Department of Mathematics and Statistics

Statistical Methods for History Matching of Hydrological Model

Dewi Tjia

This thesis is presented for the Degree of
Doctor of Philosophy
of
Curtin University

December 2016

Declaration

To the best of my knowledge and belief, this thesis contains no material previously published by any other person except where due acknowledgement has been made.

This thesis contains no material which has been accepted for the award of any other degree or diploma in any university.

.....

Dewi Tjia

December 2016

Abstract

Hydrological models have been widely used to study catchment runoff processes since the early 1970s. More recently rainfall-runoff models (RRMs) have been used for water resource estimation and management. A RRM comprises systems of mathematical equations that describe the geological and hydrological characteristics of catchments. For a given set of inputs of climatic data and catchment characteristics, a RRM computes outputs of interest such as streamflow and/or total salt. The most challenging part in implementation of a RRM is estimation of the input catchment characteristics based on primitive geological information available about the catchment. This problem can be solved by calibrating input parameters of the catchment to match historical rainfall-runoff responses. This process is known as history matching (HM). HM is a challenging task as RRMs are often nonlinear and computationally complex.

In this study, we present HM for three catchments in Western Australia: the Dandalups (700 km^2), Bates (2 km^2), and Lewis (2 km^2). The Dandalups and Bates catchment have historical streamflow data between 1960 and 1992, and between 1988 and 2010 respectively. Meanwhile, for Lewis catchment historical data of streamflow and total salt are available for period between 1992 and 2009. The RRM model employed in this study is Land Use Change Incorporated CATchment (LUCICAT) developed by Bari [108] for Department of Water, Western Australia. HM methods explored here are Gauss Levenberg Marquardt (GLM), ant colony optimization (ACO_R and $DACO_R$) and robust parameter estimation (ROPE). The latter three are multiple solution-based methods. ACO_R and $DACO_R$ mimic the swarm intelligence of ants' collective behaviours, while ROPE works on the concept of data depth. These methods explore multiple solutions. The performance of these methods for different catchments is compared in terms of the accuracy via root mean square error (RMSE), results consistency/stability, computational time and reduction of initial input parameter uncertainties. The above implementation is referred to as direct HM, wherein the RRM is executed in every iteration of HM process.

We next consider indirect HM, where proxy models for a RRM are used in the HM process. This approach significantly reduces HM computational time, especially in instances where a RRM or HM methods are computationally intensive. The proxy models are developed using multidimensional Kriging and artificial neural networks (ANNs) with Backprop and Rprop learning algorithms. We conduct sensitivity analysis test by con-

sidering the size of training data (50 and 100) before proxy models are used for HM. We additionally test number of nodes in hidden layer for ANN with Backprop and Rprop, and number of hidden layers for ANN with Rprop. The indirect HM is used for calibrating model parameter for the Dandalups catchment. The performances of indirect HM are compared with direct HM approach using the same criteria as above.

Across the catchments, all HM methods provide improvement within the range of 10% to 50% from the best expert guided industry estimates (RMSE) and reduce significantly computational time required by current practice. ACO_R and $DACO_R$ perform favourably well in all catchments, providing good accuracy on average (RMSE) and high consistency within the results for streamflow (the Dandalups and Bates) and streamflow and salt generation-focused HM (Lewis). ROPE has comparable performance with ant-inspired optimisation methods when the focus of HM is based on only streamflow or salt generation. By contrast, GLM has good performance when HM focuses on both responses. GLM is the most efficient method in terms of computational time. On the contrary, ROPE is computational intensive method. Meanwhile, the computational time required by both ant-inspired optimisation methods can be categorised as low to moderate level. All HM methods are able to reduce the initial uncertainty of input parameters by more than 45% for all catchments.

Sensitivity analysis test reveals that the larger training data size used for developing surrogate models, the higher accuracy on average (RMSE) and the higher consistency within results is obtained. However, the larger training data size requires the longer computational time. Among proxy models, the development of multidimensional Kriging is computationally extensive and ANN-Rprop with one hidden layer has poor performance on average (RMSE). Multidimensional Kriging, ANN-backprop, ANN-Rprop with two and three hidden layers are selected for developing proxy models in HM. Except for multidimensional Kriging developed from a training data size of 100, all proxy models show comparable performance in terms of average RMSE and consistency with the results provided by direct HM. Moreover, indirect HM provide a significant reduction in computational time compared to direct HM, especially when HM is implemented using multiple solution-based methods. ACO_R, $DACO_R$ and ROPE in indirect HM are able to reduce computational time by more than 80% used by these HM methods in direct HM. Meanwhile, for a single deterministic approach, GLM, indirect HM also shows a reduction of computational time by at least 16%.

In conclusion, ACO_R and $DACO_R$ can be promising tools as alternative HM methods in hydrology. The implementation of proxy models in HM has provided a significant reduction in computational time with comparable accuracy. Therefore, it will be advantageous to apply them in future HM applications.

List of publications

Conferences and committee were attended during the PhD candidature:

- D. Tjia, R. Gupta, and M. Muhammad, "Comparison of history matching methods for calibration of hydrology model," *In International Conference on Engineering and Applied Sciences Optimization-OPT-i, Greece, June 2014.*
- D. Tjia, R. Gupta, and M. Muhammad, "Ants do history matching," *In Ozwater, Adelaide, 12-14 May 2015.*
- D. Tjia, R. Gupta, M. Muhammad, and T. Stokes, "Application of automatic history matching methods in hydrology model," *Presented to Bauxite Hydrology Committee, Perth 12 June 2015.*
- D. Tjia, R. Gupta, and M. Muhammad, "R plays with ants," *Presented in Rmeetup Perth, 24 March 2016.*
- D. Tjia, R. Gupta, and M. Muhammad, "An alternative ant colony optimization (ACO_R) for calibration of hydrology model," *Hydrology Research.* (under preparation).

Acknowledgements

"Dewi, why don't you try? You have got good background, I can see you are capable of doing this?"

This encouragement was said by Ritu Gupta, my supervisor when I took her class five years ago. Since nobody trusted me and I felt no confidence of doing research. Ritu approached and inspired me to apply for the scholarship. I got it! She guided through the research, helped to pass the barriers, brought me to the end of this study and proved it to me, yes! I am capable. Thank you, Ritu from my deep heart for providing opportunity, inspiration, encouragement, guidance and helps from the beginning up to the end of this study.

I would like to express my heartfelt thank you to Muhammad Shafiqul Alam, my associate supervisor from Department of Water in providing LUCICAT software and data, and taught how to operate it, providing guidance and great help throughout this study. I also would like to thank you for providing me the opportunity to do an internship in Department of Water five years ago. This gave me very good knowledge before I started my PhD studies.

