too long for a single point crossover to efficiently explore large numbers of recombination of the parents string to obtain better offspring. Similar to the single point crossover, cutoff points are also randomly selected in multi-point crossover and the two parent strings alternatively exchange their genetic information about these cutoff points. An example of multi-point crossover is provided in Figure 4-3.

Figure 4-3: Three (Multi)-Point Crossover of Two Selected Parent Strings



In the uniform crossover, every binary bit is given a significant probability (usually 0.5) to switch genetic information with its position-based counterpart of the mating parent string. Therefore, every binary bit linkage can be a cutoff point for crossover. The maximum number of cutoff points that may take place concurrently to a binary string of length L in uniform crossover is $L\text{-}1$. However, whether or not the two selected parents strings switch genetic information at a particular binary bit position is probabilistic and can be decided by flipping a coin L times, or by applying a random binary integer generator with 1 indicating yes and 0 indicating no. The blueprint for which bits to exchange or not to exchange is referred as the crossover mask. If a probability of 0.5 is used, then the average number of bits to be switched in uniform crossover operation is $L/2$. Figure 4-4 is an example of uniform crossover for a given crossover mask. Notice that in bit positions 1 and 6, the genetic information exchanged is identical; thus the crossover at these two positions has no overall effect in differing the fitness of offspring from the parent strings. Therefore, in this example, without an additional exchange of the binary bit in position 3, the offspring produced in this uniform crossover operation will be identical to the two parents.

Figure 4-4: Uniform Crossover of Two Selected Parent Strings with a Randomly Generated Crossover Mask



Despite the fact that the effect of crossover sometimes may not be very pronounced due to very similar parents strings, most of the time the crossover operator creates new offspring with profound implications. The creation of new offspring implies that two additional points in the solution space are to be searched and tested. Thus an effective

usage of the crossover operator will greatly enhance the search efficiency of GAs. This leads to the discussion on setting values for P_c, the probability of crossover.

The mixture of genetic information from both parents through crossover creates offspring that may out-perform or under-perform both parents. In general, the values for P_c should be between 0.6 and 0.8. Goldberg (1989) suggested a P_c value between 0.6 and 1. An increase in P_c would positively increase the number of recombinations of genetic building blocks and enhance the chance of finding the optimal string (solution), but it inevitably destructs the reasonably good strings already in existence. This is especially true if the P_c is relatively high. If a P_c value of 1 is used, it is doubtful that the chance of finding a better near optimal string will outweigh the impact of losing the best solution found so far. In order to preserve a good string, the reproduction operator and advance operators, such as elitism, should be used. The elitism operator will be discussed in a later section.

The choice of types of crossover operators to use is less controversial. In general, the more crossover cutoff points that are available, the better the GA search efficiency. For example, Figure 4-1 in section 4.4.3 shows that a uniform crossover operator tends to outperform a single-point crossover operator. For the calibration of two watersheds using the UBC Watershed Model, experiments conducted for the present thesis show that the uniform crossover tends to converge the fastest and generate the best results.

4.7.3 Mutation

Mutation is one of the three major types of genetic operation in which every gene of a parent string is assigned a uniform probability to randomly alter the bit value. The probability of mutation is usually small as observed in nature. The parent string with randomly altered genes then becomes the new offspring. If strings are binary coded, a mutation operator will change bit value of a gene from 1 to 0 or vice versa. The mutation

process is deliberately given a small probability so only a limited number of genes are altered. Otherwise, given higher probability, the process can turn into a random search.

Mutation can be implemented implicitly or explicitly. In the implicit scheme, the mutation operator is embedded in the reproduction and crossover operation. After an offspring string is created via reproduction or crossover, a small uniform bit-wise mutation probability is assigned to each gene of the string. The typical bit-wise mutation probability, P_m' used is between 0.01 and 0.02 (Carroll 1996). Note that the apostrophe indicates the mutation probability is bit-wise. Thus for a string with a length of 20 bits, the probability of having at least one gene mutation along the full string length is between 0.2 and 0.4. As the string length increases (say to a length of 100 bits), bit-wise mutation will almost certainly occur.

In the explicit mutation scheme, the mutation operator is probabilistically selected for use just like reproduction and crossover operators. For example, to create offspring from selected parent strings, the probabilities for reproduction (P_r), crossover (P_c) and mutation (P_m) can be assigned as 0.3, 0.6 and 0.1, respectively, thus the mutation operator will be used about 10% of the time. When the explicit mutation scheme is used, the sum of P_r, P_c, P_m should add up to 1. To clearly differ from the reproduction operator, the GA code is arranged so that at least one bit mutation will occur somewhere in the string. For example, if a 100-bit string is selected for explicit mutation instead of reproduction or crossover, each gene will have a bit-wise mutation probability of 0.01 (the reciprocal of 100). Users can also specify the number of bit mutations to occur depending on their preference. Nonetheless, allowing a large number of bit-wise mutations can turn the GA search into a random search. The value of P_m for the explicit scheme ranges from 0.2 to 0.4 depending on the parent string selection method used. The advantage of having a high P_m in both mutation schemes is to reintroduce genetic diversity that is lost when a selection method such as the roulette wheel scheme purges the strings with low fitness. However, the disadvantage is that a high P_m value will easily transform the GA search into a random search by altering the parameter (substring) values at random.

Because the alteration of a gene is probabilistic, in the implicit mutation scheme there is not a concrete rule preventing the change of more than one bit. The position of a mutation in a string is randomly chosen based on a uniform probability distribution along the full string length. In the explicit mutation scheme, one or two bit mutations are usually pre-specified, and a random generator is applied to determine the position of bit mutation in a string. Thus, in both mutation schemes the offspring created through mutation often bears close resemblance to the original parent strings. The intent of mutation is, therefore, to allow small-scale genetic information alteration (compared to large-scale alteration in crossover) in the hope that high performance strings (candidate solutions) can be found through local and minor genetic improvement. For example, at a particular position of a string, the current value is 0 and this value needs to be changed to 1 to form the optimal string (i.e. string which would achieve optimal objective function), mutation is the best means to change. Neither reproduction nor any form of crossover operators can accomplish the change from 0 to 1 without the risk of disturbing any of the existing genetic information in other positions of the string.

In addition to finding a possible optimal solution through local gene change, mutation is used to continuously insert changes into candidate strings to maintain a high level of diversity in the gene pool. It is important to emphasize that because the roulette wheel selection method chooses parent strings according to their fitness, the gene pool diversity is gradually reduced as less fit strings die out and become extinct. So mutation is useful in maintaining genetic diversity in the population.

4.7.4 More Notes on Choosing Probabilities of Genetic Operators

In the process of genetic operation, the question arises of how these genetic operators are probabilistically selected for use. The reproduction probability is usually set to about 0.1 to 1/3. The crossover probability is usually set to about 0.6 to 0.8 or even to 1. The mutation probability normally ranges from 0.2 to 0.4 in explicit scheme and 0.01 to 0.02 for bit-wise mutation in implicit scheme. If only the reproduction, crossover and explicit

mutation operators are used, the sum of their respective probabilities should be 1. If the implicit mutation scheme is used, the sum of P_r and P_c should be 1.

The choice of P_r, P_c and P_m (or P_m') is problem specific and their assigned values depend upon the complexity of the objective function of the optimization problem or the measure the model performance. Thus, to achieve high GA search efficiency the choice of P_r, P_c and P_m (or P_m') may require several sensitivity analyses. In practice, the probability values are often initially set based on some guidelines suggested in the literature and then they are fine-tuned by trial and error from empirical GA search efficiency results. The distribution of the probabilistic weights among the three major operators is a subject that remains inconclusive and needs to be further investigated by GA researchers.

4.7.5 Elitism

Elitism is a genetic operation in which the best one or two of the candidate strings of a generation are by default automatically reproduced for the immediate next generation without using any parent string selection method. The intent of using the elitism operator is to preserve the solution that ranks high in terms of fitness and prevent crossover and mutation operators from accidentally destroying the best solution found so far in a search. If elitism is used, the best-of-generation fitness will always be equal or greater than the fitness value of the previous generation because the best string always survives.

De Jong (1975) found that the elitism operator improves the average fitness of strings for a uni-modal objective function, but decreases the performance of strings for a multi-modal objective function. However, in the GA calibration of the UBC Watershed Model conducted for the present thesis, elitism was found to greatly increase average fitness of strings despite that the objective function to be optimized in model calibration is indeed multi-modal.

4.7.6 Scaling

Scaling is a genetic operation in which the fitness of high performance strings is under-emphasized to prevent rapid loss of gene diversity in population. It is also used to over-emphasize the difference between strings of similar fitness so that an optimal one can be identified quickly. In GAs where the roulette wheel selection method is used in choosing parent strings for the creation of offspring, it is possible that a few very high performance strings are overly selected as parent strings due to their high fitness. These very fit strings crossover with each other, and may produce offspring identical to the parents, which severely reduces the diversity of gene pool of the population. This phenomenon is known as *premature convergence* because the identical offspring produced due to lack of gene diversity can easily mislead the GA users to believe that a converged solution near the global optimal has been achieved. To prevent the premature convergence in GA search, a scaling operator is used to reduce the weight of high performance strings rather than allocating the weights directly proportional to their fitness. An example of simple linear scaling transformation of raw fitness values is shown below:

$$f' = a \cdot (f - f_{average}) + f_{average} \qquad \textbf{(Equation 4-7)}$$

where f is the fitness of a particular string to be scaled

$f_{average}$ is the average fitness of the entire population

f' is the scaled fitness

a =a scaling parameter which is usually less than 1.

Applying equation 4-7 with values of $a < 1$, strings with high fitness will be assigned smaller fitness and allocated smaller slots in the roulette wheel selection. Table 4-5 summarizes the scaled fitness of the five strings listed in Table 4-4 using a scaling parameter of 0.7.

Table 4-4: Fitness without Scaling

String ID	Raw Fitness	No. of times selected after 100 roulette wheel turns
A	1.5	13
B	2.0	17
C	1.0	9
D	5.0	43
E	2.0	17

Table 4-5: Scaled Fitness

String ID	Scaled Fitness	No. of times selected after 100 roulette wheel turns
A	1.7	15
B	2.1	18
C	1.4	12
D	4.2	36
E	2.1	18

At the first glance, the changes may appear small and insignificant. However, as the number of generations increases, the accumulative effect of scaling will become significant and pronounced. Note that it can be proven that the value of $f_{average}$ remains unchanged in the linear scaling transformation.

The scaling operator is also useful in facilitating the search of the optimal from a group of near-optimal solutions when premature convergence is not a concern. For example, if the fitness of all strings is very close in magnitude, then each parent string would be selected a roughly equal number of times and the roulette wheel selection would lose its bias towards the better strings. Under these circumstances, it is necessary to emphasize the superiority of one string over another by using the scaling operator. This can be easily achieved by applying equation 4-7 with a coefficient $a > 1$ to magnify the fitness differences among parent strings.

4.7.7 Niching

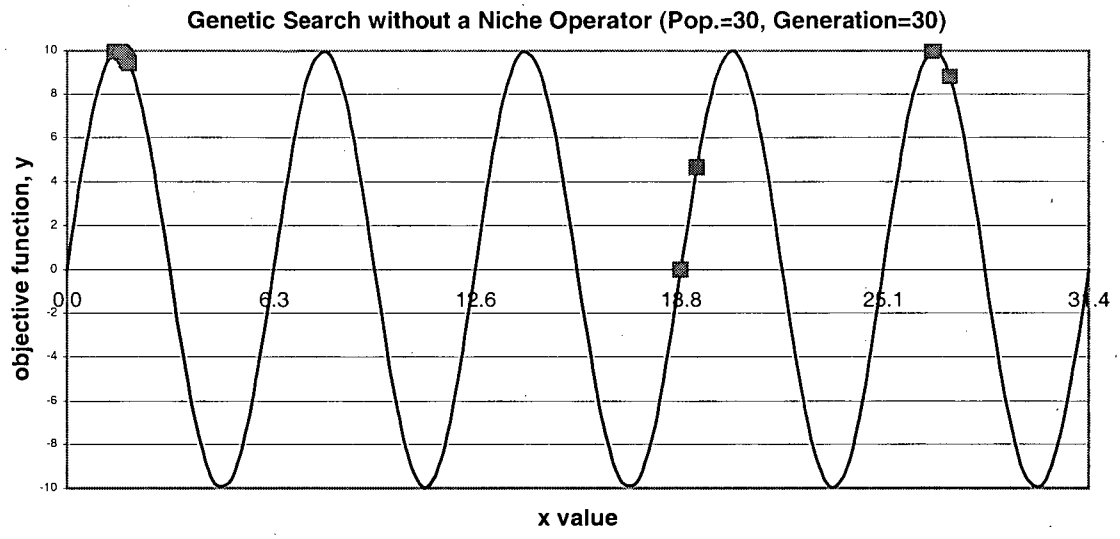
Niching is a genetic operation in which the clustering of a large number of strings in a high fitness region of the search space is intentionally prevented. A GA optimizer which uses the major genetic operators discussed until now is sufficient if the optimization problem is single-objective and the objective function is uni-modal, or the objective function is multi-modal but only the global maxima (minima) is of interest. However, sometimes it is important to know the values of all maxima (both the global and local ones) and their corresponding parameter values (locations in multi-dimensional search space). Or sometimes, the single-objective function may be multi-modal but with peaks of equal magnitude which need to be simultaneously identified. Under these circumstances, the GA optimizer must be further improved with a niching operator.

For example, if the objective function of a single-objective problem can be approximated as a sine function with five equal peaks, applying genetic-algorithms search technique without a built-in niching operator will yield the location of only one of five equal peaks and indicate nothing about the presence of the remaining four peaks. This phenomenon is known as the *"genetic drift"* caused by the intended bias in the weighted roulette selection of the fitter parent string. The *"genetic drift"* occurs when only one of the equal peaks or near equal peaks is first found during the search and the peak receives overwhelming weight in the roulette selection process. In other words, as soon as a relatively high peak or an enclave of high fitness is found, the strings (candidate solutions) in other sub-domains where other equal peaks can be found will die off to make room for new-born offspring of the newly found peak. Therefore, without niching, after running GAs for a number of generations, the entire string population would tend to cluster at one location rather than scattering at various locations.

Figure 4-5 provides the GA search results for a simple sine function y=10*sin(x) without a niching operator. Of the thirty candidates created to search, twenty-five candidate solutions landed near the first peak (x=3.14) while only three landed near the fifth peak

(x=26.70). Therefore, although two equal-magnitude peaks were identified without niching, the fifth peak could have been easily overlooked by GA search without niching.

Figure 4-5: Genetic Search without a Niche Operator for a Simple Sine Function



Hence, the niching operator can be viewed as a fitness sharing function which discourages the clustering of candidate solutions near an already found optimum or a particular sub-domain of high fitness, and encourages the continuous search of other possible peaks that may be present as well. The niching operator used in the GA code of this thesis is a sharing function which reduces the individual objective function based on the number of candidate solutions clustered within a nearby small sub-domain. The sharing function in one-dimensional form can be expressed as:

$$f_{niching} = \frac{f_{raw}}{\sum_{j=1}^{n=pop.} \left(\frac{X_{\max} - X_{\min}}{X - X_j} \right)^2} \qquad \textbf{(Equation 4-8)}$$

where f_{raw} is the evaluated fitness of a particular string before niching

$f_{niching}$ is the modified fitness of the particular string after niching

X_{max} = the upper bound of the feasible range of the one-dimensional parameter X,

X_{min} = the lower bound of the feasible range of the one-dimensional parameter X,

X = the value of parameter X for the particular string,

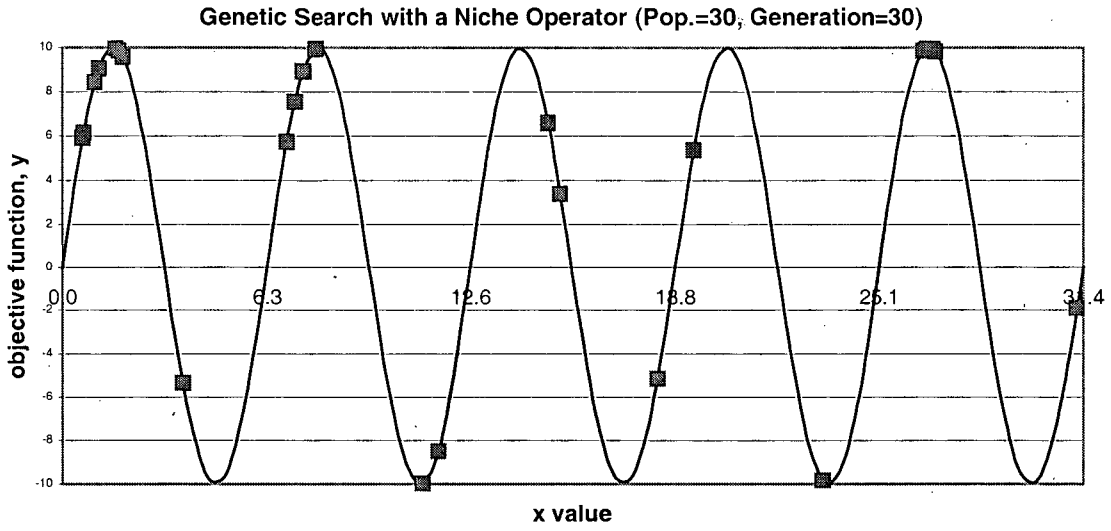
X_j = the value of parameter X for all other strings in the population.

Equation 4-8 forces a broader search away from the known high fitness region. Therefore, based on equation 4-8, a candidate solution with a lower fitness (objective function), but with no other candidate solution nearby will have a higher chance of survival than a candidate solution with a high fitness (objective) and many candidate solutions nearby. By using the niching operator, the GA search is armed to prevent undesirable premature genetic clustering, referred to as genetic drift in GA terminology, which may mislead users to a sub-optimal solution. However, the niching operator by no means implies that no candidate solution is allowed to stay near the already found peak. In conclusion, one may say that the word "niche" used in GAs means that stable numbers of sub-populations of strings (candidate solutions) should search various sub-domains of the objective functions and stay there even if the maxima found by them are local ones, regardless of additional higher maxima being found somewhere else in the domain of the objective function.

Figure 4-6 provides the GA search results for a simple sine function y=10*sin(x) with a niching operator. Three peaks (first, second and fifth) were clearly identified and the remaining two peaks would be identified if additional generations of simulation were given. Candidate solutions were also less clustered compared to the ones in Figure 4-5. Therefore, the niche operator has served its purpose of identifying almost all the equal-magnitude peaks.

One interesting observation worth noting is that the GA experimental results show that when using a niching scheme, the average non-niched fitness of the entire string population is usually less than the average fitness of the strings obtained without niching. The slower search efficiency is a trade-off for the added ability to map the behavior of the objective function in more detail.

Figure 4-6: Genetic Search with a Niche Operator for a Simple Sine Function



4.7.8 Other Operators

Besides the fundamental genetic operators mentioned, several other GA operators have been devised, based on observed genetic processes in nature, with the hope of improving the robustness of GA search. However, the success of these operators remains relatively limited and tends to be case specific. Two of the better-known operators that fall into this category are dominance and inversion.