A special thank you to Tim Sparks for giving permission to study in the Department of Water, and the opportunity for me to use the LUCICAT software and data in implementing statistical methods for HM.

Thank you to Todd Stokes and Andrew Briggs from ALCOA for providing the Bates and Lewis data for this study.

I would like to extend my appreciation for my editor, Eddy for careful editing my thesis and also, for Casey who helped the improvement of my thesis.

The most important thanks go to my husband, Dony, for his endless help, support and encouragement. Thank you for your love, your patience and your understanding. Without you beside me, I would not have been able to finish this work.

I am grateful for the never-ending love and encouragement from my parent, Lily, my brothers and my sisters, my sister and brother-in-law, Nensi and Lukman.

I am greatly indebted to my best friend, Edison who provided great help in the process of writing code to implement the algorithms used, and discussions on some material that required deep understanding during this study.

Finally, I would like to thank you for all my friends that provided technical and moral

support to me during the hard times of this study, especially Francisca, Pauline, Juan, Lishuang, Raja and Shican (Room 444 and 445).

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1 Is Australia Experiencing Water Crisis?

Water is essential for human life. Without food, a human can survive up to three weeks [26]. However, without water a human cannot live beyond a week [26]. Water on the Earth is estimated to be 1.386×10^{15} Megalitres [172]. Only 3% of this water is available as freshwater for human consumption [172]. The remaining water comprises salt water, frozen ice or glaciers. Freshwater can be obtained from surface flow, such as rivers and reservoirs or groundwater. Freshwater availability in future is in doubt due to population growth, economic development, urbanization and climate change. Therefore, there is a need to manage water resources wisely.

Australia is the driest inhabited continent on the Earth. Managing water resources is very crucial to assure water resource availability in the future. The following sections present the factors contributing to water crisis in Australia and the significance of this study in attaining sustainable water management.

1.1.1 Factors Influencing Water Crisis in Australia

Australia is the driest inhabited continent, with the greatest rainfall variability on Earth. Out of the 100 years between 1895 and 1995, 39 years have been classified as serious drought years in some parts of Australia. In the twelve years beginning in 1995 many areas in the central and southern mainland have experienced seven to ten years of severe drought [182]. Furthermore, rivers in Australia have high fluctuation in annual flows between different extreme seasons. For comparison, the ratio between the maximum and minimum annual flows in the Murray and Hunter Rivers in Australia are 15.5 and 54.3 respectively, while for the Rhine River in Switzerland the ratio is 1.9 and for the Potomac River in the USA the ratio is 3.9 [27].

In addition to being a very dry continent with highly variable rainfall, Australia has recently experienced rainfall decline due to climate change [25, 45, 47]. In the past few

decades rainfall has declined in the southwest and southeast of the continent. The southwest of Australia has experienced a 17% decline in average winter rainfall since 1970 [25]. Meanwhile, the southeast of the continent has had a 25% rainfall decline between April and May since the mid-1990s [25]. Reduction in rainfall has had a significant impact on streamflow generation. In the far southwest, streamflow has reduced by more than 50% since the mid-1970s and in the far southeast, the streamflow decline has been around half the long-term average between 1997-2009 [25]. The reduction of streamflow has affected the volume of water stored in dams. Australia has nearly 500 large dams with a total storage capacity of almost 84,000 GL. In 2004 only 44,000 GL was stored in dams across Australia, and fell below 40,000 GL in 2005 [182].

Moreover, the Australian population has significantly increased in the past ten years, as depicted in Figure 1.1. In 2006 the population size was just above 20 million and by year 2014 this figure was nearly 23.5 million [177]. The rate of population growth fluctuated between 0.6% (2006) and 2.1% (2014) per year [177]. Current estimated population (17 February 2016) is 24 million [22] and forecast to be 46 million by the end of 2075 [20].

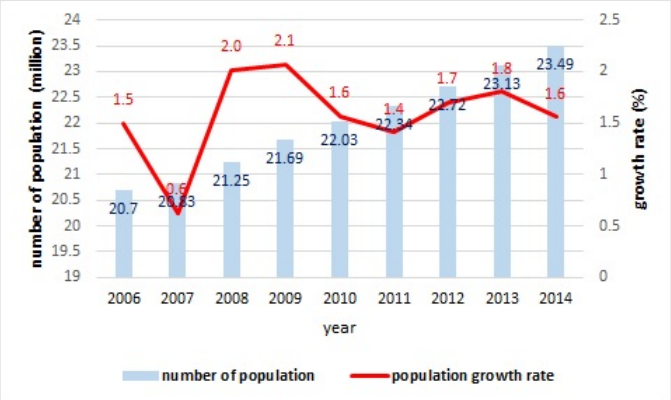

Figure 1.1: Population growth between 2006 and 2014, source: [177]

Population growth exerts pressure on water resources. Increasing population leads to increased demand for water supply, food production and manufacturing. The Australian Bureau of Statistics (ABS) [21] reported that most states have increased water consumption between 1% and 28% from 2011/2012 to 2012/2013, except Western Australia where water consumption has decreased by 10% due to water restrictions applied by the Water Corporation. Of the total water consumption, agriculture has dominated water use at nearly 65%, domestic/household and manufacturing industries have consumed 11% and 3% respectively [19]. In addition, an increased demand for food drives expansion of agricultural land leading to increased water usage. This also requires extensive land clearing which leads to increased salinity of land and water resources.

1.1.2 Stream and Dryland Salinity

The Australian landscape is prone to deposition of large amount of salt on land. These salts came from the ocean, evaporated into the atmosphere and falling on the ground surface within rainfall and dust. The quantity of salts precipitated annually was from 100 to 200 kg/ha around the coast area. This quantity gradually decreased to 10 and 20 kg/ha in the inland area [56]. Over long processes of evaporation, transpiration and weathering, saltfall has infiltrated and percolated to the soil profile and deposited in unsaturated and saturated zones. The movement of salts in shallow groundwater is relatively high, resulting in low concentration of salts. By contrast, in deep groundwater, a high concentration of salt is observed due to slow mobilization of salt [144].

Extensive land clearing for agriculture during 1940-1970 has caused an increase in streamflow and salinity within Western Australian catchments. The rise of streamflow comprises two stages: first, the reduction of evapotranspiration immediately after land clearing, and, secondly, the expansion of the saturated stream zone induced by groundwater. When the groundwater levels increase it brings the salt deposit from surrounding soil with it to the land surface, creating salinity in the land and when the groundwater level reaches the stream base, stream salinity is generated.

Stream and dry land salinity are key environmental problems affecting Australia. Salinity reduces the productivity of plants, affects the road and rail network and threatens the availability of drinking water sources. According to National Land and Water Resources Audit (2001) [133], the estimated high potential of dryland salinity is 5.7 million ha and will increase to 17 million ha by 2050. Of all the States, Western Australia has the greatest salt-affected land area, contributing 80% towards the national salinity problem [133].

ABS [18] reported there were 20,000 farms or about 2 million hectares of land showing actual signs of salinity in Australia. Of this land about 42% is unfit for production [18]. It was recorded that 6,918 farms or almost 1.2 million hectares of agricultural area in Western Australia by affected by salinity, and 45.7% of the affected area was unproductive [18].

Figure 1.2 and Figure 1.3 are examples of formation of salt in one creek of South-Western Australia and in the area adjacent of a river in South Australia respectively.

1.1.3 How to Overcome this Water Crisis?

Groups from the hydrology community, such as the Bauxite Hydrology Committee and the Hydrology Water Resources Panel WA comprising hydrologists and government agencies have worked together to provide and implement a sustainable water resources plan. These bodies rely on information from hydrological models of the characteristics water reservoirs, catchments and stream flow. A reliable hydrological model can be used to understand

Figure 1.2: Doradine Creek (2003), a in the Dumbleyung Catchment in South-Western Australia. White salt on the banks and flows are formed from deep drains upstream intersecting saline groundwater. (Source: [46])

Figure 1.3: Dead trees and salinity at a floodplain along the River Murray, South Australia (Source: [31])

characteristics and behaviour of a catchment, and be used for prediction of water resources volumes. This modelling builds the base of knowledge in planning and managing water resources.

Hydrological models have been widely used to mimic natural catchment runoff processes since the early 1970s. Hydrological models comprise of complex mathematical equations that model the geological and hydrological characteristics of a catchment, including lateral hydraulic conductivity (K_{ll}) and vertical conductivity (K_{uv}). For a set of inputs of climatic data and catchment characteristics, hydrological models compute streamflow and/or total salt. Precise input parameter values of the hydrological models are not known and typically only uncertainty ranges of values are available based on geological studies of a catchment.

Hydrological models are sensitive to the input parameter values. In practice for a hydrological model to give reliable estimates of stream flow, the precise values of the input parameters are required. These precise input values are obtained by calibrating the hydrology model to match observed streamflow and/or total salt. The iterative processes of tuning the input parameters to minimize the differences between output response and historical data is known as history matching (HM). A successful implementation of the HM process provides a new set of reduced input parameter ranges. With these new set of input values, a hydrological model provides a more reliable range of future predicted streamflow and/or total salt. These estimates are vital for the hydrology community to contribute to the development of sustainable water resources management. Figure 1.4 depict the process of HM and the results of HM for future prediction.

Nation-wide programs and actions have been implemented to conserve water resources and reduce salinity [18]. By implementing water reform planning guided by reliable estimates of resources, the water crisis problems in Australia can be managed.

1.2 Who will Benefit from this Thesis?

HM is a challenging task as hydrological models are often nonlinear and computationally complex. Hydrologists commonly require substantial computational time to estimate good model parameters using current techniques.

This thesis is aimed at providing effective and time efficient HM methods. The results of this study will benefit the hydrology community, especially hydrologists and/or government agencies who want to develop a reliable hydrology model efficiently and effectively for water resource planning. Indirectly, the outcome of this research would benefit people in the community by having continuous water supply. We hope that the proposed HM methods will enhance the performance of current history matching practice. Reduction of computational time in HM with high accuracy and low uncertainty of model parame-

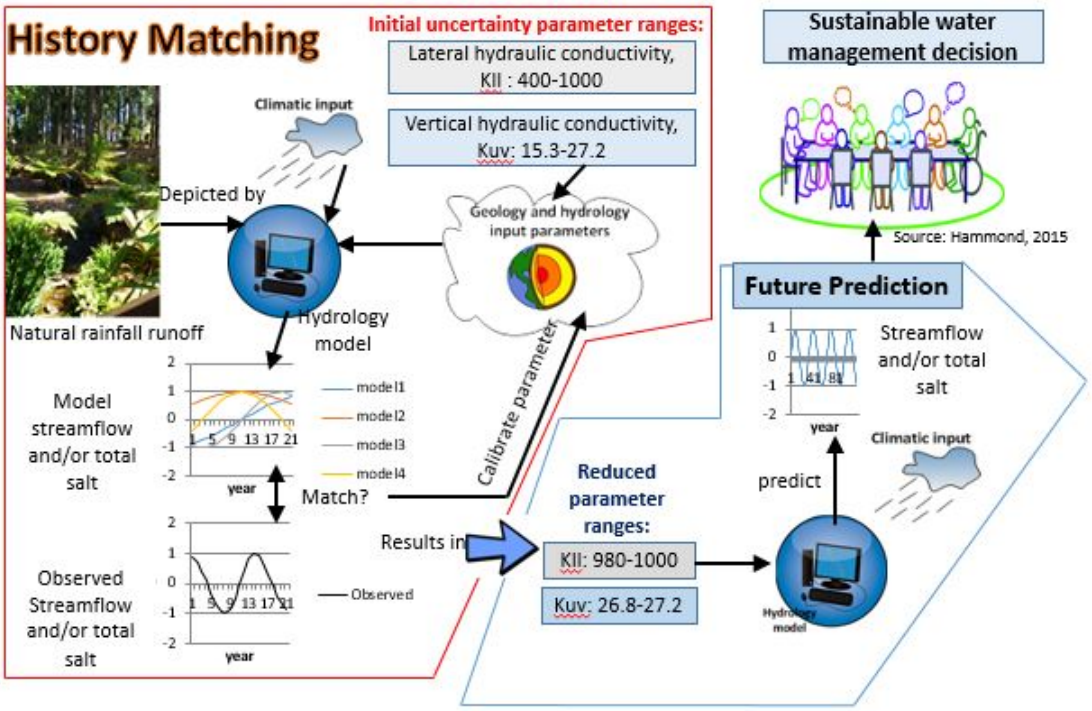

Figure 1.4: HM process

ter ranges are the main findings of this research. Although the proposed techniques are applied for HM in a RRM, these techniques can also be used for other purposes, such as optimisation of water reservoir operation or groundwater monitoring and other optimisation problems in different fields, especially in Exploration and Production industry. The HM methods presented here can be directly used for reserves prediction given available production history.

1.3 Objective of this Research

In this research, we aim to:

(i) Investigate the performances of a range of HM techniques in estimation of hydrological model parameters in terms of accuracy, consistency within results, computational time and reduction of uncertainties in initial input parameters. The approaches presented in this study are Gauss Levenberg Marquardt (GLM), Ant colony optimization on continuous domain (ACO_R and Diversity ACO_R ($DACO_R$)) and robust parameter estimation (ROPE). The first of these methods is based on a single population approach, whereas the remaining techniques are multiple solution-based methods.