Until now, every candidate solution discussed is coded in one single string as the single-stranded chromosome of uncomplicated forms of life observed in nature, (Goldberg 1989). But what if one attempted to code a candidate solution with two strings similar to a double-stranded chromosome found in more complicated life forms in nature? What advantage in GA search efficiency can be achieved by using a two-string system to represent a candidate solution? Under the two-string (double-stranded) system, each string of a candidate solution contains both dominant and recessive genes (genotypes). To determine what model parameter values the coded candidate solution actually represents, a third string is implicitly created by comparing the two strings bit-by-bit based on their gene position. For every gene position, only the dominant gene is admitted to the same gene position of the third string. The recessive gene is admitted

only when there is no dominant gene in the same position of both strings. The third string created is the phenotype of the candidate solution that can then be decoded as a single-string coded candidate solution. The process of creating the decodable third string is the so-called dominance operation. Thus in brief, dominance is the mapping of the genotypes (two parallel strings with both dominant and recessive genes) into the phenotype (a finalized single string decodable to mathematically represent a solution in search space). It is believed (Goldberg 1989) that a candidate solution of two coded strings shields recessive genes for future use and may positively affect the search efficiency. No conclusive finding can be found in the literature.

Inversion in genetics is a reordering operator which reverses the genes of a coded string about some given switch locations. Thus, inversion operation can be seen as the reverse shuffling of bits in a coded string. The inversion operator can be applied on both single-string and double-string candidate solutions. How inversion can improve GA search efficiency is beyond the scope of this thesis and will not be discussed.

4.8 Computer Procedures of Genetic Algorithms

After providing an overview of how GAs can be used in model calibration and explaining the essential components of a genetic-algorithms based optimizer and model calibrator, let us summarize the detailed computational procedures both in writing and in a flow chart with proper GA terminology.

The conventional genetic algorithms usually follow the steps below:

- Start by asking the users to input the string population size and maximum number of generations to evolve. As mentioned before, there are no simple rules to determine the optimal population size and maximum number of generations. The binary type of coding representation of genetic strings is recommended because it can be decoded and programmed conveniently.

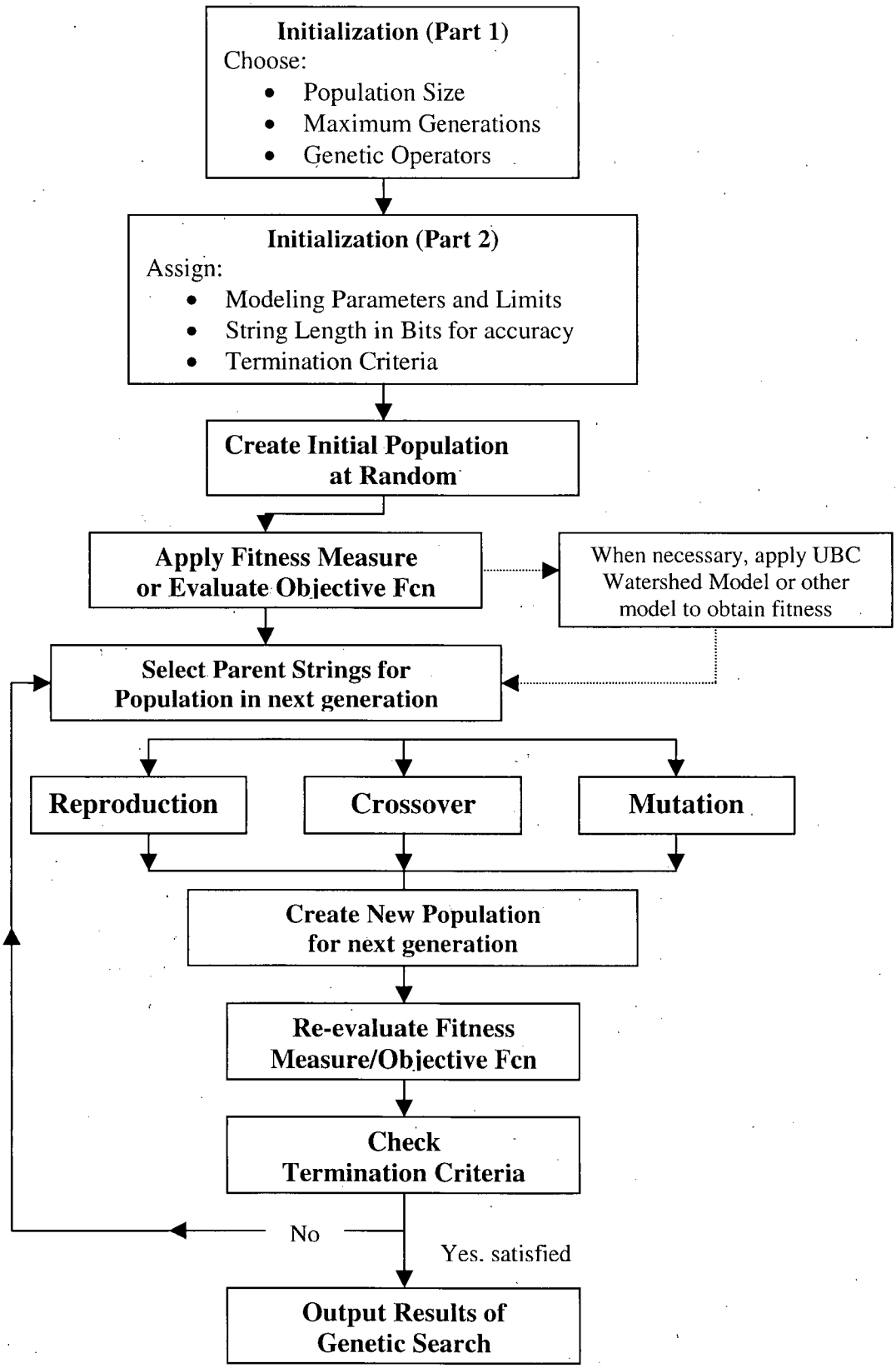
- Allow the users to choose the appropriate genetic operators. The users should have some fundamental understanding of the operators and their effects on search efficiency. Usually, for any practical application, one needs the three basic genetic operators at least; namely reproduction, crossover and mutation.
- Allow the users to decide the probability of reproduction, crossover, and mutation operators. Again there is no simple rule to decide their values though some guidelines were provided in the literature. The selection of probabilities is usually done through some experiments on GA search efficiency on a trial and error basis. For a rough start, try a crossover probability of 0.65, a reproduction probability of 0.35 and a bit-wise mutation of 0.02.
- Prescribe the number of dependent or model parameters to be estimated and assign their upper and lower bounds. Use the string length measured in bits to obtain the desirable level of accuracy for each parameter value. Use equation 4.3 in section 4.3 to determine the required string length.
- Assign GA termination criteria. Usually the users are asked to specify the maximum allowable generation number to run (criterion 1) and the degree of convergence to achieve by the best candidate solution (criterion 2). Note that only one of two criteria is needed to terminate the GA process.
- Create the initial string population at random to achieve high diversity in the gene pool. This requires the GA code to have a population do-loop that will assign genetic characteristics of all modeling parameters for every single string (candidate solution) of the population.
- For a simple optimization exercise, go to the objective function subroutine of the GA code and write the objective function of interest directly in the programming language used. The fitness of each string can then be evaluated directly by calling the subroutine repeatedly. For a model calibration exercise, choose an appropriate statistic measure as the objective function to optimize and for each string call for the computational model (for example, UBC Watershed Model) to produce the value of the fitness measure. Note that the measure chosen as the objective function should be able to statistically describe the agreement of the calibrated model results with the observed data.

- Select the parent string for mating based on the fitness of candidate solutions (strings). The weighted roulette wheel and tournament selection methods are often applied.
- Probabilistically choose one of the three major genetic operators to create offspring strings from the selected parent strings of high fitness.
- Evaluate the fitness of each newly created offspring string by calling the objective function subroutine or computational model.
- If the new offspring strings satisfy one of the two termination criteria, the GA code will stop the loop of continuous mating process and output the best candidate string and the near-best strings. These strings are then decoded to return the sets of modeling parameter values they represent. They are the best and near-best solutions.
- If none of the termination criteria is satisfied, restart the parent string selection and produce new offspring for the next generation. The loop of string mating continues until at least one termination criterion is satisfied.

The implementation of computational procedures of genetic algorithms as an optimizer or as a model calibrator can be best illustrated in a flow chart as shown in Figure 4-7.

Goldberg (1989) provides several simple examples demonstrating the manual computational process of GAs.

Figure 4-7: Implementation Flowchart of the Genetic Algorithms



5.0 FUNDAMENTAL THEOREM OF GENETIC ALGORITHMS: WHY GENETIC ALGORITHMS WORK

At first glance, one might see genetic algorithms as a technique similar to random search algorithm with an additional capability to preserve one, or a few, best solutions. Fortunately, GAs offer more than a search procedure that uses random choice as a tool to guide a search of the solution space. The high probability of survival granted to fitter solutions implicitly steers the GA search systematically towards the regions (schemata) of high objective function value in the multi-dimensional search space. As discussed in an earlier example, for a search space of 20 dimensions, random search can be as inefficient as having to explore a large portion of all possible 20^{20} combinations of parameter values as the parameter value step size is 1/20 of the feasible range of each parameter.

5.1 Schemata: Genetic Building Blocks

To analytically demonstrate why genetic algorithms work, it is imperative to discuss the concept of schemata, which are the fundamental basis of genetic algorithms. A schema, as defined by Holland (1975), is a similarity template describing a subset of strings with similarity at certain positions of strings. For example, the two 7-bit strings below:

1001000
1001101

belong to the same schema *1001*0**, where * (asterisk) means the coded gene value is not fixed; it can be either 0 or 1. Because the schema *1001*0** contains asterisks in two positions, in total the schema should contain a subset of four strings. They are:

1001000
1001101
1001001
1001100

Thus the number of strings in a binary-coded schema can be expressed as:

$$N_{ss} = 2^{x} \qquad \textbf{(Equation 5-1)}$$

where x is the number of asterisks, the non-defined gene value in a schema.

For visualization, a schema may be assumed as a plane with a number of search points (a subset of strings) on its surface in a solution space. It is therefore possible that a newly created offspring string remains in the same schema as one of its parent strings if the new offspring happens to be on the same surface plane. In other words, an offspring string does not have to be identical to any of the parent strings, yet belongs to the same schema family as one parent string (but must be partially identical in some genes). For example, a parent string *1001001* can produce an offspring string *1001100* by mutation (at fifth and seventh gene positions). But both strings still belong to the same schema family *1001*0**. On the other hand, if an offspring string *0001001* is created by mutation at the first fixed-value gene, then the new string is no longer considered a member of schema family *1001*0**.

With the basic understanding of a schema, the mystery of why genetic algorithms work well in systematically finding an optimal solution can begin to be unraveled. The essence of GAs is that if a schema contains strings of high fitness such that the schema's average string fitness is higher than other schemata, then this schema (plane) will be searched for an optimal solution in more detail than the others (to be proven mathematically in the next section). This is achieved by producing more offspring strings belonging to this specific schema of high fitness. Therefore, unlike an individual string (candidate solution) whose survival depends solely on the individual string fitness, the survival of a schema in genetic operation depends on the average fitness of all strings belonging to the schema. If a schema contains fitter strings, then the number of strings that belong to this schema will grow and the schema will survive. On the contrary, if a schema contains poor fitness strings, the number of strings that belong to this schema will decline and the schema will not survive.

Hence, all schemata are not created equal (Goldberg 1989). Comparing the schema *1001*0** with schema *100*****, how do their genetic differences affect their chance of survival as schemata? The inherent differences in schemata prompt the use of two schema measures, namely the schema order and schema defining length. These two schema measures allow us to determine the possibility for an offspring string to have the same schema as its parent string and thereafter quantify the possibility for a schema to survive generations of genetic operations. Mathematical proofs of the Fundamental Theorem of Genetic Algorithms will be provided one step at a time in the following sections with the aides of schema order and schema defining length.

5.1.1 Order of Schema

The order of a (binary coded) schema is the number of bit positions with a fixed gene value rather than an * (asterisk) which denotes it can be either 0 or 1. The schema *1001*0** therefore has an order of five whereas the schema *100***** has an order of three. The order of a schema is important because it measures how vulnerable a string and a schema are to mutation. Clearly, the higher the order of a schema, the more fixed-value genes it has, and consequently, the easier the schema can be destroyed. On the contrary, the lower the order of a schema, the fewer fixed-value genes it has, and consequently, the less likely the schema can be destroyed. Thus, the survival probability of a string's schema in mutation, which in other words is the probability of a parent string from schema H to create an offspring belonging to the same schema H, is:

$$P_{survival}(H) = \left(1 - P_m'\right)^{o(H)} \cong 1 - o(H) \cdot P_m' \qquad \textbf{(Equation 5-2)}$$

where H = a specific schema to which the parent string of interest belongs

$o(H)$ = the order of the schema H

P_m' = bit-wise mutation probability

Because the survival of a schema is measured by the number of member strings in the schema family, the equation above is also the schema's survival probability in mutation.

Conversely, the destruction probability of a string's schema, which is the probability of a parent string from schema H to create an offspring not belonging to the same schema H, can be approximated as directly proportional to the schema order provided the bit-wise mutation probability is small.

$$P_{destruction}(H) = 1 - P_{survival}(H) \cong o(H) \cdot P_m' \qquad \textbf{(Equation 5-3)}$$

For example, a parent string from schema *1001*0** (order 5) has a probability of $(1-P_m')^5$ to create a offspring in the same schema *1001*0**. This is analogous to the chance of having a 100-year return period storm in five consecutive years, which can be expressed as $\left(1-\frac{1}{100}\right)^5 \cong 1-5*\left(\frac{1}{100}\right)$.

5.1.2 Defining Length of Schema

The defining length of a (binary coded) schema is the number of bit linkages between the first and the last fixed-value genes. It is similar to the number of possible cutoff points for crossover discussed in section 4.7.2, except only the cutoff points in between the first and the last fixed-value genes are considered and counted. For instance, the schema *1001*0** has a defining length of 6-1=5 whereas the schema *100***** has a defining length of 3-1= 2 bits.

The schema defining length is important because it measures how vulnerable a schema is to crossover. Quantitatively, it determines the destruction probability of a string's inherent schema when the string undergoes a crossover operation. The longer the defining length, the greater possibility a string's schema can be destroyed in crossover and create a new offspring not belonging to the same schema. In schema *1001*0** (a

defining length of 5), if the crossover occurs at any of the first five cutoff points, the crossover operator will potentially destroy the schema. However, if the crossover occurs at the sixth cutoff point (in between the sixth and seventh bits), then crossover cannot possibly destroy the schema because the seventh bit can be an arbitrary gene value.

Thus, if only the crossover operator takes place in genetic operation, the probability of a parent string to lose its associated schema in a crossover operation is:

$$P_{destruction}(H) = \frac{d(H)}{l-1} \qquad \textbf{(Equation 5-4)}$$

where $d(H)$ = the defining length of schema, H.

Conversely, the probability of a string to maintain its schema in a crossover operation is:

$$P_{survival}(H) = 1 - P_{destruction}(H) = 1 - \frac{d(H)}{l-1} \qquad \textbf{(Equation 5-5)}$$

When other genetic operators are used in conjunction with the crossover operator in mating and the roulette wheel selection method is used, the survival probability of a schema for a given crossover probability of P_c is:

$$P_{survival}(H) \geq 1 - P_c \cdot \frac{d(H)}{l-1} \qquad \textbf{(Equation 5-6)}$$

A greater-than-or-equal sign is used because even if a crossover takes place within the defining length of a string, it is still possible that a string crosses over and mates with a string of similar or identical genetic information to form an offspring of the same schema.

Comparing the schema *1001*0** with schema *100*****, it is now clear that their probabilities of survival through crossover operation are very different. The schema

*100***** is much more likely to survive crossover because it has a short defining length of 2 whereas the schema *1001*0** contain has a long defining length of 5.

5.2 Effects of Genetic Operations on Schemata

When a GA run is conducted, the genetic operators process the selected parent strings for mating and create a population of mostly new strings. However, implicitly, the genetic operations may not necessarily lead to a population of new schemata. As discussed earlier, a schema which contains fitter strings will grow and survive by producing more numbers of strings that belong to the same schema while a schema that contains poor fitness strings will decline by producing fewer and fewer numbers of strings that belong to the schema and eventually become extinct. Hence, in order to validate the above statement it is important to observe the effects of different genetic operations on a schema and its member strings.

Goldberg (1987) provided the mathematical proofs on how reproduction, crossover and mutation (the three major genetic operators) individually and in-combination affect the survival of a schema. These mathematical proofs are presented in the following sections. The effects of mutation and crossover have already been considered separately in the previous sections during the introduction of schema order and schema defining length.

5.2.1 Effect of Reproduction

When a reproduction operation takes place, a selected parent string simply duplicates itself to create an identical offspring string and the offspring string will always belong to same schema as the parent. Thus the survival of a schema undergoing reproduction, unlikely crossover or mutation, is totally independent of its schema order and defining length. Indeed, the survival of a schema undergoing reproduction depends solely on the average fitness of the string members in the schema family.

To prove this statement, recall from Section 4.6.1, statistically the expected number *(N)* of offspring that a parent string would contribute to produce depends on the parent string's individual fitness and the average population fitness as follows:

$$N = n \cdot \frac{f_{individual}}{\sum_{i=1}^{Pop.} f_i} = \frac{f_{individual}}{f_{average}} \qquad \textbf{(Equation 4-9)}$$

If the roulette wheel selection method is again used for reproduction, then the expected number of strings belonging to schema *H*, in the immediate next generation of the population can be analogously approximated as:

$$m(H,t+1) = m(H,t) \cdot n \cdot \frac{f(H)_{average}}{\sum_{1}^{n} f} = m(H,t) \cdot \frac{f(H)_{average}}{f_{average}} = n \cdot \frac{\sum_{1}^{m(H,t)} f(H)}{\sum_{1}^{n} f} \qquad \textbf{(Equation 5-7)}$$

where n = the number of strings in a population of candidate solutions

$m(H,t) = m$ number of strings within schema H in the population of generation t

$f(H)$ = fitness of a particular string with schema H

f = fitness of a string in population

Therefore, in reproduction alone, a schema grows or decays in size based on the ratio of the schema's average fitness to the general population's average fitness. If the schema's average string fitness is greater than the general population's average string fitness, then this schema will grow, otherwise it will decay.

5.2.2 Effect of Crossover

On the other hand, if only the crossover operator takes place in genetic operation, as discussed in section 5.1.2, the schema experiences a probability of destruction

proportional to the schema defining length. The probability of a string to lose its associated schema can be expressed as:

$$P_{destruction}(H) = \frac{d(H)}{l-1} \qquad \textbf{(Equation 5-4)}$$

When other genetic operators are used in conjunction with the crossover operator in mating, for a given crossover probability of P_c, the combined probability for a string to undergo crossover and also lose its associated schema in the respective crossover operation is:

$$P_{destruction}(H) = P_c \cdot \frac{d(H)}{l-1} \qquad \textbf{(Equation 5-8)}$$

Thus, with the combined effect of reproduction and crossover operations considered, the expected number of strings belonging to schema, H, in the next generation of the population can then be calculated as:

$$m(H,t+1) \geq m(H,t) \cdot \frac{f(H)_{average}}{f_{average}} \left[1 - P_c \cdot \frac{d(H)}{l-1}\right] \qquad \textbf{(Equation 5-9)}$$

5.2.3 Effect of Mutation

Lastly, if only the mutation operator takes place in genetic operation, as discussed in section 5.1.1, the schema experiences a probability of schema destruction approximately proportional to the schema order. Thus, the probability of a string to lose its associated schema in a mutation operation is:

$$P_{destruction}(H) \cong o(H) \cdot P_m \qquad \textbf{(Equation 5-3)}$$

Therefore, after considering the combined effect of all three major genetic operators, reproduction, crossover and mutation, the expected number of schema, H, in the next generation of the population can be calculated as:

$$m(H,t+1) \geq m(H,t) \cdot \frac{f(H)_{average}}{f_{average}} \left[1 - P_c \cdot \frac{d(H)}{l-1} - o(H) \cdot P_m \right] \qquad \textbf{(Equation 5-10)}$$

The equation derived above mathematically demonstrates the possible growth or the decay of a schema, i.e. the number of member strings belonging to the schema, when only the three major genetic operators are used.