(ii) Understand the effect of tuning parameters on the performance of HM.

(iii) Understand the robustness of model parameters obtained from HM methods in different time period.

(iv) Perform HM using multiple criteria.

(v) Develop the process of indirect HM, where the hydrological model is replaced by a surrogate model.

(vi) Investigate the performances of various types of surrogate models in HM. We consider two surrogate modelling methods, namely multidimensional Kriging and artificial neural networks (ANN) for this study.

(vii) Understand the effect of the size of training data and surrogate model complexity on the performances and computational time of surrogate models and performance of HM.

1.4 Structure of the thesis

The thesis is organized as follows:

Chapter 2 presents an overview of HM methods. Here the framework of HM methodology is presented along with information on the HM methods (ACO_R, $DACO_R$, GLM and ROPE) used in the process of HM. Next the indirect HM method is presented, where surrogate models are developed to replace the hydrological model. In this chapter, the theory of HM methods and surrogate models is presented.

Chapter 3 reviews the catchments and hydrological model explored in this study. We use three study areas, namely the Dandalups, Bates and Lewis catchments, all located in Western Australia. We apply the Land Use Change Incorporated CATchment model (LUCICAT) to mimic the natural rainfall runoff system within these catchments.Details of this model are presented here.

Chapter 4 focuses on direct HM (using the LUCICAT model) performed by HM methods. In this chapter, the performances of HM methods is evaluated in terms of the accuracy via root mean square error (RMSE), consistency/stability results, computational time and reduction of initial input parameter uncertainties. Key findings regarding the parameters are presented.

Chapter 5 presents indirect HM using surrogate models. The HM is applied using all HM methods (GLM, ACO_R, $DACO_R$ and ROPE). The focus of this chapter is to calibrate model parameters for the Dandalups based on 33 annual daily peaks of streamflow. The performance of surrogate models as well as HM methods is evaluated in terms of accuracy and computational time. The performance of surrogate models is presented.

Chapter 6 presents the summary of research, major contributions and key conclusions. Recommendations for future work are also provided in this chapter.

CHAPTER 2

Literature Review

2.1 Overview

History matching (HM) methods play important roles in various computer-based modelling. The presence of HM methods reduces significantly the time and effort dedicated by scientists to improve the accuracy of computer-based modelling. Since their advents, HM methods have evolved from providing single deterministic approaches to multiple solution-based methods. Both methods have been widely implemented in hydrology fields for calibrating input parameters of rainfall-runoff models (RRMs). Three HM approaches of both classes, namely ACO_R and Diversity ACO_R ($DACO_R$), Gauss Levenberg Marquardt (GLM), and robust parameter estimation (ROPE) will be highlighted in this study. The first two methods are inspired by the foraging behaviour of ants, GLM is a gradient based approach and ROPE is an algorithm using the concept of data depth, i.e. degree of centrality of a point with respect to a set of data. Ant colony optimisation and ROPE are categorised as multiple solution-based methods, while GLM is a single solution approach. These methods have shown successful application in many science and engineering fields.

The application of these methods can be performed via either direct and indirect HM. In direct HM, all methods are used directly with the computer-based modelling. However, in indirect HM these approaches are combined with a surrogate model for the computer-based modelling. Two surrogate models will be focused on are multidimensional Kriging and artificial neural network.

This chapter is divided into four main sections. Firstly, we present the position of HM methods in computational modelling and the evolution of HM methods. Secondly, a description of the direct HM approach is provided as well as some applications of direct HM in hydrology. In the next section we present the theory of HM methods in this study and their application to HM problems. Finally, the background of indirect HM methods and several applications of indirect HM approach to RRMs are presented.

The objectives of this chapter are:

(i) To present the importance of HM methods in computational modelling.

(ii) To review the evolution of HM methods.

(iii) To introduce the direct HM procedure, and the challenges of its use in practice.

(iv) To review the HM methods (ACO_R, $DACO_R$, GLM and ROPE) for this study and the reason these methods were chosen.

(v) To introduce the indirect HM procedure and review the surrogate model (multidimensional Kriging, artificial neural network) for indirect HM.

2.2 History Matching Methods in Computational Modelling

Computational modelling is a collection of computational codes implementing mathematical equations describing complex natural physical processes. The development of computational modelling in various fields has progressed dramatically since the emergence of more powerful computers, compared to the difficulties of studying natural phenomenon via physical experiments. The input parameters of a computer-based modelling need to be calibrated in order to develop a "reliable" model. Reliable in this context means that a model consistently provides good performance for a series of time points. It may then be used for projecting a response, studying a phenomenon, or intervention of interest. The process of input parameter adjustment by matching model output and observed response is known as history matching (HM).

Commonly, HM is performed by manual or by computational methods. In manual work, one or a few input parameters are adjusted manually to improve the match. Such an approach largely depends on expert judgement and experience of the practitioner, and available budget [150]. Manual HM is very tedious and time consuming [48, 70]. The presence of HM methods since the early 1960s has helped scientists and engineers make significant improvements in computer-based modelling. HM methods start with an initial set of input parameters, and this set is recursively updated by optimising an objective function representing the differences between the observed and modelled response.

HM methods have been widely applied in many areas of science from economics and engineering. In economics, Semmler and Gong [189] introduced a maximum likelihood procedure combined with simulated annealing for parameter estimation of a business cycle model for US macroeconomic time series data. Blueschke et al. [38] proposed an established algorithm (OPTCON) [65] and differential evolution (DE) [155] for parameter estimation for three macroeconomics models. HM has also been applied to a telecommunication model. Venkatesan and Kumar [156] performed a comparison study using some HM methods to estimate the parameters of the Bass model [58], i.e. a model which can

be used for predicting future sales per period, the magnitude of peak sales and the time to maturity during the growth phase of a new product life cycle. Sun *et al.* [190] proposed the Cuckoo Search to optimise a Markov Chain Grey Model parameters, i.e the model which can effectively capture the randomness of uncertain systems with limited sample of data, for forecasting the variability of international tourists in China.

In medicine, simulated annealing and a genetic algorithm were used to estimate parameters of micro-simulation model of lung cancer development, progression, detection, treatment, and survival [37]. The Cubature Kalman Filter method [75] was applied to calibrate parameters of a neural mass model, which was developed to study the mechanism underlying certain brain activities or phenomena that lead to status epilepticus [17]. Liu *et al.* [103] proposed the artificial immune network algorithm to model parameter adjustment of tracer kinetic modelling in positron-emission tomography (PET). That is, a quantitative method to measure physiological and biochemical processes by detecting radioactivity of a radioactive tracer injected into a peripheral vein. In medical science, ACO has been used to optimise structured based drug design [135].