Hence the schema growth-decay factor in generation t+1 is:

$$g(H,t+1) \geq \frac{m(H,t+1)}{m(H,t)} = \frac{f(H)_{average}}{f_{average}} \left[1 - P_c \cdot \frac{d(H)}{l-1} - o(H) \cdot P_m \right] \qquad \textbf{(Equation 5-11)}$$

From Equation 5-11, one may conclude that the number string in schema, H, in the subsequent generation grows or decays depending on the ratio of schema's average fitness to the general population's average fitness, schema order and defining length. Thus, above-average fitness, short defining length and few order schemata will receive exponentially growing numbers of strings of the same schemata in future generations. This leads to exponentially increasing search opportunities in fitter schemata with potential to reach the optimal solution. This conclusion is so important to genetic algorithm that it is called the Schema Theorem or the Fundamental Theorem of Genetic Algorithms (Goldberg 1989).

Goldberg (1989) pointed out that the growth or decay of every schema is carried out in parallel because a string (candidate solution) can be a member of several schemata concurrently. This type of implicit parallelism is unique in GAs and crucial for GAs' ability in obtaining an optimal solution.

Lastly, it is important to understand that although the Schema Theorem mathematically proves how GA search is steered towards high-fitness regions (schemata) of the search space, it falls short in predicting the number of strings and generations needed to find an optimum and in assigning the probabilities of various genetic operations for most efficient GA search results. Experimental work is required to determine a reasonable set of probabilities for various genetic operations to achieve an efficient search. Although the above equations are not directly applicable, they clearly show that the GA search process is superior to random search process.

6.0 CASE STUDIES: GENETIC ALGORITHMS APPLICATION IN WATERSHED MODELING

To test the usefulness of genetic algorithms in watershed model calibration, the UBC Watershed Model is used to verify the soundness of the genetic algorithms code programmed by the author, and to demonstrate the strength and capability of genetic algorithms in facilitating the model calibration process. This was achieved by integrating the stand-alone genetic algorithms code initially written in Fortran with the UBC Watershed Model written in Visual Basic.

This chapter is organized into six sections. In section 1, background information of the two watersheds used as case studies is provided. In section 2, a list of modeling parameters incorporated in the GA-based model calibrator is provided and the physical meanings of these parameters are briefly explained. In section 3, concerns regarding how the meteorological data are used in preparing the input file for the UBC watershed will be addressed. In section 4, the results of GA calibration will be summarized and compared with the findings from the previous study of the two watersheds under scrutiny. In section 5, the search efficiency of genetic algorithms equipped with various genetic operators is further investigated. In section 6, the aforementioned three statistical measures used as objective functions and indicators of model performance are compared for their ability to accurately reflect the degree of conformance between the observed and simulated streamflow.

6.1 Short Description of Campbell River and Illecillewaet River Watersheds

Two well-studied watersheds in the Province of British Columbia were used as the case studies throughout this chapter, so that comparison could be made with earlier calibration work. They are:

1. Campbell River Watershed
2. Illecillewaet River Watershed

The two watersheds will first be used to investigate how capably the existing GA code calibrates the watersheds, and whether it achieves good model agreement based on the chosen objective model performance measures. The two case studies will also be used to determine which genetic technique or combination of techniques is most efficient for finding an optimal set of watershed modeling parameters. Lastly, the two watersheds will then be used to show which statistical measure is the most consistent measure of the agreement between the model simulated data and the observed data. The following are brief descriptions of the two watersheds.

6.1.1 Campbell River Watershed

The Campbell River Watershed is located in the middle part of Vancouver Island. It drains an area upstream of a B.C. Hydro dam which forms Upper Campbell Lake. The main part of this watershed is actually within the Stratchona Provincial Park. It is bounded by the Vancouver Island mountain ranges on the east and the Stratchona Provincial Park mountain ranges on the south and west. The watershed covers an area of 1194 km^2 and 72% is covered with forest. It has a northern orientation with elevation ranges from 215 to 2065 m (Micovic 1998). Daily meteorological information for the watershed is provided by the two local AES (Canadian Atmospheric and Environmental Services) weather stations. The stations are located at an elevation of 370 and 1490 m, respectively. In modeling the watershed, it was divided into seven elevation bands. The mid-elevations and areas of the seven bands are summarized in the table below.

Table 6-1: Brief Summary of Campbell River Watershed

Elevation Band ID	1	2	3	4	5	6	7
Mid-elevation of the band (m)	223	406	721	983	1238	1485	1939
Area (km^2)	66.7	218.4	218.4	218.4	218.4	218.4	34.8

6.1.2 Illecillewaet River Watershed

Illecillewaet River is an eastern tributary to the Columbia River at Revelstoke. The Illecillewaet River watershed is located in the Selkirk Mountains and the watershed is bounded by high glaciers and icefields, the most significant being Albert Glacier on the south, Illecillwaet Glacier on the east and Dismal and Durrand Glaciers on the northwest. This is a rugged, mountainous watershed with a drainage area of 1150 km^2, 74% of which is covered with forest. It has a southwestern orientation with elevation ranges from 520 to 3260 m (Micovic 1998). Daily meteorological information for the watershed is provided by Fidelity Mountain, Roger's Pass, and Revelstoke AES stations. These three local stations are located at the elevation of 1875, 1330 and 440 m, respectively. In modeling the watershed, it was divided into eight elevation bands. The mid-elevations and areas of the eight bands are summarized in the table below.

Table 6-2: Brief Summary of Illecillewaet River Watershed

Elevation Band ID	1	2	3	4	5	6	7	8
Mid-elevation of the band (m)	1000	1360	1540	1650	1790	1915	2085	2250
Area (km^2)	230	115	115	115	115	115	115	230

6.2 Description of UBC Watershed Model Input and Calibration Parameters

The UBC Watershed Model consists of more than 60 modeling parameters. These parameters are separately stored in groups in the *.WAT* file and they physically describe a watershed model and govern the model execution. The number of parameters for each watershed is a constant but the number of elevation bands varies, depending on the modeler's experience and preference.

Under most circumstances, many model parameters can be either physically determined (physical model parameters) or have been previously calibrated for general use in all watersheds (including some of the process model parameters). The remaining parameters are the process parameters to be calibrated for each watershed. As discussed earlier in

Chapter 2, the process parameters must be estimated and refined to achieve the best model performance possible while the physical parameters remain unchanged in the calibration process. For simplicity, the parameters accepted by the GA-based model calibrator of the UBC Watershed Model are limited to the following:

- P0GRADL: Precipitation gradient factor for the elevations below E0LMID
- P0GRADM: Precipitation gradient factor for the elevations below E0LHI
- P0GRADU: Precipitation gradient factor for the elevations above E0LHI
- E0LMID: Elevation above which the precipitation gradient PROGRAM applies. Usually set at approximately 1/2 barrier height.
- E0LHI: Elevation above which the precipitation gradient PROGRAU applies. Usually set at approximately 2/3 barrier height.
- P0AGEN: Impermeable area modification factor. It is compared with how much moisture has satisfied the soil demands and used in an exponential decay function.
- P0PERC: Ground water percolation. (Maximum capacity of sub-surface storage. Excess runoff goes to interflow).
- P0DZSH: Deep zone share (lower fraction) of groundwater
- V0FLAX: Maximum flash runoff
- V0FLAS: Flash flood threshold
- P0FRTK: Rainfall fast runoff time constant
- P0FSTK: Snow melt fast runoff time constant
- P0GLTK: Glacial melt fast runoff time constant
- P0IRTK: Rainfall interflow component runoff time constant
- P0ISTK: Snow melt interflow component runoff time constant
- P0UGTK: Upper groundwater runoff time constant
- P0DZTK: Deep zone share (lower groundwater) runoff time constant
- C0IMPA (array for each elevation band): Fraction of impermeable area for the elevation band
- P0SREP (array for each AES station): adjustment factor for snowfall data
- P0RREP (array for each AES station): adjustment factor for rainfall data

Thus, all of the above can be considered as process parameters in model calibration. For a brief explanation of these watershed parameters, the reader is referred to the appendix of the UBC Watershed Model manual (UBC Mountain Hydrology Group 1995).

In addition, in the GA-based model calibration, the process model parameters of a real watershed should be constrained by the upper and lower bounds provided by the UBC watershed users' manual. By imposing limits on parameter values, the time required for GA search will be reduced and the optimal solution will always be found in the pre-determined feasible range. The parameter limits were either derived theoretically or based on years of field experiments and experience; they may be deemed as very reliable. A summary of the limits of parameter values will be provided in section 6.4, in conjunction with the modeling results for the two case studies.

Micovic (1998) found that the precipitation gradient factors and the fraction of impermeable area are the two most important modeling parameters that decide the agreement of model results with the observed results a modeler can achieve in calibration. Due to the orographic effect, the precipitation in the mountainous area tends to increase as the elevation rises. Hence in order to increase model accuracy, precipitation gradient factors are generally used as modeling parameters in the calibration process of the UBC Watershed Model to simulate the increasing precipitation from the bottom to the top of the watershed. The second parameter, fraction of impermeable area (COIMPA), is also crucial to the model accuracy in the calibration process. Theoretically, a COIMPA value should be assigned for each elevation band of the watershed. However, because the COIMPA value tends to increase as the elevation of band area increases, the GA-based model calibrator assumed a base COIMPA value for the lowest elevation band and the COIMPA value increases linearly by a delta COIMPA value for each band. The linear approximation is a simplification made to reduce the number of process parameters to be calibrated.

6.3 Preparation of Meteorological Component of Input File

Some of the meteorological stations are remotely controlled and may register false meteorological data, which are then unknowingly used for calibration. Thus, to obtain a sense of how reliable the meteorological data are at least several combinations of input meteorological data should be arranged and tested. To designate the appropriate AES stations for temperature and precipitation in different elevation bands, I0TSTA and I0PSTA parameters should be used in the input file of UBC Watershed Model.

6.3.1 Campbell River Watershed

There are two sets of meteorological data available for the watershed, one located at Elksterc (370 m above sea level) and the second at Wolf Creek (1490 m above sea level). The input file for the watershed model should make use of both sets of meteorological data with a specific weighting on various elevation bands. Intuitively, one may designate the data collected from AES station 1 to be representative for bands 1, 2 and 3, which are located at lower altitudes (with mid-elevations ranging from 223 m to 721 m), and data from AES station 2 for bands 4, 5, 6 and 7 at high altitudes (with mid-elevations ranging from 983 m to 1939 m). However, because both of these two AES stations are remotely controlled, the reliability of the two may be doubtful. It is possible that one station may break-down, or for some other reason not record the meteorological data properly. Thus, sometimes it may be useful for the model calibration process to include only one set of the meteorological data at a time to see whether the simulation and calibration results are reasonable. Sometimes, it is very possible that a modeler can encounter difficulty in the calibration of a watershed model without realizing that the source of error lies in the meteorological data, rather than in the estimates of the modeling parameters. Table 6-3 lists the combinations of meteorological data used for model simulation and calibration.

The Campbell River Watershed is simulated from October 1983 to September 1990 in this thesis. For simplicity, only one precipitation gradient (P0GRADL) will be used in

the calibration process of combinations 1 to 5 with an exception for combination 6 in which two precipitation gradients (P0GRADL and P0GRADM) will be used.

Table 6-3: Several Arrangements of Meteorological Data for Campbell River

Elevation Band ID	Band 1	Band 2	Band 3	Band 4	Band 5	Band 6	Band 7
Combination 1	1	1	1	2	2	2	2
Combination 2	1	1	1	1	1	1	1
Combination 3	2	2	2	2	2	2	2
Combination 4	1	1	2	2	2	2	2
Combination 5	1	1	1	1	2	2	2
Combination 6*	1	1	1	2	2	2	2

*Both precipitation gradients (P0GRADL and P0GRADM) are used

6.3.2 Illecillewaet River Watershed

There are three AES stations in the Illecillewaet River Watershed: Fidelity Mountain (1875 m), Roger's Pass (1330 m) and Revelstoke (440 m). Thus, similar to Campbell River watershed, the meteorological component of the watershed input file should attempt to make a combinatory use of the three data sets available with some degree of elevation-based weighting criteria. For example, because the elevation of Roger's Pass station is 1330 m, it should be at least representative and useful for elevation bands 1 to 4 (with mid-elevations ranging from 1000 m to 1650 m). On the other hand, Fidelity Mountain, located at an elevation of 1875 m, should be at least representative for bands 5 to 8 (with mid-elevations ranging from 1790 m to 2250 m). Because the lowest elevation band (band 1) has a mean band elevation of 1000 m, it appears that data collected from Revelstoke station (440 m) may not be needed due to its low elevation. Nevertheless, it was decided that the data from Revelstoke station should be kept and used with a less emphasis. Table 6-4 lists the combination of meteorological data used for the model simulation. Despite Revelstoke station's low altitude, combinations 1 and 2 assume that data collected from Revelstoke station are representative for both bands 1 & 2 and for band 1 only, respectively. To test the representativeness of each individual station, each station is used on its own. Combination 3 uses only the Revelstoke station. Combination 4 uses only Roger's Pass, and combination 5 uses only Fidelity Mountain. Combination

6 distributes the meteorological data to all eight elevation bands based on their mid-elevations, however, only the Fidelity Mountain and Roger's Pass data are used.

The Illecillewaet River Watershed was simulated from October 1981 to September 1989 in this thesis. Similar to Campbell River watershed only one precipitation gradient (P0GRADL) will be used in the calibration process in combinations 1 to 6. Combinations 3, 4, and 5 have exactly the same meteorological data arrangement as combinations 7, 8, and 9 except that two precipitation gradients (P0GRADL and P0GRADM) will be used in the later combinations.

Table 6-4: Several Arrangements of Meteorological Data for Illecillewaet River

Elevation Band ID	Band 1	Band 2	Band 3	Band 4	Band 5	Band 6	Band 7	Band 8
Combination 1	3	3	2	2	1	1	1	1
Combination 2	3	2	2	2	1	1	1	1
Combination 3	3	3	3	3	3	3	3	3
Combination 4	2	2	2	2	2	2	2	2
Combination 5	1	1	1	1	1	1	1	1
Combination 6	2	2	2	2	1	1	1	1
Combination 7*	3	3	3	3	3	3	3	3
Combination 8*	2	2	2	2	2	2	2	2
Combination 9*	1	1	1	1	1	1	1	1

*Both precipitation gradients (P0GRADL and P0GRADM) are used

6.4 Evaluation of GA Model Calibration

The automatic GA calibrator developed for the UBC Watershed Model was used to calibrate the Campbell River and Illecillewaet River watersheds, which were both studied by Micovic (1998). In this section, the GA calibration results, in the form of model parameter values and overall model performance measures, are compared with Micovic's calibration results.

6.4.1 Campbell River Watershed

Based on the meteorological data arrangements presented in Table 6-3 and the assumption that one to two precipitation gradient factors are sufficient in depicting the orographic effect of precipitation, the GA calibrator is used to generate the following calibration results for the Campbell River watershed in Table 6-5. For comparison, the upper and lower bounds of the modeling parameters are also summarized in Table 6-5. These values are provided in the UBC Watershed Model manual. For every meteorological data combination, only two top candidate solutions are presented, although the population size was set at 20 in the GA search. In combinations 1 to 5, the value of E0LMID and E0LHI are purposely raised to an elevation (of about 2400 m) above the entire watershed so that parameters P0GRADM and P0GRADU cannot affect the precipitation within the watershed. As mentioned earlier, the precipitation gradient factors are generally used in the UBC Watershed Model to simulate the increasing precipitation from the bottom to the top of the watershed as a result of the orographic effect.

After examining the resulting statistical measures of the model performance: *e!, dV/V* and *eopt!* in Table 6-5, one can see that the meteorological data arrangements, which follow the principle of assigning data to elevation bands based on their elevation proximity as in combinations 1 and 6, yields the best calibrated model performance in both *e!* and *eopt!*. The values achieved by the automatic GA calibration are 0.720 and 0.716, respectively, for the best candidate solution in combination 1 and similar values of 0.718 and 0.709 for combination 6. If only the meteorological data from AES station 1 is used as in combination 2, the values of *e!* and *eopt!* decrease significantly to 0.602 and 0.602 (the same), respectively. Similarly, if only the meteorological data from AES station 2 is used as in combination 3, the values of *e!* and *eopt*! also decrease significantly to 0.633 and 0.633 (the same), respectively. However, it is worth mentioning that despite the low *e!* and *eopt!* values achieved, the *dV/V* values are zero in both combinations 2 and 3. In combinations 4 and 5 which have meteorological data arrangements similar to combination 1 based on the vertical proximity between AES stations and elevation bands,

the calibration results are still satisfactory though not as high as the model performance measures obtained from combinations 1 and 6.

Table 6-5: GA Calibrated Model Parameter Values for Various Combinations of Campbell River Watershed

Parameter	Lower Bound	Upper Bound	Comb. 1 A	Comb. 1 B	Comb. 2 A	Comb. 2 B	Comb. 3 A	Comb. 3 B	Comb. 4 A	Comb. 4 B	Comb. 5 A	Comb. 5 B	Comb. 6 A	Comb. 6 B
P0GRADL	0	20	5.02	15.06	17.33	17.33	15.06	15.06	18.82	18.90	7.53	5.02	5.02	5.02
P0GRADM	0	20	-	-	-	-	-	-	-	-	-	-	8.31	8.31
P0GRADU	0	20	-	-	-	-	-	-	-	-	-	-	7.14	7.14
E0LMID	2400*	2401*	2401	2401	2401	2401	2400	2400	2401	2400	2400	2401	1991	1991
E0LHI	2401	2402	2402	2402	2402	2402	2402	2402	2402	2402	2401	2402	2401	2401
P0AGEN	80	120	109.49	109.49	100.86	100.86	109.49	109.80	108.39	88.31	99.14	119.53	99.45	99.45
P0PERC	10	50	21.29	21.29	43.73	41.06	31.65	31.65	43.73	43.57	21.29	21.61	21.45	21.45
P0DZSH	0	1	0.40	0.40	0.56	0.60	0.44	0.44	0.18	0.06	0.77	0.77	0.91	0.91
V0FLAX	1700	1900	1724	1724	1850	1837	1818	1818	1881	1812	1846	1844	1724	1724
V0FLAS	20	60	25.02	27.53	40.39	40.39	27.69	27.69	28.47	28.47	45.88	48.55	25.65	25.65
P0FRTK	0	2	0.64	0.51	0.49	0.49	0.62	0.62	0.57	0.51	0.51	0.56	0.51	0.51
P0FSTK	0	2	1.62	0.87	0.05	0.05	0.87	0.87	1.04	0.53	0.49	0.31	1.62	1.62
P0GLTK	0	2	1.85	0.82	0.35	0.35	0.78	0.79	0.86	0.86	1.41	1.41	1.85	1.85
P0IRTK	3	10	7.23	3.71	5.00	5.00	9.59	9.59	8.16	6.08	6.38	4.84	3.99	3.99
P0ISTK	3	10	8.35	9.23	6.65	6.65	8.46	5.17	7.04	3.19	4.98	8.27	8.44	8.44
P0UGTK	10	50	46.55	49.69	36.20	35.57	38.86	38.86	25.22	20.20	27.41	12.35	36.98	36.98
P0DZTK	100	300	121.96	121.96	296.08	270.98	220.78	170.59	170.59	270.98	290.59	240.39	120.39	120.39
C0IMPA1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
C0IMPA2	0	0.3	0.22	0.21	0.22	0.22	0.19	0.19	0.19	0.22	0.20	0.17	0.21	0.21
Δ C0IMPA	0	0.14	0.11	0.11	0.08	0.08	0.07	0.07	0.06	0.06	0.12	0.12	0.11	0.11
P0SREP1	-1	1	0.37	0.37	-0.85	-0.85	0.00	0.00	-0.49	-0.93	-0.65	-0.52	-0.54	-0.66
P0SREP2	-1	1	-0.47	-0.48	0.00	0.00	-0.33	-0.33	-0.43	-0.43	-0.33	-0.33	-0.36	-0.36
P0RREP1	-1	1	-0.11	-0.11	0.19	0.19	-	-	0.71	0.72	-0.12	-0.12	-0.08	-0.08
P0RREP2	-1	1	0.14	0.17	-	-	0.22	0.22	0.09	0.09	0.44	0.43	0.17	0.17
e!	-infinity	1	**0.720**	**0.716**	**0.602**	**0.603**	**0.633**	**0.633**	**0.692**	**0.689**	**0.709**	**0.701**	**0.720**	**0.720**
dV/V	**0**	1	**0.004**	**0.001**	**0.000**	**0.001**	**0.000**	**0.003**	**0.001**	**0.002**	**0.005**	**0.005**	**0.003**	**0.011**
eopt!	**0**	1	**0.716**	**0.715**	**0.602**	**0.602**	**0.633**	**0.630**	**0.690**	**0.687**	**0.704**	**0.696**	**0.718**	**0.709**
Rank	**1**	**20**	**1**	**2**	**1**	**2**	**1**	**2**	**1**	**2**	**1**	**2**	**1**	**2**

*For combination 6 in which two precipitation gradient factors are used, the lower and upper bound of E0LMID parameter are set at 1200 m and 2000 m.

Micovic (1998) calibrated the Campbell River watershed through a trial-and-error procedure, which included a certain degree of manual calibration and automatic random model calibration. As noted previously, under the existing model framework, direct

search of the values of all modeling parameters is not possible, and the number of modeling parameter values calibrated simultaneously is often limited to groups of about three to six at a time. The resulting parameter values from Micovic's (1998) best calibrated Campbell River (in terms of *eopt!*) is given in Table 6-6 together with the parameter values from the best GA-based model calibration.

Table 6-6: Resulting Parameter Values from Micovic's Best Model Calibration of Campbell River Watershed

Parameter	Lower Bound	Upper Bound	Micovic's calibrated value	Comb. 1 A	Comb. 2 A	Comb. 3 A	Comb. 4 A	Comb. 5 A	Comb. 6 A
P0GRADL	0	20	**2**	5.02	17.33	15.06	18.82	7.53	5.02
P0GRADM	0	20	**2**	-	-	-	-	-	8.31
P0GRADU	0	20	**0**	-	-	-	-	-	7.14
E0LMID	2400*	2401*	**963**	2401	2401	2400	2401	2400	1991
E0LHI	2401	2402	**2000**	2402	2402	2402	2402	2401	2401
P0AGEN	80	120	**100**	109.49	100.86	109.49	108.39	99.14	99.45
P0PERC	10	50	**18**	21.29	43.73	31.65	43.73	21.29	21.45
P0DZSH	0	1	**0.46**	0.40	0.56	0.44	0.18	0.77	0.91
V0FLAX	1700	1900	**1800**	1724	1850	1818	1881	1846	1724
V0FLAS	20	60	**33**	25.02	40.39	27.69	28.47	45.88	25.65
P0FRTK	0	2	**0.38**	0.64	0.49	0.62	0.57	0.51	0.51
P0FSTK	0	2	**0.4**	1.62	0.05	0.87	1.04	0.49	1.62
P0GLTK	0	2	**1.0**	1.85	0.35	0.78	0.86	1.41	1.85
P0IRTK	3	10	**2.0**	7.23	5.00	9.59	8.16	6.38	3.99
P0ISTK	3	10	**2.0**	8.35	6.65	8.46	7.04	4.98	8.44
P0UGTK	10	50	**22**	46.55	36.20	38.86	25.22	27.41	36.98
P0DZTK	100	300	**72**	121.96	296.08	220.78	170.59	290.59	120.39
C0IMPA1	1	1	**1**	1	1	1	1	1	1
C0IMPA2	0	0.3	**0.10**	0.22	0.22	0.19	0.19	0.20	0.21
Δ C0IMPA	0	0.14	**varies****	0.11	0.08	0.07	0.06	0.12	0.11
P0SREP1	-1	1	**0.25**	0.37	-0.85	0.00	-0.49	-0.65	-0.54
P0SREP2	-1	1	**0.27**	-0.47	0.00	-0.33	-0.43	-0.33	-0.36
P0RREP1	-1	1	**0.07**	-0.11	0.19	-	0.71	-0.12	-0.08
P0RREP2	-1	1	**0.08**	0.14	-	0.22	0.09	0.44	0.17
e!	-infinity	**1**	**0.723**	**0.720**	**0.602**	**0.633**	**0.692**	**0.709**	**0.720**
dV/V	**0**	1	**0.002**	**0.004**	**0.000**	**0.000**	**0.001**	**0.005**	**0.003**
eopt!	**0**	1	**0.720**	**0.716**	**0.602**	**0.633**	**0.690**	**0.704**	**0.718**
Rank	**1**	**20**	**-**	**1**	**1**	**1**	**1**	**1**	**1**

*For combination 6 in which two precipitation gradient factors are used, the lower and upper bound of E0LMID parameter are set at 1200 m and 2000 m.

**The values of C0IMPA for bands 1 to 7 are 1.00, 0.10, 0.10, 0.10, 0.30, 0.50 and 0.75.

To determine whether the GA-based calibrated Campbell River watershed model can achieve as high model performance as Micovic's in terms of statistical measures, *e!* and *eopt!*, the model statistic module of the UBC Watershed Model is run on the GA calibration results. The statistical summary of GA-calibrated combination 1 results by water year is given in Table 6-7. The summaries of other GA-calibrated combinations of Campbell River Watershed with high *e!* and *eopt!* values are given in the appendix. Statistical summaries for combinations with low *e!* and *eopt!* values are not provided.

Table 6-8 is a summary of the model performance statistics from the best calibrated Campbell River Watershed as provided by Micovic (1998). Unfortunately, because Micovic used a slightly different meteorological input file in his calibration work, the resulting overall *e!* and *eopt!* values for the whole simulation period cannot be directly compared. However, in an effort to provide some equal-comparison basis, GA calibration results were re-run in the UBC Watershed Model to provide the *e!* values for each water year. The overall *e!* and *eopt!* value are found to be 0.722 and 0.720, respectively from Micovic's results and 0.720 and 0.716, respectively from GA calibration results. It should be noted that Micovic's calibration uses interpolation of the meteorological data, using the two AES stations, Elksterc and Wolf Creek, as opposed to the gradient algorithm used in the present work.

Table 6-7: Statistics of Model Performance for Combination 1 of Campbell River Watershed after GA Calibration

STATISTICS FOR THE OCT 1, 1983 - SEP 30, 1990 WATER YEARS

	Mean Q_{obs} *(cms/d)*	Mean Q_{est} *(cms/d)*	Tot Q_{obs} *(cms)*	Tot Q_{est} *(cms)*	Tot Q_{obs} -Tot Q_{est}	Coeff.of Eff	Coeff.of Det
YEAR	(831001-840930)						
YEAR	75.49	76.78	27631.09	28100.67	-469.58	0.68	0.69
YEAR	(841001-850930)						
YEAR	59.65	58.47	21773.00	21340.08	432.92	0.71	0.77
YEAR	(851001-860930)						
YEAR	75.47	68.35	27546.70	24946.18	2600.52	0.8	0.84
YEAR	(861001-870930)						
YEAR	89.59	84.38	32701.29	30799.69	1901.59	0.65	0.66
YEAR	(871001-880930)						
YEAR	70.23	68.35	25705.90	25015.31	690.60	0.72	0.73
YEAR	(881001-890930)						
YEAR	63.57	65.86	23203.41	24039.11	-835.70	0.74	0.76
YEAR	(891001-900930)						
YEAR	65.46	75.5	23893.69	27556.45	-3662.75	0.74	0.77
WHOLE	PERIOD	(831001-900930)					
PERIOD	71.36	71.1	182455.20	181797.40	657.81	0.72	0.72

Table 6-8: Statistics of Model Performance from Best Calibration of Campbell River Watershed (Micovic 1998)

STATISTICS FOR THE OCT 1, 1983 - SEP 30, 1990 WATER YEARS

	Mean Q_{obs} *cms/d*	Mean Q_{est} *cms/d*	Tot Q_{obs} *cms*	Tot Q_{est} *cms*	Tot Q_{obs} -Tot Q_{est}	Coeff.of Eff	Coeff.of Det
YEAR	(831001-840930)						
YEAR	75.5	79.2	27631.09	28998.6	-1367.6	0.6745	0.6812
YEAR	(841001-850930)						
YEAR	59.7	55.6	21773.00	20284.1	1488.9	0.7220	0.8002
YEAR	(851001-860930)						
YEAR	75.5	70.6	27546.70	25779.7	1767.0	0.8216	0.8262
YEAR	(861001-870930)						
YEAR	89.6	92.5	32701.29	33765.3	-1064.0	0.7262	0.7307
YEAR	(871001-880930)						
YEAR	70.2	69.0	25705.90	25249.3	456.6	0.5603	0.6117
YEAR	(881001-890930)						
YEAR	63.6	64.5	23203.41	23551.8	-348.7	0.6600	0.6602
YEAR	(891001-900930)						
YEAR	65.5	66.9	23893.69	24418.1	-524.4	0.7526	0.7855
WHOLE	PERIOD	(831001-900930)					
PERIOD	71.4	71.2	182455.20	182046.9	408.2	0.7226	0.7331

Including the Campbell River watershed, Micovic (1998) studied twelve B.C. watersheds with different drainage area, climate, topology, soil type, vegetation, geology and hydrologic regimes using the UBC Watershed Model. One of his findings is that despite the physical difference in watershed characteristics, the watersheds studied revealed there was a relatively consistent set of modeling parameter values for all watersheds except the precipitation gradients (P0GRADL, P0GRADM and P0GRADU) and fraction of impermeable area (C0IMPA). Micovic concluded that parameters which affect the time distribution of runoff along with groundwater percolation and deep zone share showed relatively low variability in the twelve studied watersheds and may be assumed as constant values. Table 6-9 summarizes the constant values of the modeling parameters with low variability suggested by Micovic. For comparison, the calibrated values of these parameters from GA search are presented together. From the table, one may see that the calibrated modeling parameters values from the GA are usually close to the suggested constant parameter values with the exception of P0ISTK (Snow melt interflow component runoff time constant) and P0UGTK (Upper groundwater runoff time constant).

Table 6-9: Suggested Constant Values for Modeling Parameters with Low Variability vs. GA Calibrated Parameter Values of Campbell River Watershed

Parameter	Description of Modeling Parameter	Lower Bound	Upper Bound	Constant Value	Unit	Comb. 1 A	Comb. 1 B	Comb. 6 A	Comb. 6 B
POPERC	Ground water percolation. (Maximum capacity of sub-surface storage. Excess runoff goes to interflow.)	10	50	**25**	mm	21.29	21.29	21.45	21.45
PODZSH	Deep zone share (lower fraction) of groundwater	0	1	**0.3**	unitless	0.40	0.40	0.91	0.91
P0FRTK	Rainfall fast runoff time constant	0	2	**0.6**	day	0.64	0.51	0.51	0.51
P0FSTK	Snow melt fast runoff time constant	0	2	**1**	day	1.62	0.87	1.62	1.62
P0IRTK	Rainfall interflow component runoff time constant	3	10	**3**	day	7.23	3.71	3.99	3.99
P0ISTK	Snow melt interflow component runoff time constant	3	10	**4**	day	8.35	9.23	8.44	8.44
P0UGTK	Upper groundwater runoff time constant	10	50	**20**	day	46.55	49.69	36.98	36.98
P0DZTK	Deep zone share (lower groundwater) runoff time constant	100	300	**150**	day	121.96	121.96	120.39	120.39

6.4.2 Illecillewaet River Watershed

In the GA calibration process of the Illecillewaet River Watershed, it is again assumed that one to two precipitation gradient factors are sufficient to depict the orographic effect of precipitation. The simulations are based on the meteorological data arrangements presented in Table 6-4. The GA model calibrator yields the results in Table 6-10 and Table 6-11. Table 6-11 is just a continuation of Table 6-10. For every meteorological data combination, only the top two candidate solutions are presented, although the population is still set at 20. In combinations 1 to 6, the value of E0LMID and E0LHI are again purposely raised to about 2400 m, which is above the entire watershed, so that parameters P0GRADM and P0GRADU cannot affect the precipitation within the watershed in any way. In combinations 7, 8, and 9, because two precipitation gradients are to be used, the value of E0LMID is allowed to range freely from 1600 m to 2400 m while E0LHI is still fixed to about 2400 m. This means that gradient factor P0GRADL is effective to adjust the precipitation data assigned to elevation bands between the lowest band and the calibrated value of E0LMID. The gradient factor P0GRADM is effective above E0LMID while P0GRADU remains ineffective.

In combination 1 where Revelstoke AES station (station 3) is used to represent the meteorological conditions in both elevation bands 1 and 2 (mid-elevations of 1000 m and 1360 m), despite the station's low elevation of 440m, the calibrated model performance turned out to be surprisingly high, with an *e!* of 0.924 and an *eopt!* of 0.914. In combination 2 where Revelstoke AES station (station 3) is used to represent the meteorological condition only in elevation band 1, the calibrated model performance turned out to be lower than the results in combination 1, with an *e!* of 0.909 and an *eopt!* of 0.908. The calibration result from combination 2 is not originally anticipated because combination 2 is closer than combination 1 in following the principle of assigning meteorological data to elevation bands based on their elevation proximity. It begins to appear that the meteorological data from Roger's Pass (station 2) may not be representative of the areas with similar altitudes. This presumption is verified in combination 4, which using data from Roger's Pass alone, yields extremely poor model

performance of 0.688 for *e!* and 0.673 for *eopt!*, respectively. To reconfirm the finding, combination 4 is re-calibrated (now called combination 8) with the aide of two precipitation gradients, but once again poor model performance is generated. It is concluded that meteorological data from Roger's Pass are problematic.

Table 6-10: GA Calibrated Model Parameter Values for Various Combinations of Illecillewaet River Watershed

Parameter	Lower Bound	Upper Bound	Comb. 1 A	Comb. 1 B	Comb. 2 A	Comb. 2 B	Comb. 3 A	Comb. 3 B	Comb. 4 A	Comb. 4 B	Comb. 5 A	Comb. 5 B	Comb. 6 A	Comb. 6 B
P0GRADL	0	20	8.63	8.63	8.71	8.71	5.10	8.08	1.02	1.02	8.63	6.12	11.77	11.77
P0GRADM	0	20	-	-	-	-	-	-	-	-	-	-	-	-
P0GRADU	0	20	-	-	-	-	-	-	-	-	-	-	-	-
E0LMID	2400	2401	2401	2401	2401	2401	2401	2401	2400	2401	2401	2401	2401	2401
E0LHI	2401	2402	2401	2401	2401	2401	2401	2401	2402	2402	2401	2401	2401	2401
P0AGEN	80	120	100.86	100.86	110.90	110.90	102.59	113.88	104.47	104.47	117.18	117.18	95.22	95.06
P0PERC	10	50	38.39	38.39	18.94	18.00	40.75	40.75	33.37	33.37	37.77	32.59	11.88	21.29
P0DZSH	0	1	0.70	0.95	0.76	0.64	0.78	0.78	0.25	0.18	0.64	0.64	0.58	0.64
V0FLAX	1700	1900	1779	1779	1778	1767	1850	1848	1817	1710	1780	1727	1764	1865
V0FLAS	20	60	51.84	51.84	60.00	52.47	23.61	23.61	47.77	47.77	34.59	59.69	57.02	50.75
P0FRTK	0	2	1.14	1.14	1.39	1.39	1.67	1.67	1.65	1.52	1.46	1.91	0.59	0.09
P0FSTK	0	2	0.78	0.78	0.54	0.79	1.48	1.41	1.37	1.37	0.79	0.79	0.99	0.93
P0GLTK	0	2	1.33	1.33	0.21	0.21	1.40	0.97	1.47	0.46	0.76	0.35	.	0.01
P0IRTK	3	10	6.84	6.84	7.50	3.99	4.10	3.33	3.91	3.91	6.90	7.72	6.57	6.57
P0ISTK	3	10	4.10	4.10	7.83	8.05	6.76	6.76	5.00	5.00	7.28	9.04	5.11	3.33
P0UGTK	10	50	20.67	20.67	17.37	11.10	43.26	30.71	13.77	33.84	21.14	23.65	36.98	39.80
P0DZTK	100	300	109.41	109.41	184.71	184.71	269.41	272.55	224.71	224.71	134.51	135.29	294.51	295.29
C0IMPA	0	0.3	0.27	0.27	0.05	0.05	0.04	0.03	0.06	0.07	0.12	0.27	0.12	0.08
Δ C0IMPA	0	0.1	0.09	0.09	0.10	0.07	0.09	0.10	0.08	0.08	0.09	0.07	0.06	0.09
P0SREP1	-1	1	-0.26	-0.26	-0.28	-0.28	-	-	-	-	-0.27	-0.26	-0.33	-0.16
P0SREP2	-1	1	0.21	0.21	-0.80	0.33	-	-	0.15	0.15	-	-	0.01	0.14
P0SREP3	-1	1	-0.10	-0.10	0.97	0.97	-0.15	-0.47	-	-	-	-	-	-
P0RREP1	-1	1	-0.98	-0.98	0.30	0.06	-	-	-	-	0.05	0.02	-0.57	-0.76
P0RREP2	-1	1	-0.08	-0.08	-0.08	-0.58	-	-	0.22	0.22	-	-	0.83	0.33
P0RREP3	-1	1	0.91	0.91	0.53	-0.98	0.85	0.85	-	-	-	-	-	-
e!	- infinity	**1**	**0.924**	**0.926**	**0.909**	**0.885**	**0.861**	**0.855**	**0.688**	**0.681**	**0.919**	**0.918**	**0.891**	**0.854**
dV/V	**0**	**1**	**0.011**	**0.013**	**0.001**	**0.002**	**0.004**	**0.002**	**0.015**	**0.012**	**0.002**	**0.027**	**0.010**	**0.001**
eopt!	**0**	**1**	**0.914**	**0.912**	**0.908**	**0.883**	**0.857**	**0.853**	**0.673**	**0.669**	**0.917**	**0.891**	**0.881**	**0.854**
Rank	**1**	**20**	**1**	**2**	**1**	**2**	**1**	**2**	**1**	**2**	**1**	**2**	**1**	**2**