In biotechnology, Chen and Campanella [60] applied gradient descent to estimate microbial survival parameters of a kinetic model, used for determining microbial inactivation in thermal processing of food. Lunelli *et al.* [24] used a genetic algorithm for parameter estimation of a kinetic model of lactic acid production. The firefly algorithm was also applied to search for optimal parameters of the osmotic dehydration model in food preservation [147]. A Tabu search was applied for calibration of a mechanistic growth model for tomato seedlings in unheated beds [130].

In engineering, HM methods have been widely applied to estimate parameters of energy recovery models: the shuffled complex evolution method used for parameter estimation of a pyrolysis kinetic reaction model [198]; a genetic algorithm used for searching optimal parameters in a microbial fuel cell model in dairy wastewater treatment [153]; chaos particle swarm optimisation for estimating the parameters of a photovoltaic/solar model [73]. The use of HM methods in petroleum engineering is very popular as the reservoir model is very complex, requiring extensive computational time. Rwechungura [150] stated the advantages using HM methods in reservoir modelling and studied the performance of gradient and non-gradient algorithms for reservoir parameter estimation. The ensemble Kalman filter has been recently used to estimate reservoir parameters and quantify the uncertainty of future reservoir performance [49, 199]. Peters et al. [105] compared the results of various HM methods (ensemble-based methods, global optimisation methods and gradient-based methods) for several reservoir models. Baskan and Haldenbilen [134] implemented the ant colony algorithm (ACO) to optimise traffic signal timing. Lung [30] applied ACO in green manufacturing to provide an effective dis-assembly process before discarded products were reused, re-manufactured or disposed of. Chico [61] optimised

electrical distribution using ACO. Pederson *et al.* [169] introduced ACO in order to track and interpret faults from 3D seismic data in petroleum engineering. Byuksaatci [161] applied the bat algorithm (BA) for the single row facility layout problem in manufacturing systems to improve the efficiency of operations and reduce total operating expenses. Yang *et al.* [197] used BA in topology optimisation to find the best geometrical shape that minimises the amount of materials used in microelectronic optimisation. The CS algorithm was also used in an inverse heat transfer problem [179].

The application of HM algorithms in water resources has been widely reported. Zeng and Wang [32] introduced the Tabu search to identify parameter structure in groundwater modelling. Xie and Zhang [191] applied the ensemble Kalman filter to improve a RRM reliability and reduce predictive uncertainties. Todini [53] reviewed that Bayesian method can derive the probability density function for the hydrological parameters as well as quantify the predictive uncertainties of predictand (river stage, discharge or runoff volume). Krausse and Cullmann [173] proposed ROPE approach using concept of data depth (degree of a point based on its centrality of a data set) for parameter calibration of a RRM.

Many applications of HM methods in other fields, such as mathematics, statistics, computer science, physics, and chemistry that have not been mentioned here. Readers who are interested may access the public domain sources. We illustrate the application of HM methods in the world of computer-based models in Figure 2.1. HM methods are used for model parameter estimation, and the fitted model may then be used for prediction of model response to study a natural behaviour or the impact of a treatment for example.

2.3 The Evolution of History Matching Methods

HM methods have been widely applied in computer-based modelling since the 1960s. At the advent of HM methods the focus involved numerous mathematical methods directed to finding a single local solution. However, since 1970 the search strategy of HM methods has shifted to providing a set of solutions. These later methods, also known as multiple solution-based methods, involve stochastic rules to locate a global solution. We briefly describe below some of these methods and their modelling applications. An overview of the evolution of HM methods is shown in Figure 2.2 below.

Alternating-Direction Method

The first implementation of HM methods in finding a single solution in oil and gas field exploration was introduced by Kruger [183]. He used the alternating-direction implicit method to adjust the basic absolute permeability data to match past reservoir performance in flooding (water injection) or cycling (gas injection) project. This project aims to

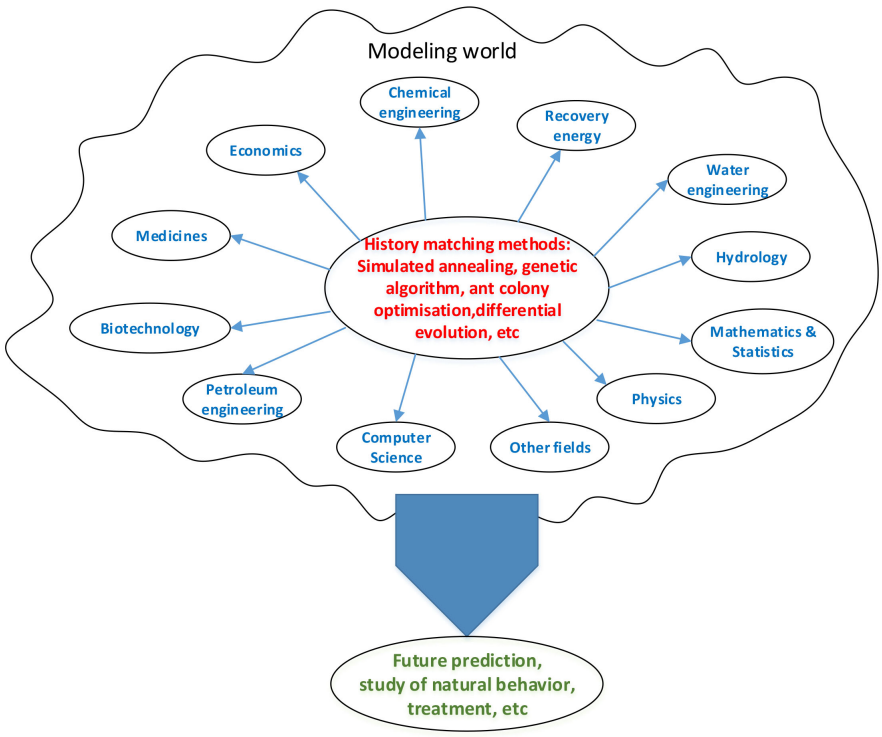

Figure 2.1: Position of HM methods in computer-based modelling

enhance oil recovery.

Least Squares

In 1968, Coat et al. [93] proposed the least squares and linear programming methods, which required less CS runs than Kruger's study, to determine reservoir input parameters in single and multiphase flow problems.

Gradient Method

This approach is another single solution-aimed technique using opposite direction of gradient of objective function to locate local minimum. The method starts with a set of initial parameter values and these parameters are iteratively updated toward a set of parameters that minimise a function (error function). The process of updating parameters is performed by taking steps in the negative direction of gradient of a function (or approximated gradient of a function). This method was firstly applied by Dawdy and O'Donnell [41] for RRM parameter estimation and by Slater and Durer [64] to solve HM problems in the oil and gas industry.

Gauss-Newton (GN)

Thomas et al. [101] proposed GN method to overcome nonlinear problems in oil and gas reservoir model. The method is based upon minimising the error between observed and calculated performance data in terms of least-squares approach. Kitanidis and Lane

[138] used GN to find Maximum Likelihood Estimation (MLE) of spatially correlated hydrological parameters.

Gauss Levenberg Marquardt (GLM)
In oil modeling, Watson and Lee [15] presented another single solution method to solve a nonlinear problem of oil production data using a dual-porosity model.

In water resources, GLM was applied for estimating model parameters of unsaturated soil hydraulic properties of transient flow [86]. GLM in hydrology has become widely used since a free package of software, Parameter Estimation (PEST), was introduced by Doherty [79].

Bayesian framework
A Bayesian parametric method, was firstly adopted by Gavalas *et al.* [66] for solving HM problem in oil model. This approach was successfully implemented to calibrate a modest number of input parameters (approximately 100 input parameters [171]).

Generalised Likelihood Uncertainty Estimation (GLUE)
GLUE was introduced by Beven and Binley in 1992 [96] in modelling of the rainfall-runoff transformation. This methods applies informal likelihood function to find a set of representation (model input, model error) that are behavioural within acceptable threshold. A large number of input parameter combinations are generated using a prior distribution. Each combination is then assigned with a likelihood value, a value is calculated by comparing the model outputs and observations. Lower values indicate bigger deviation between model outputs and observations. With a threshold, these combinations are split into behavioural and non-behavioural sets. The behavioural sets are eventually used for generating cumulative distribution function of output prediction.

Tabu search (TS)
Tabu Search (TS) is a meta-heuristic approach developed by Glover [54]. TS moves from a current solution in the feasible space to the best solution in its neighbourhood. Tabu applies two rules, namely at each step the algorithm accepts worse moves if there is no improvement available and prohibits repetition of the same solutions (tabu).

Simulated Annealing (SA)
The SA algorithm was introduced by Ouenes [12] for an HM problem in the oil and gas industry. This algorithm is inspired by the annealing process in metallurgy. In SA, a random solution is constructed and accepted as the new solution based on a probability, which is dependent on the difference of corresponding function values and a control parameter T (temperature). T decreases gradually as the solution approaches to the

minimum. The successful outcome of SA was also demonstrated by Summer et al. [132] in order to find the optimal values of seven sensitive parameters of a RRM.

Genetic Algorithm (GA)

GA belongs to the class of evolutionary algorithm that entails random solution space searching. This algorithm is inspired by the Darwinian theory of natural selection. GA maintains the multiple solution with N individuals on which each individual holds two properties: a location or chromosome containing genes and quality (fitness value). Information on good individuals will be passed to the next generation to develop better individuals through gene exchange between two or more good individuals (crossover), and mutation (small changes happening in its own individuals). GA was first introduced by Wang [142] in hydrology. Emerick et al. [7] and Nogueira and Schiozer [139] proposed GA in the oil and gas industry.

Evolutionary Strategies (ES)

Another variant of GA is ES. This approach maintains N individuals and performs two genetic operations in developing the next generation. Crossover in ES is called intermediate crossover, meaning two random selected chromosomes of good individuals (parents) are averaged to generate a new individual. Every new individual is then mutated with a normal distribution disturbance. N individuals finally are selected from the population of the new generations and parents based on their best fitness values. The application of ES in oil and gas HM problems has been studied by Schulze-Riegert et al. [152], Selberg, et al. and [171].

Ant Colony Optimisation (ACO)

ACO algorithm is one of the swarm intelligent methods inspired by the study of ant colony foraging behaviour to find the shortest path between food and their nest. ACO was initially proposed by Dorigo [113] to solve discrete problem such as "the travelling salesman" problem [114].

The first application of ACO in the water industry was introduced by Abbaspour et al. [92] in 2001 to search the optimal parameters of an unsaturated soil hydraulic model. ACO was applied by Li and Hilton [202, 203] for groundwater monitoring. Maier et al. [72] applied ACO to design a water distribution network. ACO was also applied in improving the efficiency and resiliency in routing of pipe networks [162]. A hybrid of ACO and GA was reported by Li et al. [201] to maximize cost effectiveness of sampling in groundwater monitoring. Li et al. [168] also proposed a hybrid of ACO and SA for groundwater parameter estimation.

This method has recently been developed to deal with continuous optimisation problems. Razavi and Jalali-Farahani [100] used Continuous ACO (CACO) for parameter

estimation in an HM problem in oil exploration, to determine the optimum number of phase separators and separator pressure to maximise cumulative oil production. Another continuous ACO, known as ACO_R, was introduced by Socha and Dorigo [99]. Hajizadeh et al. [200] applied ACO_R to solve simple and complex high dimensional optimisation problems in reservoir models, and compared the results with the neighbourhood algorithm. Their study showed that the convergence of ACO_R toward the good input parameter regions was consistent from different initial settings and outperformed the neighbourhood algorithm in terms of providing better model fits and time efficiency. In electrical engineering, Simsek and Mergen applied ACO_R [57] for minimising multiple objective functions in induction motor design. Dariane and Moradi [2] and Ananda Babu [91] proposed modified ACO_R to improve the accuracy and time efficiency in reservoir operations for agriculture and domestic needs. Moreover, ACO_R was applied to calibrate model parameters of a RRM for a Western Australian catchment [14]. Leguizamon and Coello Coello [63] proposed diversity ACO_R ($DACO_R$) in handling continuous and large dimensional problems.

Shuffled Complex Evolution (SCE)

The SCE method was proposed by Duan [141]. The algorithm starts by randomly generating a set of input parameters and the quality of each set is evaluated. The input parameters are then split into several subgroups, known as complexes. Each group of input parameters has a similar quality of solution. The solutions within each group are improved using the simplex algorithm before they are shuffled into a new population and new complexes are formed. SCE has been widely applied to the calibration of RRM parameters [78, 87, 160].

Particle Swarm Optimization (PSO)

Another swarm intelligent algorithm is particle swarm optimization (PSO) developed Eberhart and Kennedy in 1995 [83]. PSO is inspired by the social behaviour of flocking birds. In PSO the potential solutions, called particles, each have a fitness value. These particles are guided with velocities to fly through the input parameter space toward the optimal solutions.

In petroleum engineering, Mohamed et al. [104] applied PSO for HM and compared the efficiency of PSO with the Neighbourhood Algorithm (NA) and Hamiltonian Monte Carlo (HMC). Their study demonstrated that PSO performed well, using fewer simulations for their particular model. PSO was also implemented for seismic history matching and production optimization by Fernandez et al. [81]. In RRM, Khoi and Thom [40] applied PSO to analyze parameter uncertainty analysis of streamflow in one catchment in Vietnam.