Table 6-11 GA Calibrated Model Parameter Values for Various Combinations of Illecillewaet River Watershed (Continuation of Table 6-10)

Parameter	Lower Bound	Upper Bound	Comb. 7 A	Comb. 7 B	Comb. 8 A	Comb. 8 B	Comb. 9 A	Comb. 9 B
POGRADL	0	20	6.67	6.67	0.86	0.86	8.63	8.63
POGRADM	0	20	13.80	13.80	12.39	12.39	2.98	13.02
POGRADU	0	20	-	-	-	-	-	-
EOLMID	1600	2400	2011	2011	2372	1970	2344	1942
EOLHI	2401	2402	2401	2401	2402	2401	2401	2401
POAGEN	80	120	100.08	102.59	104.47	104.47	102.12	112.16
POPERC	10	50	31.96	31.96	43.73	48.43	47.80	47.80
PODZSH	0	1	0.59	0.53	0.31	0.50	0.70	0.70
VOFLAX	1700	1900	1728	1829	1847	1748	1777	1777
VOFLAS	20	60	23.45	23.45	23.92	47.77	54.04	33.96
POFRTK	0	2	1.01	1.51	1.62	1.57	1.96	1.96
POFSTK	0	2	1.73	1.23	0.87	0.80	0.78	0.78
POGLTK	0	2	0.42	1.43	0.34	0.46	0.01	0.01
POIRTK	3	10	4.26	4.21	4.57	4.57	7.45	7.39
POISTK	3	10	3.25	3.25	5.06	5.06	7.17	7.17
POUGTK	10	50	33.22	33.22	34.47	34.47	21.14	21.14
PODZTK	100	300	219.22	169.02	207.45	257.65	109.41	110.20
COIMPA	0	0.3	0.05	0.05	0.06	0.06	0.14	0.12
Δ COIMPA	0	0.1	0.08	0.09	0.09	0.08	0.09	0.09
POSREP1	-1	1	0.00	0.00	-	-	-0.28	-0.28
POSREP2	-1	1	-	-	0.24	0.21	-	-
POSREP3	-1	1	-0.34	-0.34	-	-	0.00	0.00
PORREP1	-1	1	-	-	-	-	0.05	0.04
PORREP2	-1	1	-	-	0.11	0.10	-	-
PORREP3	-1	1	0.79	0.79	0.00	0.00	0.00	0.00
e!	- infinity	**1**	**0.870**	**0.855**	**0.698**	**0.689**	**0.917**	**0.915**
dV/V	**0**	**1**	**0.002**	**0.017**	**0.005**	**0.009**	**0.002**	**0.006**
eopt!	**0**	**1**	**0.869**	**0.837**	**0.693**	**0.680**	**0.914**	**0.910**
Rank	**1**	**20**	**1**	**2**	**1**	**2**	**1**	**2**

In combination 3, Revelstoke AES station (station 3) alone is used to represent the meteorological condition of the entire watershed. Although this arrangement is contrary to the principle of assigning meteorological data to elevation bands based on their elevation proximity, the satisfactory model performance of 0.861 for *e!* and 0.857 for *eopt!* are only slightly lower than results of combinations 1 and 2, yet much better than the results of combination 4.

In combination 5, Fidelity Mountain AES station (station 1) alone is used to represent the meteorological condition of the entire watershed. Although bands 1, 2, and 3 are

considered quite far below the station, the model performance achieved is as high as 0.919 for *e!* and 0.917 for *eopt!* after calibration. This implies that a good set of meteorological data, with reasonable fine-tuning of the parameters such as precipitation gradient factors or AES adjustment factors for precipitation (P0RREP and P0SREP), can still yield high model accuracy without strictly following the principle of assigning meteorological data to elevation bands based on their elevation proximity. This finding is useful because in remote watersheds where no meteorological station is available for modeling purposes, the modeler can "borrow" meteorological data from an adjacent watershed (relative speaking) with a meteorological station.

Of all 9 meteorological data combinations listed in Table 6-4, combination 6 is the closest in following the principle of assigning meteorological data to elevation bands based on their elevation proximity. However the resulting model performance of 0.891 for *e!* and 0.881 for *eopt!* is not the highest among the nine combinations arranged. Combination 6's low model performance is attributed to the suspected problems with the meteorological data from Roger's Pass station.

Combinations 7 and 8 are almost identical to combinations 3 and 4, except that two precipitation gradients are used in calibration rather than one, and this results in slightly higher model performance. The better results are anticipated because one additional precipitation gradient means one additional modeling parameter to calibrate and more flexibility in a larger search space. However, the above statement is only true if the extra parameter gives a better description of the physical distribution of precipitation.

Similar to Campbell River model, Micovic (1998) also calibrated Illecillewaet River through a trial-and-error procedure. The resulting parameter values from the best calibrated Illecillewaet River (in terms of *eopt!*) are given in Table 6-12 together with the parameter values from the GA-based model calibration.

A statistic summary of the GA calibration results for combination 1A by water year is given in Table 6-13. The corresponding model performance measured in *e!* and *eopt!* for the whole simulation period are 0.924 and 0.914.

Table 6-14 is a statistic summary of the model performance for the best calibrated Illecillewaet River watershed provided by Micovic (1998). The *e!* and *eopt!* values obtained for the whole simulation period are 0.909 and 0.875. The statistical summary of GA results for some other combinations of Illecillewaet River Watershed with high *e!* and *eopt!* values are given in the appendix. The statistical summaries of GA results for combinations with low *e!* and *eopt!* values are not provided. It should be noted that Micovic's calibration uses interpolation of the meteorological data, using just two of the three stations, Revelstoke and Fidelity Mountain.

To reconfirm Micovic's finding of low variability in eight modeling parameters of time distribution constants of runoff, groundwater percolation and deep zone share, the GA calibrated parameter values are compared with the suggested constant parameter values. Table 6-15 summarizes the calibrated values of modeling parameters against the suggested constant values from Micovic. Because Micovic's best calibrated Illecillewaet River model achieves high *e!* value each water year, for fairness, only GA calibrated results from combinations with *e!* values > 0.90 are considered (combinations 1A, 2A, 5A, and 9A are the only ones > 0.90).

Table 6-12: Resulting Parameter Values from Micovic's Best Model Calibration of Illecillewaet River Watershed (Micovic 1998)

Parameter	Lower Bound	Upper Bound	Micovic's calibrated value	Comb. 1 A	Comb. 2 A	Comb. 3 A	Comb. 4 A	Comb. 5 A	Comb. 6 A	Comb. 7 A	Comb. 8 A	Comb. 9 A
P0GRADL	0	20	**4**	8.63	8.71	5.10	1.02	8.63	11.77	6.67	0.86	8.63
P0GRADM	0	20	**2**	-	-	-	-	-	-	13.80	12.39	2.98
P0GRADU	0	20	**0**	-	-	-	-	-	-	-	-	-
E0LMID	varies	varies	**1009**	2401	2401	2401	2400	2401	2401	2011	2372	2344
E0LHI	2401	2402	**2481**	2401	2401	2401	2402	2401	2401	2401	2402	2401
P0AGEN	80	120	**100**	100.86	110.90	102.59	104.47	117.18	95.22	100.08	104.47	102.12
P0PERC	10	50	**31**	38.39	18.94	40.75	33.37	37.77	11.88	31.96	43.73	47.80
P0DZSH	0	1	**0.25**	0.70	0.76	0.78	0.25	0.64	0.58	0.59	0.31	0.70
V0FLAX	1700	1900	**1800**	1779	1778	1850	1817	1780	1764	1728	1847	1777
V0FLAS	20	60	**36**	51.84	60.00	23.61	47.77	34.59	57.02	23.45	23.92	54.04
P0FRTK	0	2	**0.78**	1.14	1.39	1.67	1.65	1.46	0.59	1.01	1.62	1.96
P0FSTK	0	2	**1.0**	0.78	0.54	1.48	1.37	0.79	0.99	1.73	0.87	0.78
P0GLTK	0	2	**1.7**	1.33	0.21	1.40	1.47	0.76	.	0.42	0.34	0.01
P0IRTK	3	10	**2**	6.84	7.50	4.10	3.91	6.90	6.57	4.26	4.57	7.45
P0ISTK	3	10	**3**	4.10	7.83	6.76	5.00	7.28	5.11	3.25	5.06	7.17
P0UGTK	10	50	**17**	20.67	17.37	43.26	13.77	21.14	36.98	33.22	34.47	21.14
P0DZTK	100	300	**168**	109.41	184.71	269.41	224.71	134.51	294.51	219.22	207.45	109.41
C0IMPA	0	0.3	**0.1**	0.27	0.05	0.04	0.06	0.12	0.12	0.05	0.06	0.14
Δ C0IMPA	0	0.1	**varies***	0.09	0.10	0.09	0.08	0.09	0.06	0.08	0.09	0.09
P0SREP1	-1	1	**-0.22**	-0.26	-0.28	-	-	-0.27	-0.33	0.00	-	-0.28
P0SREP2	-1	1	**-0.10**	0.21	-0.80	-	0.15	-	0.01	-	0.24	-
P0SREP3	-1	1	**-0.11**	-0.10	0.97	-0.15	-	-	-	-0.34	-	0.00
P0RREP1	-1	1	**-0.14**	-0.98	0.30	-	-	0.05	-0.57	-	-	0.05
P0RREP2	-1	1	**-0.17**	-0.08	-0.08	-	0.22	-	0.83	-	0.11	-
P0RREP3	-1	1	**-0.11**	0.91	0.53	0.85	-	-	-	0.79	0.00	0.00
e!	- infinity	1	**0.909**	**0.924**	**0.909**	**0.861**	**0.688**	**0.919**	**0.891**	**0.870**	**0.698**	**0.917**
dV/V	**0**	1	**0.034**	**0.011**	**0.001**	**0.004**	**0.015**	**0.002**	**0.010**	**0.002**	**0.005**	**0.002**
eopt!	**0**	1	**0.875**	**0.914**	**0.908**	**0.857**	**0.673**	**0.917**	**0.881**	**0.869**	**0.693**	**0.914**
Rank	**1**	**20**	-	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

*The values of C0IMPA for bands 1 to 8 are 0.10, 0.10, 0.10, 0.10, 0.10, 0.40, 0.45, and 0.50.

Table 6-13: Statistics of Model Performance for Combination 1 of Illecillewaet River Watershed after GA Calibration

STATISTICS FOR THE OCT 1, 1981 - SEP 30, 1989 WATER YEARS

	Mean Q_{obs} *(cms/d)*	Mean Q_{est} *(cms/d)*	Tot Q_{obs} *(cms/d)*	Tot Q_{est} *(cms/d)*	Tot Q_{obs} -Tot Q_{est}	Coeff.of Eff	Coeff.of Det
YEAR	(811001-820930)						
YEAR	56.95	56.11	20787.69	20479.49	308.20	0.94	0.95
YEAR	(821001-830930)						
YEAR	52.28	48.89	19083.01	17846.49	1236.52	0.84	0.85
YEAR	(831001-840930)						
YEAR	52.94	48.89	19376.00	17895.16	1480.85	0.93	0.93
YEAR	(841001-850930)						
YEAR	49.44	50.56	18045.53	18453.25	-407.72	0.94	0.95
YEAR	(851001-860930)						
YEAR	54.26	53.09	19804.49	19379.42	425.07	0.94	0.94
YEAR	(861001-870930)						
YEAR	51.87	50.51	18934.20	18437.21	496.98	0.94	0.95
YEAR	(871001-880930)						
YEAR	49.36	52.05	18065.39	19052.11	-986.72	0.94	0.96
YEAR	(881001-890930)						
YEAR	47.28	50.07	17255.66	18274.75	-1019.09	0.90	0.94
WHOLE	PERIOD	(811001-890930)					
PERIOD	51.8	51.27	151352.00	149817.90	1534.09	0.92	0.93

To reconfirm Micovic's finding of low variability in eight modeling parameters of time distribution constants of runoff, groundwater percolation and deep zone share, the GA calibrated parameter values are compared with the suggested constant parameter values. Table 6-15 summarizes the calibrated values of modeling parameters against the suggested constant values from Micovic. Because Micovic's best calibrated Illecillewaet River model achieves high *e!* value each water year, for fairness, only GA calibrated results from combinations with *e!* values > 0.90 are considered (combinations 1A, 2A, 5A, and 9A are the only ones > 0.90).

Table 6-14: Statistics of Model Performance from Best Calibration of Illecillewaet River Watershed (Micovic 1998)

STATISTICS FOR THE OCT 1 , 1981 - SEP 30 , 1989 WATER YEARS

	Mean Q_{obs} *(cms/d)*	Mean Q_{est} *(cms/d)*	Tot Q_{obs} *(cms/d)*	Tot Q_{est} *(cms/d)*	Tot Q_{obs} -Tot Q_{est}	Coeff.of Eff	Coeff.of Det
YEAR	(811001-820930)						
YEAR	57.0	53.6	20787.69	19579.3	1208.39	0.9124	0.9227
YEAR	(821001-830930)						
YEAR	52.3	49.2	19083.01	17960.6	1122.41	0.9200	0.9228
YEAR	(831001-840930)						
YEAR	52.9	45.3	19376.00	16561.5	2814.5	0.8948	0.9131
YEAR	(841001-850930)						
YEAR	49.4	47.9	18045.53	17490.7	554.83	0.9429	0.9554
YEAR	(851001-860930)						
YEAR	54.3	49.1	19804.49	17924.7	1879.79	0.9172	0.9246
YEAR	(861001-870930)						
YEAR	51.9	52.2	18934.20	19045.2	-111	0.9125	0.9173
YEAR	(871001-880930)						
YEAR	49.4	52.6	18065.39	19255.2	-1189.81	0.8940	0.9089
YEAR	(881001-890930)						
YEAR	47.3	50.6	17255.66	18451.2	-1195.54	0.9003	0.9354
WHOLE	PERIOD	(811001-890930)					
PERIOD	51.8	50.0	151352.00	149817.90	5083.57	0.9087	0.9147

Table 6-15: Suggested Constant Values for Modeling Parameters with Low Variability vs. GA Calibrated Parameter Values of Illecillewaet River Watershed

Parameter	Lower Bound	Upper Bound	Suggested Constant Value	Micovic's Calibrated Value	Comb. 1 A	Comb. 2 A	Comb. 5 A	Comb. 9 A
POPERC	10	50	**25**	**31**	38.39	18.94	37.77	47.80
PODZSH	0	1	**0.3**	**0.25**	0.70	0.76	0.64	0.70
P0FRTK	0	2	**0.6**	**0.78**	1.14	1.39	1.46	1.96
P0FSTK	0	2	**1**	**1**	0.78	0.54	0.79	0.78
P0IRTK	3	10	**3**	**2**	6.84	7.50	6.90	7.45
P0ISTK	3	10	**4**	**3**	4.10	7.83	7.28	7.17
P0UGTK	10	50	**20**	**17**	20.67	17.37	21.14	21.14
P0DZTK	100	300	**150**	**168**	109.41	184.71	134.51	109.41

Unlike the similarity observed between the suggested constant value and the GA calibrated parameter values in the Campbell River watershed, the results in Illecillewaet River Watershed are relatively different from the suggested constant parameter values. The parameters showing some difference in values are POPERC, PODZSH, P0FRTK, P0ISTK and P0IRTK although P0ISTK and P0IRTK are low sensitivity parameters

which have little influence on the overall model performance. The largest difference is in P0DZSH. No reasonable explanation can be offered. It is possible that Illecillewaet River is one of the very few watersheds which exhibit high variability in parameters of time distribution constants of runoff, groundwater percolation and deep zone share, while the majority of 12 studied watersheds exhibit low variability.

6.5 Evaluation of Search Efficiency for Various Genetic Algorithms Techniques

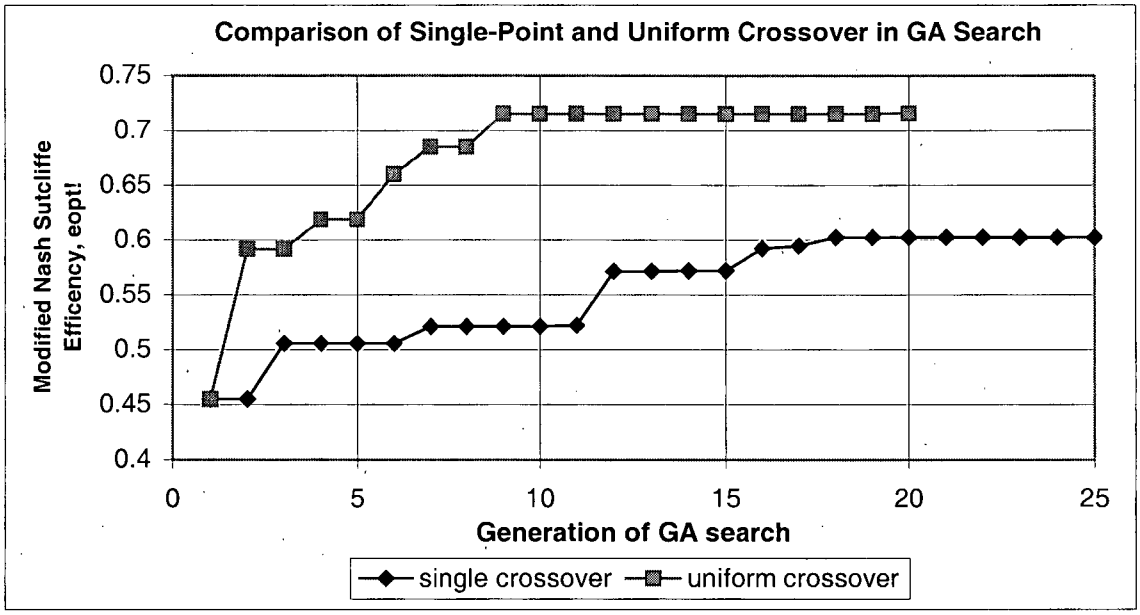
In Chapter 4, types of major genetic operators and their alternative forms were discussed. The effect of these operators in enhancing the GA search efficiency and facilitating the calibration process will be demonstrated in this section. Although six operators were introduced in Chapter 4, only three operators: crossover, elitism and niching will be demonstrated.

6.5.1 Comparison of Crossover Operators

In Figure 4-1, it was shown that the uniform crossover operator tends to outperform the single crossover operator in a simple single-objective optimization problem. To demonstrate how the type of crossover operator chosen can seriously affect the GA search efficiency in model calibration, combination 1 of the Campbell River watershed is used as a test. As shown in Figure 6-1, because the initial string population and all the genetic operators used for the calibration of Campbell River watershed are identical, except for the type of crossover operator, the initial *eopt!* values obtained were the same for the first generation. As more generations of search elapse, the *eopt!* values begin to depart from each other significantly. The calibration run with the uniform crossover operator is clearly the winner with an *eopt!* value of 0.716 at the end of 20-generation search while the run with the single-point crossover operator only achieves an *eopt!* value of 0.603 after 20 generations and remains unchanged despite that an additional five-generation search was carried out in the hope that the search efficiency would soon improve. Based on this observation, one may therefore conclude that the uniform

crossover operator should be used by the automatic GA calibrator to improve the GA search efficiency in the calibration process of a UBC Watershed Model. Note that in the two calibration runs, both elitism and niching operators were implicitly used.