Robust Parameter Estimation (ROPE)
The ROPE method was firstly proposed by Bardossy and Singh [4] to address parameter and uncertainty estimation problem. This method is a nonparametric approach using the concept of data depth. Data depth is a statistical method to measure the centrality (depth) of a point with respect to multivariate data sets. The application of ROPE for HM in RRM was explored by Bardossy and Singh [4], Cullmann et al. [78], and Krausse and Cullmann [173].

Firefly Algorithm (FA)
FA is a swarm intelligent method proposed by Yang [192]. This algorithm is inspired by the flashing light produced by fireflies to attract mating partners and to warn potential predators. The FA uses a physical formula where light intensity decreases with the square of distance.

Cuckoo Search (CS)
CS was developed by Yang and Deb [195]. The CS was inspired by brood parasitism of some cuckoo species. In brood parasitism, female cuckoos lay their eggs in the nests of other hosts and remove one of the eggs of a host bird. As a result the young cuckoo is nurtured by a host species. In a few cases a host identifies an alien egg and either throws it from the nest or abandons the nest to build a new one.

Bat Algorithm (BA)
Another novel algorithm, developed by Yang [193], the BA adopted the echolocation features of Microbats. The BA uses a frequency-tuning technique to improve diversity of solutions in a population and applies an automatic zooming technique using the variations of pulse emission rates and loudness of bats to balance exploration and exploitation in the search.

Multiple Objective Optimisation algorithm
Many HM problems in engineering field involve multiple objectives. For instance, RRMs generally have multiple responses, such as groundwater level/elevation, river discharge and salinity, high and low streamflow, multiple data location. Researchers have proposed some approaches to handle HM with multiple objective problems. One of them is by aggregating multiple objective into one objective function, applying weight for each objective. Then, utilising algorithms previously discussed to calibrate model parameters. In the late 1990s, researchers have introduced the other approach, known as multi-objective algorithms. In multi-objectve approach, previous algorithms are modified to search pareto optimal front. For minimisation problem, pareto optimal sets are obtained by searching non-dominated vectors. For instance, considering two solutions vectors, $u = (u_1, ..., u_n)$ and $v = (v_1, ..., v_n)$, u is said to dominate v if only if there is no component of u is larger

than the corresponding component of v and at least one component is smaller. Detail can be read in Yang [194]. Among HM methods, multiple solution-based search methods have been demonstrated to be powerful search approaches for solving multi-objective optimisation problems. Boyle et al. [48] applied multi-objective complex evolution (MOCOM-UA) in hydrological modelling. Deb et al. [94] introduced the non-dominated sorting genetic algorithm II (NSGA-II). The Multi-objective PSO (MOPSO) algorithm was proposed by Gill et al. [118] to estimate a RRM model's parameters. Recently, Yang [194] and Yang and Deb [196] have introduced the multi-objective firefly (MOFA) and multi-objective cuckoo search (MOCS) algorithms for solving design optimisation problem in engineering.

We present the evolution of HM methods (some of them) in Figure 2.2.

2.4 Direct History Matching Approach in RRM models

The HM process typically begins with construction of initial approximations of the RRM parameters. These approximations are then updated through a systematic process of reduction of an objective function measuring the discrepancy between observation and modelled response. In direct HM, the model response is provided directly from RRM output. In order to understand the direct HM process, we present the following workflow in Figure 2.3

As illustrated in Figure 2.3, a RRM model requires a set of input parameter values $\mathbf{x} = (x_1, x_2, \ldots, x_p)$ to generate the output response $\mathbf{y} = (y_1, y_2, \ldots, y_q)$. This can be represented as $\mathbf{y} = f(\mathbf{x})$ where the function $f(.)$ has unknown non-linear form, and is usually very complex. For a given setting of input parameters $\mathbf{x}^i = (x_1^i, x_2^i, ..., x_p^i)$ a single run of a RRM provides deterministic output values $\mathbf{y}^i = (y_1^i, y_2^i, ..., y_q^i)$. The input parameters for a RRM are adjusted to match the corresponding historically observed values of $\mathbf{y}^h = (y_1^h, y_2^h, ..., y_q^h)$. The HM process is iteratively performed to reduce the gap between the observed and modelled response until the gap obtained is less than a threshold δ.

Some applications of various HM methods of various RRMs for different catchments are summarised in Table 2.1.

In this study, we considered to apply four methods, namely GLM, ACO based on continuous domain, namely ACO_R and diversity ACO_R (DACO_R), and ROPE algorithms for history matching the RRM. This selection was based on the successful application of these HM methods to solve HM problems in various fields of computer-based modelling. Table 2.2 lists the comparison between the performances of the four methods and other