Figure 6-1: Comparison of GA Search Efficiency using Single-Point and Uniform Crossover

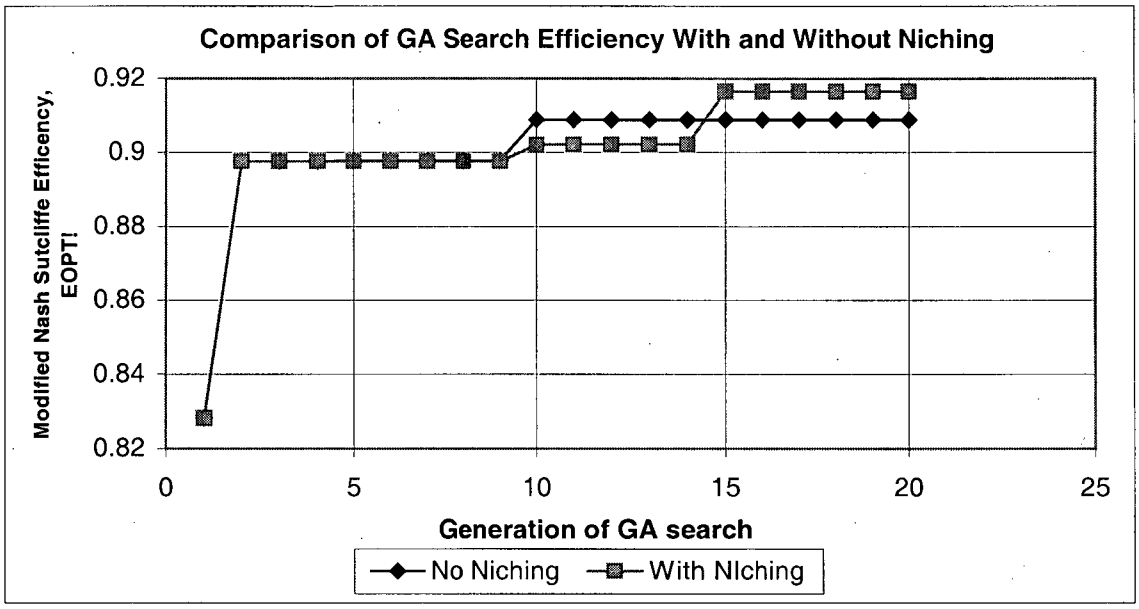


6.5.2 With and Without Niching Operator

The advantage of using the niching operator in genetic algorithms has been discussed in section 4.7.7. To demonstrate how the niching operator can be used to improve the GA search efficiency in the calibration process, combination 5 of the Illecillewaet River watershed is used. In the non-niching option, the niching operator is temporarily disabled in the GA search. The two slightly differently coded GA calibrators were then used to commence the search to find the best set of parameters with the maximal model performance, which is selected to be *eopt!*. As shown in Figure 6-2, the search performance seems to be identical until the 10^{th} generation when the non-niching option begins to take the lead. However, the niching option quickly catches up and reaches an *eopt!* of 0.917 at the end of the 20 generation search while the non-niching option only reaches an *eopt!* of 0.909, a difference of about 1%. Although an improvement of 1%

appears to be negligible superficially, this improvement is actually quite difficult to achieve if the trial-and-error calibration procedure is used. Thus, the niching operator will be used by the automatic GA calibrator to improve the GA search efficiency in the calibration process of the UBC Watershed Model.

Figure 6-2: Comparison of GA Search Efficiency with and without Niching



6.5.3 With and Without Elitism Operator

The elitism operator, as discussed in section 4.7.5, is used to preserve the solutions that achieve high fitness. The intent of using the elitism operator is to prevent genetic operations from accidentally destroying the best solution found so far. When the elitism operator is activated in the GA calibrator, it automatically preserves the best candidate solution of every generation. Only one candidate solution is allowed in this GA calibrator because De Jong (1975) reportedly found preserving more candidate solutions decreases the search performance of multi-modal objective functions.

Combination 1 of the Campbell River watershed is again used to exemplify the role of elitism in improving the GA search efficiency. The choice of using or not using the niching operator in conjunction with the elitism operator has been noted in this study to

have a profound impact, so much so that elitism may actually slow down the GA search process. It is warranted to discuss the combined effect of the niching and elitism operators interactively.

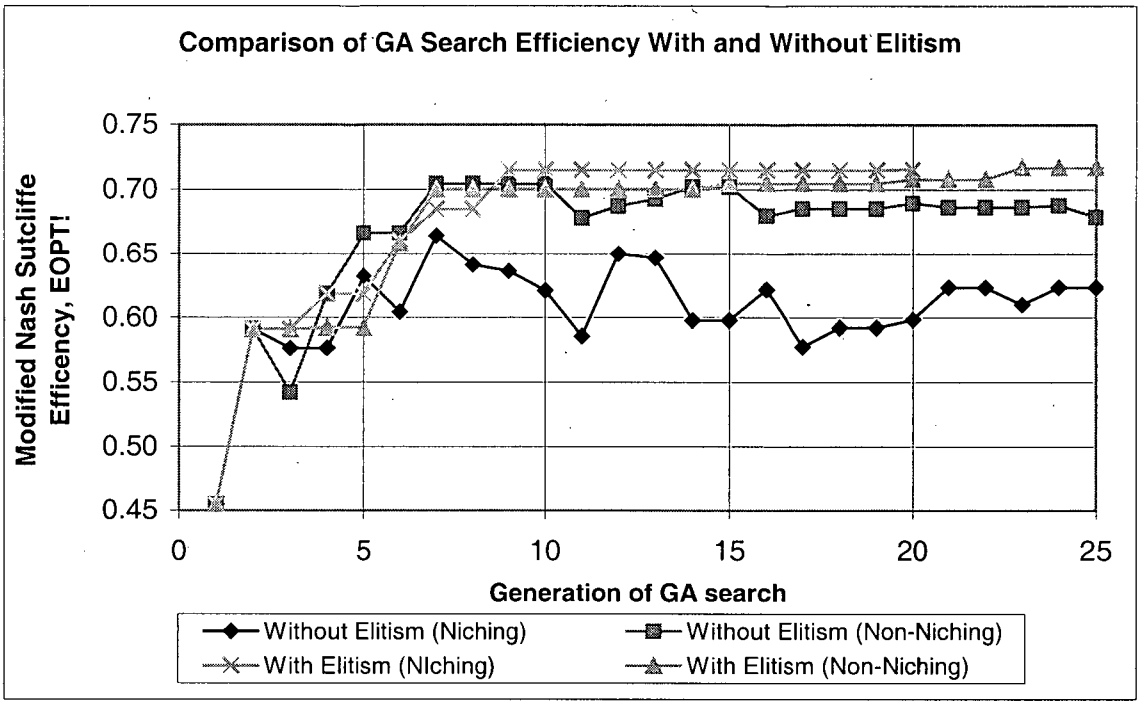
To investigate, assume all other genetic operators remain unchanged except the niching and elitism operators. Therefore, when niching is used without elitism in GA calibration, the objective function of the GA search is simply a sharing function, which helps to break the clustering of candidate solutions rather than an objective function capable of measuring model performance in terms of *e!* or *eopt!*. As a result, the sharing function will not be able to steer the GA search towards finding higher *e!* and *eopt!*. This understanding is confirmed in the GA calibration of Campbell River watershed as shown in Figure 6-3 when the resulting GA search for the combinatorial use of niching and non-elitism exhibits little or no steady trend of improved performance. On the other hand, if niching is not used, then without using elitism, the GA search could still gradually improve model performance and achieve high *e!* and *eopt!* values (as shown in Figure 6-3).

Figure 6-3 compares the GA search efficiency for the following elitism and niching usage:

- without elitism, with niching
- without elitism, without niching
- with elitism, with niching
- with elitism, without niching

Based on the empirical observation from Figure 6-3, one may conclude that if niching is used, then elitism should always be enabled in the GA calibration, otherwise the search efficiency will be relatively low.

Figure 6-3: Comparison of GA Search Efficiency with and without Elitism and Niching



* Uniform crossover is used in all four GA searches.

6.5.4 Summary of Default Genetic Algorithm Techniques and Parameters used for Model Calibration

Based on these findings on search efficiency, the following default genetic operator options were programmed in the GA model calibrator. It is believed that this combination of genetic operators will yield the best GA search efficiency and ensure that a global-optimal or at least a near global-optimal solution is found. However, the users still have the flexibility to change the default whenever necessary. The default genetic operator options are:

- Use random selection rather than tournament selection in reproduction,
- Use uniform type of crossover operator,
- Use elitism to preserve the best solution of each generation of GA search;
- Use niching to prevent clustering of candidate solutions.

6.5.5 Calibration Time

For consistency, all the GA calibration runs for the Campbell River and Illecillewaet River Watersheds use a population of 20 strings and an evolution of 20 generations unless specified otherwise. This requires the GA-based model calibrator to call the UBC Watershed Model 400 times in every run. On an Intel Celeron 533MHz MMX system, each calibration takes about 45 to 55 minutes to complete.

6.6 Comparison of Three Statistical Measures used as the Objective-Functions in GA Model Calibration

Three commonly used model performance measures have already been discussed earlier in Chapter 2. They will be tested in this section to see if they can be effectively used as objective functions in minimizing the differences between the observed and simulated data and maximizing the model performance in the calibration of the UBC Watershed Model. These three objective statistical measures are:

1. Nash-Sutcliffe Coefficient of Efficiency

$$e! = 1 - \frac{\sum_{i=1}^{n}(Q_{obs_i} - Q_{sim_i})^2}{\sum_{i=1}^{n}(Q_{obs_i} - Q_{obs_{mean}})^2} = 1 - \frac{residual\ variance}{total\ variance} \qquad \textbf{(Equation 6-1)}$$

2. Least Squares Difference Objective Function

$$z_{\substack{least \\ squares}} = 1 - \frac{\sum_{i=1}^{n}(1 - \frac{Q_{sim_i}}{Q_{obs_i}})^2}{n} \qquad \textbf{(Equation 2-6)}$$

3. Least Absolute Difference Objective Function

$$z_{\substack{absolute \\ difference}} = 1 - \frac{\sum_{i=1}^{n} abs(1 - \frac{Q_{sim_i}}{Q_{obs_i}})}{n} \qquad \textbf{(Equation 2-9)}$$

The testing procedures are follows:

- Allow a total string population of 20 and an evolution of 20 generations. The number of the population and the evolution are chosen arbitrarily though they are a compromise between long GA computational search time and a thorough search of the multi-dimensional solution space. Run the genetic algorithms model calibrator for the Illecillewaet River Watershed three times:

  - For the first run, use the Nash-Sutcliffe coefficient of efficiency alone as the objective function to indicate model performance and guide the GA search. However, upon the completion of a 20 generation-search, the surviving candidate solutions (strings) of the 20^{th} generation will have their corresponding least squares difference and least absolute difference objective function values implicitly calculated.
  - For the second run, use the least squares difference objective alone as the objective function to indicate model performance and guide the GA search. Upon the completion of a 20 generation-search, the surviving candidate solutions will have their corresponding Nash-Sutcliffe and least absolute difference objective function values implicitly calculated.
  - For the third run, use the least absolute difference objective alone as the objective function to indicate model performance and guide the GA search. Upon the completion of a 20 generation-search, the surviving candidate solutions will have their corresponding Nash-Sutcliffe and least squares difference objective function values implicitly calculated.

- Upon the completion of all three GA runs, graphically plot the corresponding streamflow volume discrepancy, *dV/V* against the three model performance measures (objective functions) respectively. The three statistical measures are then ranked based on how consistently the streamflow volume discrepancy decreases with increasing objective function values.

The streamflow volume discrepancy, *dV/V* used is non-dimensionalized with the total observed streamflow volume and is defined as:

$$\frac{dV}{V} = \frac{abs\left(V_{\substack{total \\ observed}} - V_{\substack{total \\ estimated}} \right)}{V_{\substack{total \\ observed}}} = abs\left(1 - \frac{\sum_{i=1}^{n} (Q_{sim})_i}{\sum_{i=1}^{n} (Q_{obs})_i} \right) \qquad \textbf{(Equation 6-2)}$$

where $V_{\substack{total \\ observed}}$ is the total observed streamflow volume integrated over the duration of the model simulation period,

$V_{\substack{total \\ estimated}}$ is the total estimated streamflow volume integrated over the duration of the same model simulation period.

Clearly, if one objective function is to be named a good model performance indicator in calibration, then the higher the value it is, the smaller the streamflow volume discrepancy (*dV/V*) should be. The inverse relationship is essential for an objective statistical measure to qualify as a good model performance indicator. Thus, the test devised for comparison in this section can fairly determine which objective statistical measure is a better and more consistent indicator of a well calibrated model, and allows the GA user to clearly see the strength or limitations of the three objective statistical measures for model performance.

For the first GA run, using the Nash-Sutcliffe as the objective function for GA search and calculating the corresponding least squares difference and least absolute difference objective functions, the relationships between *dV/V* and the three objective functions are graphically represented in Figure 6-4, Figure 6-5, and Figure 6-6.

Figure 6-4: Discrepancy of Streamflow Volume vs. Nash-Sutcliffe Coefficient of Efficiency (Run 1)

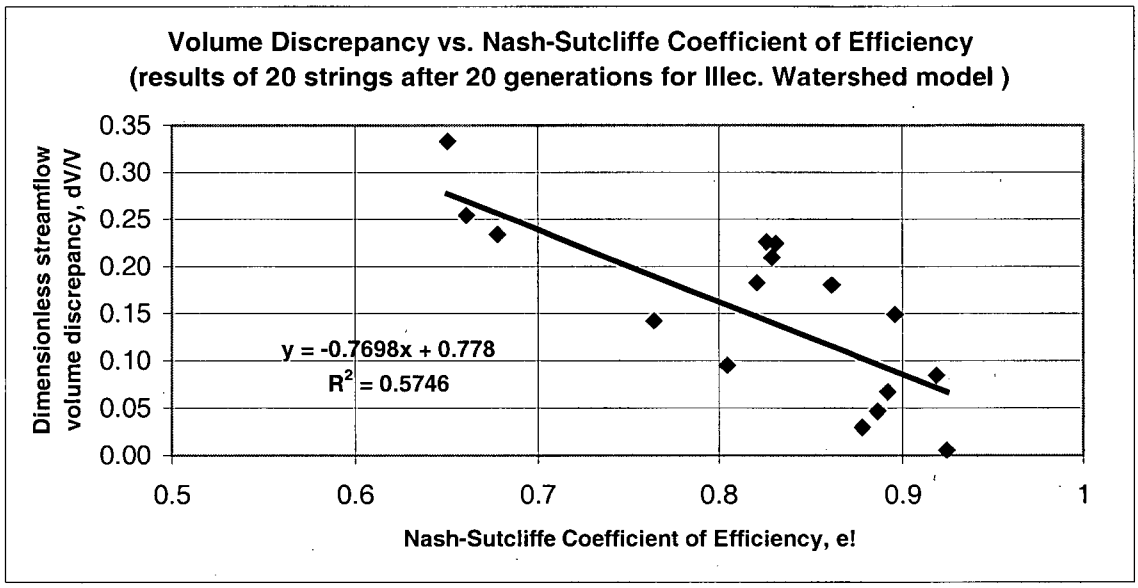


Figure 6-5: Discrepancy of Streamflow Volume vs. Least Squares Difference Objective Function (Run 1)

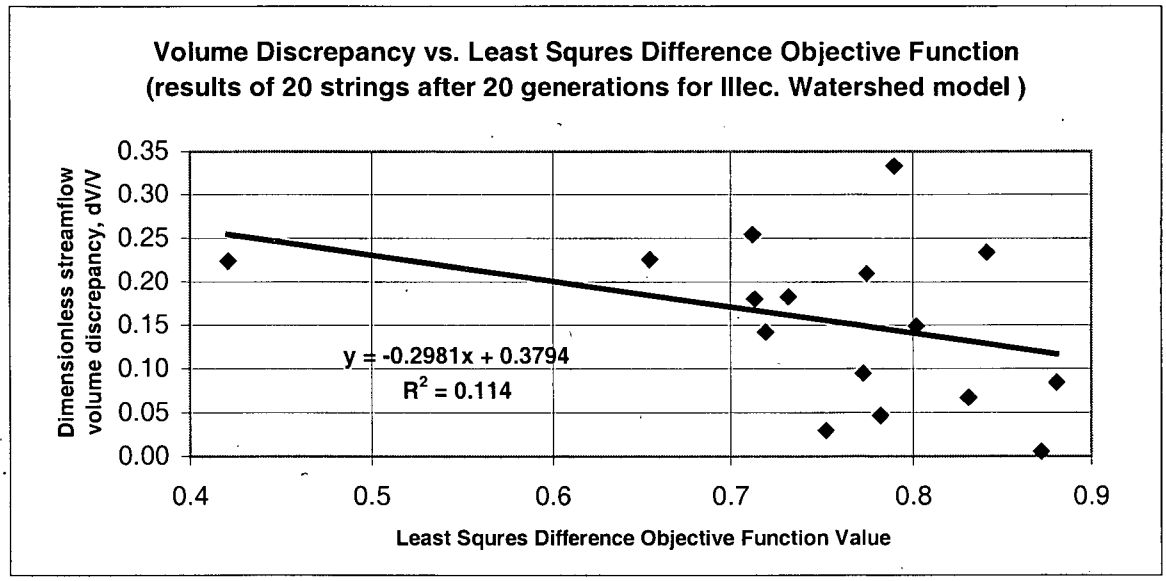
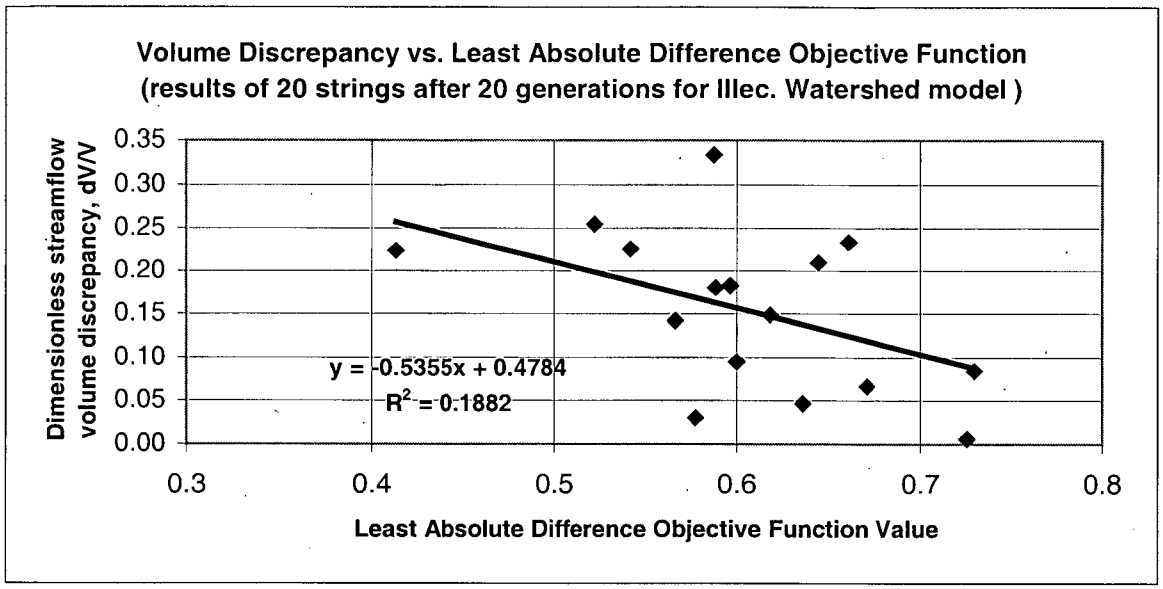


Figure 6-6: Discrepancy of Streamflow Volume vs. Least Absolute Difference Objective Function (Run 1)



As anticipated, the trend observed from Figure 6-4, Figure 6-5 and Figure 6-6 is that the discrepancy in streamflow volume generally decreases as the values of the three objective functions increase. In particular, the desirable inverse relationship between the discrepancy in streamflow volume and the objective functions for calibration is the strongest when the Nash-Sutcliffe coefficient of efficiency is used, although some inconsistency still remains. Ideally, a perfect model performance indicator in GA-based model calibration is an objective function in which the discrepancy in streamflow volume would always decrease as the value of chosen objective function increases.

In addition, as shown in Figure 6-4 the values of the Nash-Sutcliffe coefficient on X-axis is quite high because it is the objective function used to guide GA search. On the contrary, the ranges of the least squares and least absolute difference objective functions on the X-axis in Figures 5.2 and 5.3 tend to be much lower because they are not the guiding objective function in the GA search. Because the niching operator is used, the Nash-Sutcliffe coefficient values have a wide range on X-axis. If the niching operator were not used, the Nash-Sutcliffe coefficient values of all candidate solution would cluster tightly together near x= 0.93.

For the second GA run, the least squares difference is used as the objective function to guide the GA search and the corresponding Nash-Sutcliffe coefficient of efficiency and least absolute difference objective function are then implicitly calculated. The relationships between the dimensionless volume discrepancy and the three objective functions are graphically represented in Figure 6-7, Figure 6-8, and Figure 6-9.

From Figure 6-7 and Figure 6-9, one can again observe that *dV/V* generally decreases as the values of the two objective functions increases. The trend is particularly strong between *dV/V* and the Nash-Sutcliffe coefficient. However, in Figure 6-8, no apparent trend can be determined between *dV/V* and the least squares difference objective function despite the fact that the least squares difference objective function is the chosen objective function to guide the GA search. Thus under this circumstance, the least squares difference objective function fails to be a good indicator of the model performance in the calibration process while the Nash-Sutcliffe coefficient and the least absolute difference objective function behave reasonably well as good performance indicators. However, similar to the results of run 1, it appears in run 2 that the Nash-Sutcliffe coefficient behaves most consistently with a decreasing trend of *dV/V*, as desired.

Figure 6-7: Discrepancy of Streamflow Volume vs. Nash-Sutcliffe Coefficient of Efficiency (Run 2)

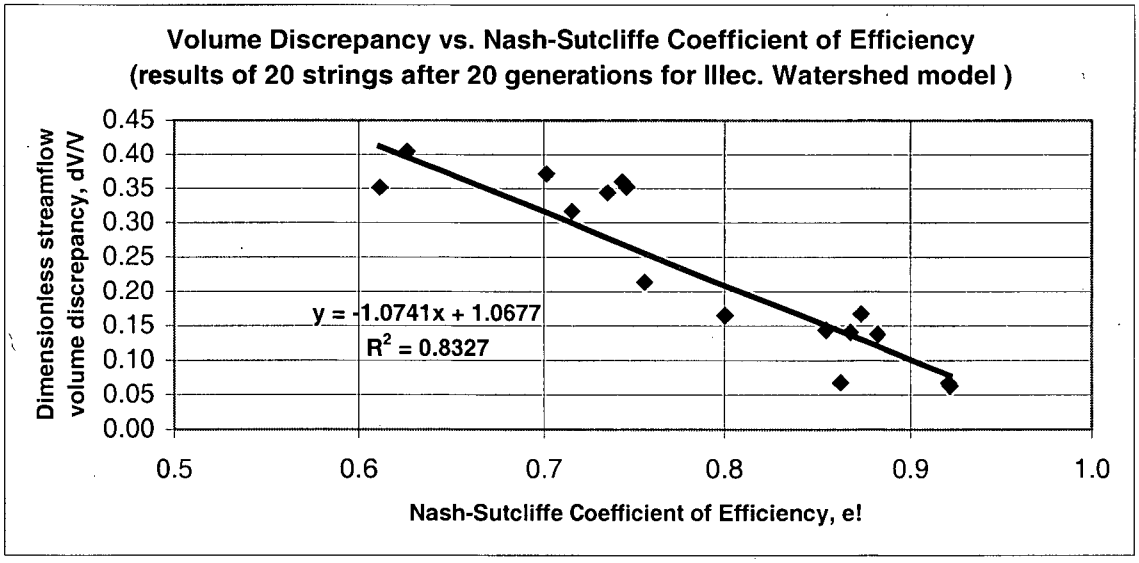


Figure 6-8: Discrepancy of Streamflow Volume vs. Least Squares Difference Objective Function (Run 2)

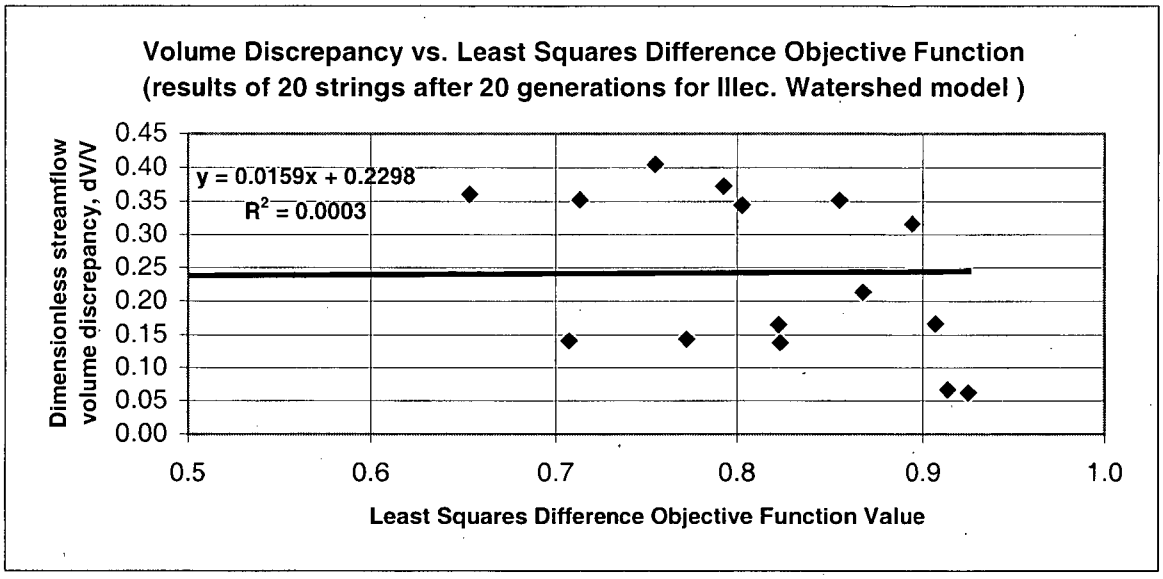
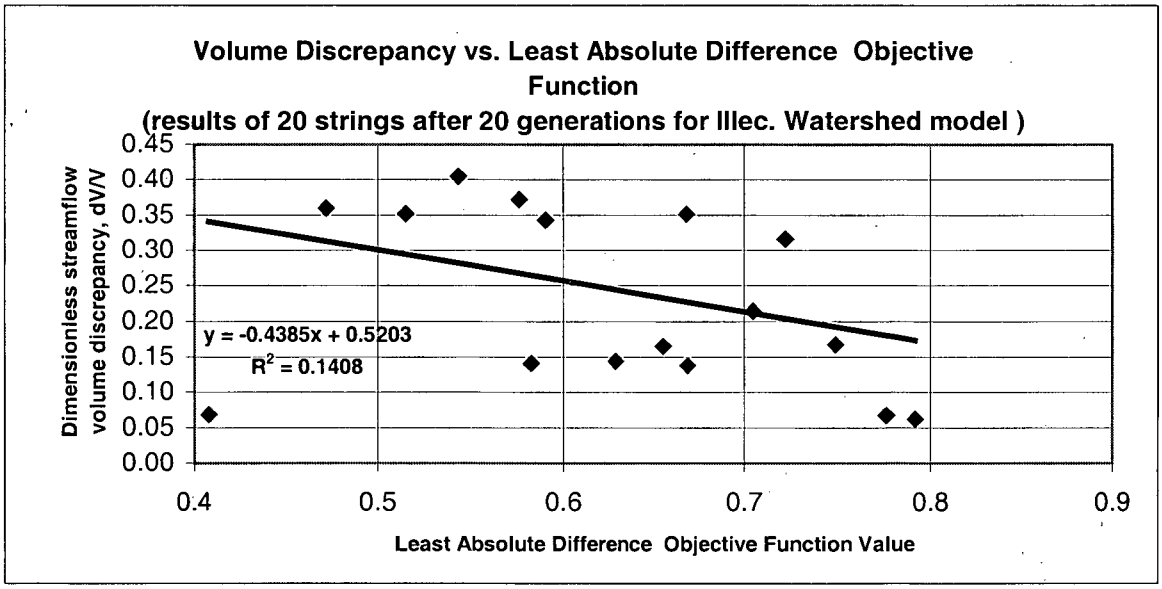


Figure 6-9: Discrepancy of Streamflow Volume vs. Least Absolute Difference Objective Function (Run 2)



For the third GA run, using least absolute difference as the objective function to guide GA search and then calculating the corresponding Nash-Sutcliffe coefficient of efficiency and least squares difference objective function, the relationships between the

dimensionless volume discrepancy and the three objective functions are then graphically represented in Figure 6-10, Figure 6-11, and Figure 6-12.

Figure 6-10: Discrepancy of Streamflow Volume vs. Nash-Sutcliffe Coefficient of Efficiency (Run 3)

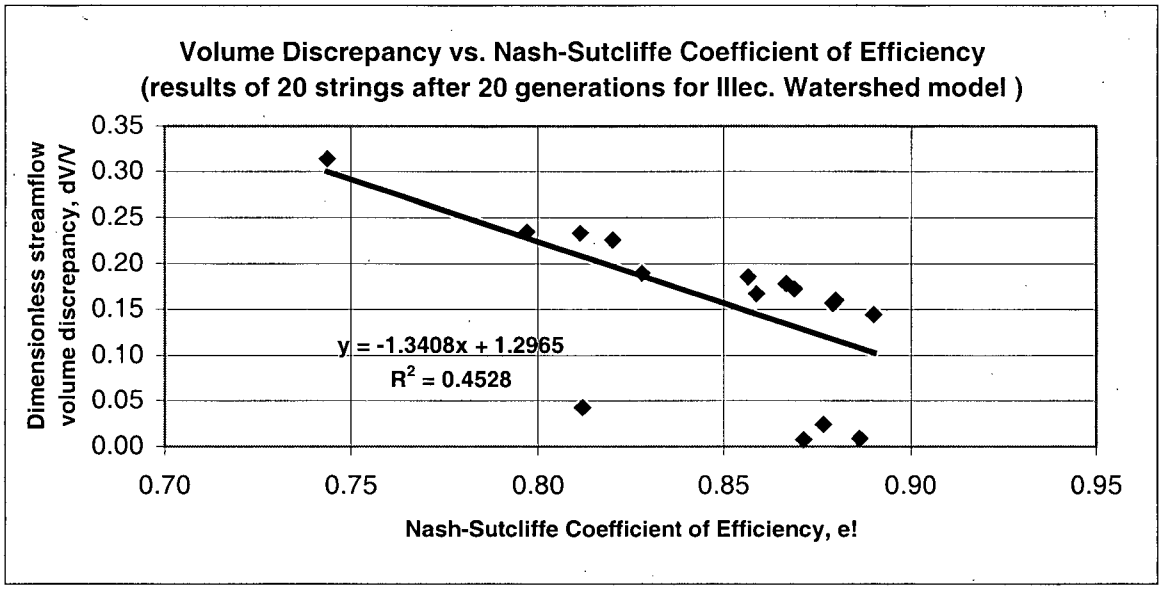


Figure 6-11: Discrepancy of Streamflow Volume vs. Least Squares Difference Objective Function (Run 3)

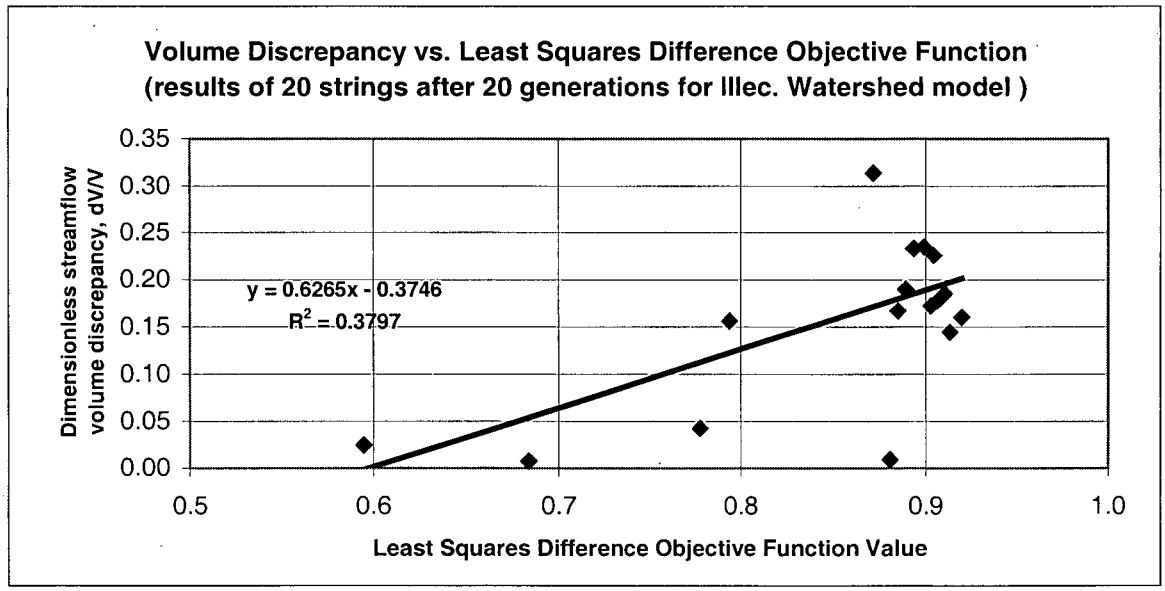
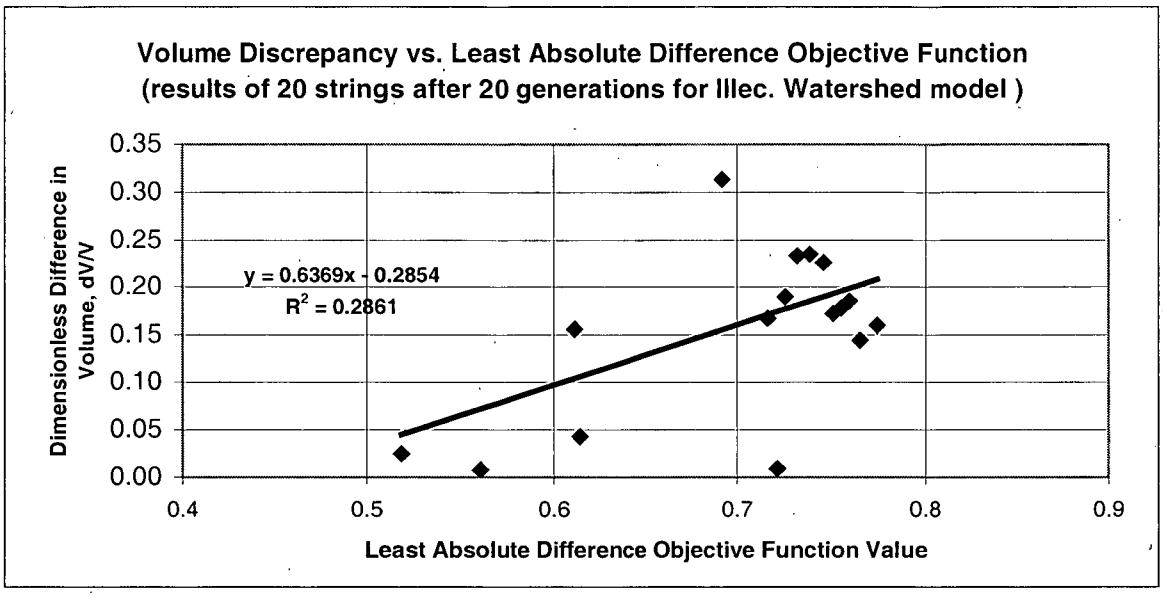


Figure 6-12: Discrepancy of Streamflow Volume vs. Least Absolute Difference Objective Function (Run 3)



From Figure 6-10, as anticipated, one can again observe that *dV/V* generally decreases as the value of Nash-Sutcliffe coefficient increases. The inverse trend remains reasonably strong. However, against what is anticipated, Figure 6-11 and Figure 6-12 show that dV/V values do not decrease inversely as the values of least squares and least absolute difference objective functions increase, despite the fact that the least absolute difference objective function is the chosen objective function to guide the GA search. Thus, under this circumstance, both least squares and least absolute difference objective functions fail to behave as good model performance indicators in minimizing streamflow volume discrepancy. However, the Nash-Sutcliffe coefficient, unlike the other two objective functions, continues to behave reasonably well as a model performance indicator, so that *dV/V* decreases consistently as the Nash-Sutcliffe coefficient increases.

Based on the observation of the results from three GA runs, one may conclude that of the three objective statistical measure discussed in the thesis, the Nash-Sutcliffe coefficient behaves most consistently with a strong decreasing trend of *dV/V* as the value of the Nash-Sutcliffe coefficient increases. However from Figure 6-4, Figure 6-7 and Figure 6-10, one can clearly observe that the value of *dV/V* does not always decrease with an increasing Nash-Sutcliffe coefficient. To overcome this inherent limitation of the Nash-

Sutcliffe coefficient, a new statistical measure based on a combination of the Nash-Sutcliffe coefficient and *dV/V* is therefore used.

The modified statistical measure, which is briefly mentioned in Chapter 2 (Equation 2-3), places equal weights on both the Nash-Sutcliffe coefficient of efficiency and the agreement in total streamflow volume. The resulting combinatory statistical measure is already used as a modification of the Nash-Sutcliffe coefficient in the existing UBC Watershed Model calibration, and can be written as:

$$eopt! = e! - abs\left(1 - \frac{\sum_{i=1}^{n}(Q_{sim})_i}{\sum_{i=1}^{n}(Q_{obs})_i}\right) = e! - abs\left(\frac{dV}{V}\right) \qquad \textbf{(Equation 6-3)}$$

It emphasizes that for a model to be considered to be well calibrated, the observed and simulated streamflow data should concurrently achieve a high Nash-Sutcliffe coefficient of efficiency and small discrepancy in streamflow volume.

7.0 CONCLUSIONS

The usefulness of a watershed model or any model in general depends on how well it is calibrated, namely, how closely it actually predicts the physical behavior of the modeled system. Under the existing UBC Watershed Model structure, an automatic random search capability is provided in the calibration module. The calibration module limits the number of simultaneously calibrated modeling parameters to groups of about three to six at a time. The user then proceeds through further groups of parameters, moving from the more sensitive parameters to the less sensitive ones, to refine the parameter values.

Therefore, in this thesis, the aim was to develop a calibration method based on the genetic algorithm approach. This approach would permit the simultaneous evaluation of all modeling parameters, which could be very useful because of the non-linear interactions between the parameters. Although the best solution found in the GA search cannot be analytically proven as the optimal solution, by designing the GAs to maintain sufficient diversity, the search should adequately cover the entire solution space. Therefore, there should be no other superior results within the permissible variable ranges.

This thesis describes the use of genetic algorithms in the development of a non-conventional optimization and search technique, which is used to develop an automatic model calibrator for the UBC Watershed Model.

The calibrator allows all modeling parameters to be simultaneously evaluated. Using this GA calibrator, two well-studied watersheds in British Columbia: Campbell River and Illecillewaet River watersheds were successfully calibrated. The best model performance of the GA calibrated Campbell River and Illecillewaet River watersheds achieves *eopt!* values of 0.718 and 0.914, respectively for the entire duration of the simulated water years whilst Micovic (1998) reported best *eopt!* values of 0.720 and 0.875, obtained using the existing UBC random search procedure. This may appear to be only a marginal

improvement, but the main difference is the reduced time and effort required to achieve a reasonable calibration.

Because the types of genetic operators jointly used in a GA search can affect the search efficiency, several computational experiments were conducted to investigate the impact crossover, elitism, and niching operators. The experimental results in the GA calibration of the Campbell River and Illecillewaet River watersheds show the following:

1. The uniform crossover operator tends to out-perform the single crossover operator in search efficiency.
2. In presence of an elitism operator, the GA calibrator equipped with a niching operator performs slightly better than the one without a niching operator. Even though the small improvement appears to be negligible, experienced modelers may often find the improvement quite difficult to achieve through a random trial-and-error calibration procedure.
3. The choice of using or not using an niching operator in conjunction with the elitism operator has been noted in this study to have a profound impact on the elicited advantage of using an elitism operator, so much so that elitism may actually slow down the GA search process.
4. If niching is used, then an elitism operator should also be concurrently in the GA calibration, otherwise the search efficiency will be relatively low.

To objectively determine the performance of a calibrated watershed model, the difference between the observed and the simulated streamflow has to be statistically measured. The five statistical measures introduced in this study are:

1. Coefficients of Linear Correlation and Determination, r and r^2
2. Nash & Sutcliffe Coefficient of Efficiency, *e!*
3. Least Squares Objective Function
4. Least Absolute Difference Objective Function
5. Modified Nash & Sutcliffe Coefficient of Efficiency, *eopt!*

The coefficients of linear correlation and determination were shown to measure only shape similarity between the observed and simulated streamflow hydrographs and fail to acknowledge the discrepancy error in streamflow volume. Thus, they were quickly discarded and not used any further. To fairly compare the Nash & Sutcliffe coefficient of efficiency, least squares and least absolute difference objective functions as model performance indicators, the relationships between the streamflow volume discrepancy (*dV/V*) and the values of the three statistical measures were experimentally obtained and graphically plotted. Experiment results obtained in the GA calibration of the Campbell River River watershed show that the Nash & Sutcliffe coefficient of efficiency exhibits the most consistently decreasing trend of *dV/V* as the value of *e!* increases. For a statistical measure to be named a good model performance indicator in calibration, there must exist an inverse relationship, i.e. the higher the statistical measure value is, the smaller the streamflow volume discrepancy should be. Thus, the Nash & Sutcliffe coefficient best indicates the agreement between the observed and simulated streamflow hydrographs. However, because the value of *dV/V* does not always decrease with an increasing Nash-Sutcliffe coefficient, a modified coefficient was suggested. The modified coefficient places equal weights on both the Nash-Sutcliffe coefficient of efficiency and the streamflow volume discrepancy. It ensures the optimal or near-optimal set of model parameter values found at the end of a GA search achieves both high *e!* and low *dV/V* at the same time.

In calibrating the Illecillewaet River watershed, it is found that meteorological data collected from a low-elevation station can be adjusted through precipitation gradient factors and AES precipitation factors to successfully represent the condition in a high-elevation watershed and achieve high model performance in terms of *e!* and *eopt!*. This implies that a good set of meteorological data can still yield high model accuracy without strictly following the principle of assigning meteorological data to elevation bands of a watershed based on their elevation proximity. The finding is useful because in some remote watersheds where there is no meteorological station and no data available for

modeling purposes, the modeler can "borrow" meteorological data from adjacent watershed (relatively speaking) with a station.

In addition, it was found in the GA computational experiments that within the multi-dimensional feasible solution region, bounded by the modeling parameter constraints, objective function such as *eopt!* can often be multi-modal with the values of sub-optimal solutions very close to the value of the optimal solution found. Under these circumstances, the GA user is cautioned because the minor difference in objective function values can be well within the error range of the observed streamflow data used.

For consistency, all the GA calibration runs for the Campbell River and Illecillewaet River Watersheds use a population of 20 strings and an evolution of 20 generations unless specified otherwise. This requires the GA-based model calibrator to call the UBC Watershed Model 400 times in every run. Thus the computational efficiency in UBC Watershed Model should be re-examined to see if further coding improvement can be made in reducing the overall GA search time. On an Intel Celeron 533MHz MMX system, each calibration takes about 45 to 55 minutes to complete.

8.0 REFERENCES

Begley, S. (1995), "Software au Naturel." *Newsweek*, May 8th 1995, 70-71.

Carroll D. J. (1999), Internet website: http://www.staff.uiuc.edu/~carroll/ga.html.

De Jong, K. A. (1975), "An Analysis of the Behavior of a Class of Genetic Adaptive Systems." *Ph.D. Dissertation*, University of Michigan, Ann Arbor, Michigan.

Frey, J., Simpson, A. R., Dandy, G. C. and Murphy, L. J. (1995), "A New Breakthrough Technology Optimizes Distribution System Design and Operations." *Proceedings of AWWA Distribution Symposium* (September 12, 1995), Nashville, Tennessee.

Goldberg, D. E. (1989), *Genetic Algorithms in Search, Optimization and Machine Learning,* Addison-Wesley, Reading, Massachusetts.

Goldberg, D. E. and Kuo, C.H. (1987), "Genetic Algorithms in Pipeline Optimization." *Journal of Computing in Civil Engineering,* Vol. 1, No. 2, 128-141.

James, W. (2001), *Personal Communication.*

Holland, J. H. (1975), *Adaptation in Natural and Artificial Systems,* MIT Press, Cambridge, Massachusetts.

Hollestin, R.B. (1971), "Artificial Genetic Adaptation in Computer Control Systems." *Ph.D. Dissertation*, University of Michigan, Ann Arbor, Michigan.

Karney, B., (2000), *Hydraulic Transient Analysis for Proposed Sapperton Forcemain Section*, Report prepared for Greater Vancouver Regional District, Burnaby, British Columbia, Canada.

King, J. P., Fahmy, H. S. and Wentzel, M. W. (1997), "A Genetic Algorithm Approach for River Management." *Evolutionary Algorithms in Engineering Applications* (edited by Dasgupya, D. and Michalewicz, Z.), Springer, Berlin, Germany, 117-134.

Kouwen, N. (1997), *WaterFlood/SPL8 Flood Forecasting System*, Department of Civil Engineering, University of Waterloo, Waterloo, Ontario, Canada.

Nash, J. E., Sutcliffe, J. V. (1970), "River Flow Forecasting Through Conceptual Models." *Journal of Hydrology*, Vol.10, 282-290.

MW Soft Inc. (1999), *H2ONET Calibrator Users Manual*, MW Soft Inc., Pasadena, California.

Micovic, Z., (1998), "Regional Flow Estimation using a Hydrological Model." *M. A. Sc. Thesis*, The University of British Columbia, Vancouver, British Columbia, Canada.

Obitko, M. (1998), Internet website: http://cs.felk.cvut.cz/~xobitko/ga.

Quick, M. C. and Pipes, A., (1977), "Nonlinear Channel Routing by Computer." *Journal of Hydraulics Division*, ASCE, 101(HY6), 651-664.

Savic, D. A. and Walter, G. A. (1996), "Genetic Algorithms for Least Cost Design of Water Distribution Networks." *Journal of Water Resources Planning and Management*, ASCE, Vol.123, No. 2, March/April, 67-77.

Solomatine, D. P. (1998), "Genetic and Other Global Optimization Algorithms-Comparison and Use in Model Calibration." *Proceedings of International Conference Hydroinformatics 1998*, A. A. Balkema, Rotterdam, Netherlands.

Sorooshian, S. and Gupta, V. K. (1995), "Model Calibration." *Computer Model of Watershed Hydrology* (edited by Singh V. P.), Water Resources Publications, Highlands Ranch, Colorado, 23-68.

Tolson, B. A. (2000), "Genetic Algorithms for Least Multi-Objective Optimization in Water Quality Management under Uncertainty." *M. A. Sc. Thesis*, The University of British Columbia, Vancouver, British Columbia, Canada.

Wu, Z. Y., Boulos, B. F., Orr, C. H., and Ro, J. J. (2000), "An Efficient Genetic Algorithm Approach to an Intelligent Decision Support System for Water Distribution Networks." *Proceedings of Hydroinformatics 2000* (July 24-26), Iowa City, Iowa.

UBC Mountain Hydrology Group, (1995), *UBC Watershed Model manual version 4*, Mountain Hydrology Group, Department of Civil Engineering, University of British Columbia, Vancouver, British Columbia, Canada.

9.0 APPENDIX

The following model performance statistics are generated based on the best-calibrated set of model parameter values from each combination of the respective watershed. Only the combinations with high *e!* and *eopt!* values are given in the appendix. The statistical summaries of GA results for combinations with low *e!* and *eopt!* values are not provided.

Table: 9-1 Statistics of Model Performance for Combination 4 of Campbell River Watershed after GA Calibration

STATISTICS FOR THE OCT 1, 1983 - SEP 30, 1990 WATER YEARS

	Mean Q_{obs} *(cms/d)*	Mean Q_{est} *(cms/d)*	Tot Q_{obs} *(cms/d)*	Tot Q_{est} *(cms/d)*	Tot Q_{obs} -Tot Q_{est}	Coeff.of Eff	Coeff.of Det
YEAR	(831001-840930)						
YEAR	75.49	72.24	27631.09	26441.55	1189.54	0.61	0.61
YEAR	(841001-850930)						
YEAR	59.65	56.73	21773	20704.92	1068.08	0.59	0.72
YEAR	(851001-860930)						
YEAR	75.47	70.23	27546.7	25635.55	1911.15	0.78	0.85
YEAR	(861001-870930)						
YEAR	89.59	87.77	32701.29	32036.22	665.07	0.66	0.67
YEAR	(871001-880930)						
YEAR	70.23	74.49	25705.9	27263.96	-1558.06	0.65	0.67
YEAR	(881001-890930)						
YEAR	63.57	71.07	23203.41	25940.29	-2736.88	0.68	0.7
YEAR	(891001-900930)						
YEAR	65.46	67.85	23893.69	24763.46	-869.77	0.82	0.82
WHOLE	PERIOD	(831001-900930)					
PERIOD	71.36	71.48	182455.2	182786	-330.78	0.69	0.69

Table 9-2: Statistics of Model Performance for Combination 5 of Campbell River Watershed after GA Calibration

STATISTICS FOR THE OCT 1, 1983 - SEP 30, 1990 WATER YEARS

	Mean Q_{obs} *(cms/d)*	Mean Q_{est} *(cms/d)*	Tot Q_{obs} *(cms/d)*	Tot Q_{est} *(cms/d)*	Tot Q_{obs} -Tot Q_{est}	Coeff.of Eff	Coeff.of Det
YEAR	(831001-840930)						
YEAR	75.49	69.33	27631.09	25374.31	2256.78	0.63	0.64
YEAR	(841001-850930)						
YEAR	59.65	57.66	21773	21044.68	728.31	0.73	0.79
YEAR	(851001-860930)						
YEAR	75.47	69.41	27546.7	25334.63	2212.08	0.8	0.83
YEAR	(861001-870930)						
YEAR	89.59	88.35	32701.29	32249.3	451.99	0.66	0.67
YEAR	(871001-880930)						
YEAR	70.23	72.82	25705.9	26653.55	-947.65	0.67	0.67
YEAR	(881001-890930)						
YEAR	63.57	69.35	23203.41	25311.6	-2108.19	0.69	0.71
YEAR	(891001-900930)						
YEAR	65.46	70.53	23893.69	25743.13	-1849.44	0.76	0.78
WHOLE	PERIOD	(831001-900930)					
PERIOD	71.36	71.06	182455.2	181711.3	743.92	0.71	0.71

Table 9-3: Statistics of Model Performance for Combination 6 of Campbell River Watershed after GA Calibration

STATISTICS FOR THE OCT 1, 1983 - SEP 30, 1990 WATER YEARS

	Mean Q_{obs} *(cms/d)*	Mean Q_{est} *(cms/d)*	Tot Q_{obs} *(cms/d)*	Tot Q_{est} *(cms/d)*	Tot Q_{obs} -Tot Q_{est}	Coeff.of Eff	Coeff.of Det
YEAR	(831001-840930)						
YEAR	75.49	72.35	27631.09	26481.24	1149.85	0.68	0.68
YEAR	(841001-850930)						
YEAR	59.65	57.5	21773	20986.78	786.21	0.65	0.77
YEAR	(851001-860930)						
YEAR	75.47	68.63	27546.7	25048.95	2497.75	0.81	0.83
YEAR	(861001-870930)						
YEAR	89.59	86.8	32701.29	31680.69	1020.6	0.68	0.68
YEAR	(871001-880930)						
YEAR	70.23	72.92	25705.9	26689.6	-983.69	0.63	0.66
YEAR	(881001-890930)						
YEAR	63.57	69.87	23203.41	25504.27	-2300.86	0.72	0.73
YEAR	(891001-900930)						
YEAR	65.46	70.58	23893.69	25763.12	-1869.43	0.8	0.82
WHOLE	PERIOD	(831001-900930)					
PERIOD	71.36	71.24	182455.2	182155	300.19	0.72	0.72

Table 9-4: Statistics of Model Performance for Combination 2 of Illecillewaet River Watershed after GA Calibration

STATISTICS FOR THE OCT 1, 1981 - SEP 30, 1989 WATER YEARS

	Mean Q_{obs} *(cms/d)*	Mean Q_{est} *(cms/d)*	Tot Q_{obs} *(cms/d)*	Tot Q_{est} *(cms/d)*	Tot Q_{obs} -Tot Q_{est}	Coeff.of Eff	Coeff.of Det
YEAR	(811001-820930)						
YEAR	56.95	53.39	20787.69	19487.56	1300.13	0.89	0.90
YEAR	(821001-830930)						
YEAR	52.28	54.23	19083.01	19792.51	-709.5	0.92	0.92
YEAR	(831001-840930)						
YEAR	52.94	48.89	19376.00	17895.16	1480.85	0.93	0.93
YEAR	(841001-850930)						
YEAR	49.44	47.26	18045.53	17249.23	796.3	0.91	0.91
YEAR	(851001-860930)						
YEAR	54.26	49.84	19804.49	18190.78	1613.71	0.92	0.92
YEAR	(861001-870930)						
YEAR	51.87	54.38	18934.2	19847.91	-913.71	0.92	0.92
YEAR	(871001-880930)						
YEAR	49.36	53.94	18065.39	19742.17	-1676.78	0.93	0.94
YEAR	(881001-890930)						
YEAR	47.28	52.67	17255.66	19223.66	-1968	0.89	0.92
WHOLE	PERIOD	(811001-890930)					
PERIOD	51.8	51.77	151352	151279.6	72.39	0.91	0.91

Table 9-5: Statistics of Model Performance for Combination 5 of Illecillewaet River Watershed after GA Calibration

STATISTICS FOR THE OCT 1, 1981 - SEP 30, 1989 WATER YEARS

	Mean Q_{obs} *(cms/d)*	Mean Q_{est} *(cms/d)*	Tot Q_{obs} *(cms/d)*	Tot Q_{est} *(cms/d)*	Tot Q_{obs} -Tot Q_{est}	Coeff.of Eff	Coeff.of Det
YEAR	(811001-820930)						
YEAR	56.95	52.84	20787.69	19286	1501.69	0.95	0.95
YEAR	(821001-830930)						
YEAR	52.28	52.4	19083.01	19127.8	-44.79	0.92	0.92
YEAR	(831001-840930)						
YEAR	52.94	45.64	19376	16704.22	2671.79	0.86	0.88
YEAR	(841001-850930)						
YEAR	49.44	47.56	18045.53	17359.05	686.48	0.94	0.94
YEAR	(851001-860930)						
YEAR	54.26	49.95	19804.49	18232.97	1571.52	0.92	0.93
YEAR	(861001-870930)						
YEAR	51.87	55.57	18934.2	20284.32	-1350.12	0.93	0.94
YEAR	(871001-880930)						
YEAR	49.36	55.17	18065.39	20192.52	-2127.13	0.92	0.94
YEAR	(881001-890930)						
YEAR	47.28	54.44	17255.66	19871.84	-2616.17	0.88	0.93
WHOLE PERIOD	(811001-890930)						
PERIOD	51.8	51.7	151352	151058.8	293.19	0.92	0.92

Table 9-6: Statistics of Model Performance for Combination 9 of Illecillewaet River Watershed after GA Calibration

STATISTICS FOR THE OCT 1, 1981 - SEP 30, 1989 WATER YEARS

	Mean Q_{obs} *(cms/d)*	Mean Q_{est} *(cms/d)*	Tot Q_{obs} *(cms/d)*	Tot Q_{est} *(cms/d)*	Tot Q_{obs} -Tot Q_{est}	Coeff.of Eff	Coeff.of Det
YEAR	(811001-820930)						
YEAR	56.95	53.02	20787.69	19353.97	1433.72	0.95	0.95
YEAR	(821001-830930)						
YEAR	52.28	52.4	19083.01	19127.23	-44.22	0.9	0.91
YEAR	(831001-840930)						
YEAR	52.94	45.35	19376	16599.43	2776.57	0.86	0.88
YEAR	(841001-850930)						
YEAR	49.44	47.8	18045.53	17445.98	599.55	0.94	0.95
YEAR	(851001-860930)						
YEAR	54.26	49.89	19804.49	18210.96	1593.53	0.92	0.94
YEAR	(861001-870930)						
YEAR	51.87	55.55	18934.2	20275.71	-1341.51	0.93	0.94
YEAR	(871001-880930)						
YEAR	49.36	55.12	18065.39	20174.93	-2109.54	0.93	0.94
YEAR	(881001-890930)						
YEAR	47.28	54.57	17255.66	19919.46	-2663.79	0.88	0.92
WHOLE	PERIOD	(811001-890930)					
PERIOD	51.8	51.71	151352	151107.7	244.3	0.92	0.92