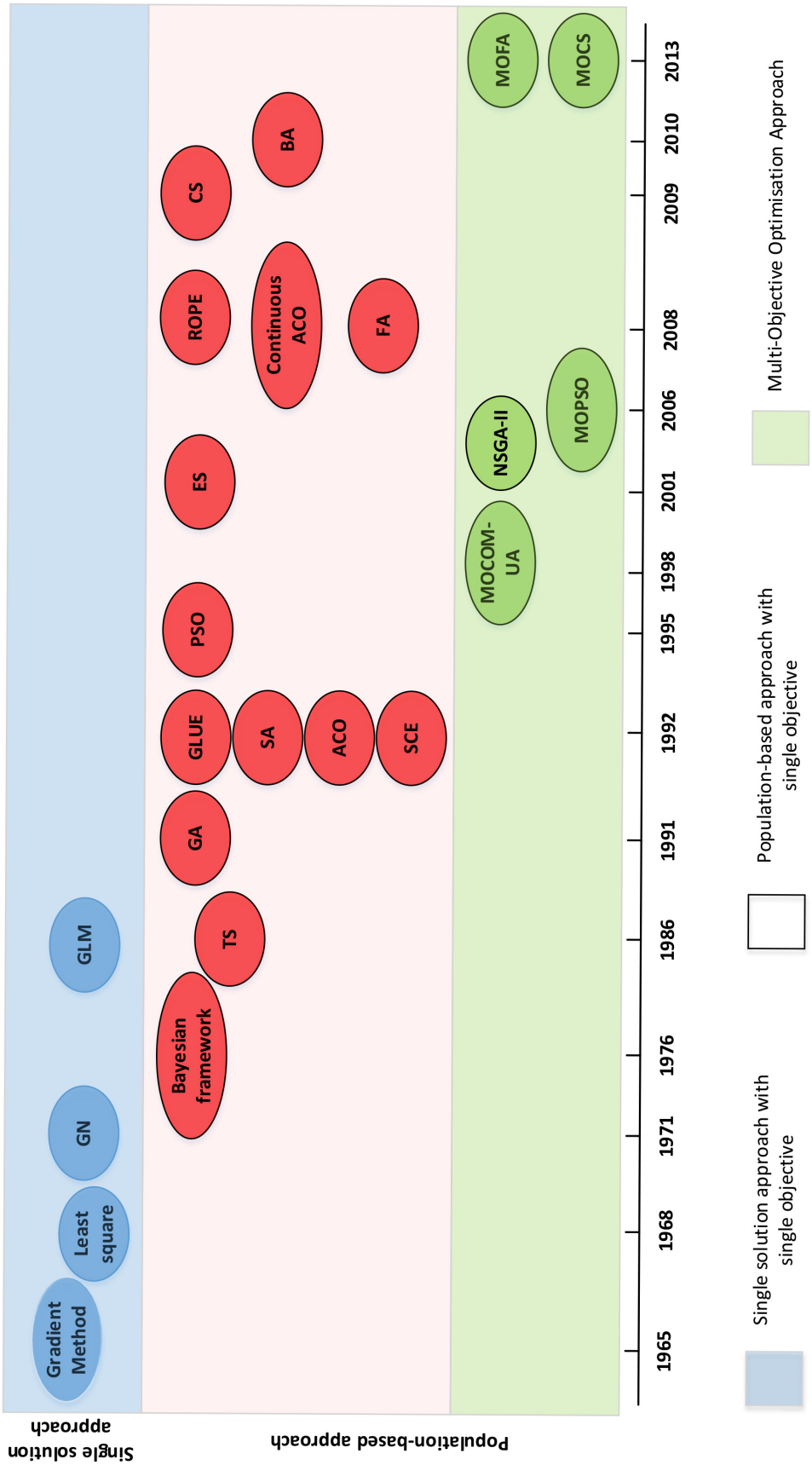

Figure 2.2: The evolution of HM methods

Table 2.1: The Application of Common HM Methods in Hydrological Modelling

Catchment	Model	Number Parameter	Response	Method	Reference
Karup River, Denmark	MIKE-SHE	5 & 11	GW elevation and Streamflow	GLM SCE	[160]
Wildcat Creek Kitsap County in Washington	HSPF	8	Streamflow	GLM	[23]
Margecany, Slovakia	Wetspa	11	Streamflow	GLM	[206] [124]
Rietholzbach Switzerland	WaSim-ETH	3	Streamflow	GLM ROPE	[78]
Trygevaelde Danish	MIKE 11/NAM	9	Streamflow	SCE	[69]
Bird Creek Oklahoma	Xianjian	7	Streamflow	GA	[142]
Sieve, Redesdale, Italy	ADM	11	Streamflow	different modified GA	[116]
Al-Kabeer, Al-Shimali, Syria	Soil Moisture Accounting	7	Streamflow	Combined GA & sequential simplex	[13]
Bay-Hang River, Taiwan	SAC-SMA	10	Streamflow	modified GA	[166]
22 cathments Australia	modified Boughton's SFB	6 & 7	Streamflow	combined SA & simplex	[132]
Neckar Germany	HBV	9	Streamflow	ROPE	[4]
Rietholzbach Switzerland	WaSim-ETH	3	Streamflow	ROPE A-ROPE ROPE & PSO	[173]
Leaf River Mississipi USA	SAC-SMA	13	Streamflow	PSO	[118]
Dandalup WA Australia	LUCICAT	5	Streamflow	ACO_R $DACO_R$ GLM	[14]

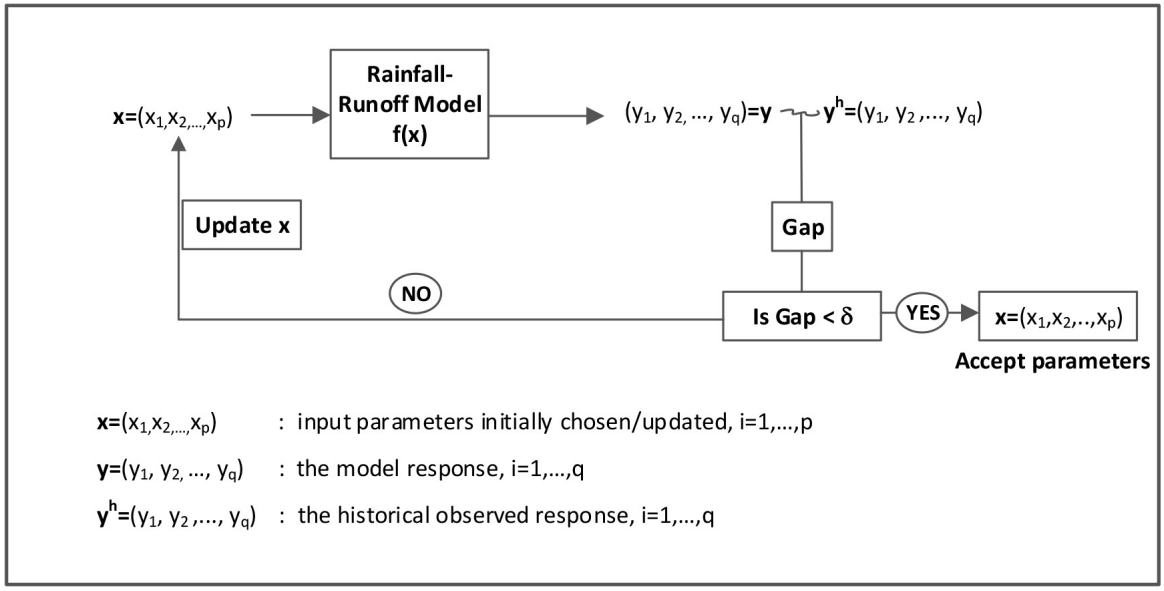

Figure 2.3: Workflow of the direct HM procedure for a RRM model

popular HM methods.

2.5 History Matching Methods for this Study

2.5.1 Gauss-Levenberg-Marquardt (GLM)

GLM is one of the robust deterministic single approaches for optimisation problems. This approach directs the search in the input parameters space by incorporating derivative information. GLM was introduced by Levenberg [97] and Marquardt [50] to solve nonlinear least squares problems. In least squares problems GLM minimises the sum of squared differences between the observed modelled outputs. GLM iteratively improves the current model parameter setting to provide the least squares solution. GLM uses two optimisation methods, the Gradient Descent and Gauss-Newton methods. As a gradient descent approach GLM updates the input parameters in the direction that results in the greatest reduction of the least squares function. While as the Gauss-Newton method, GLM finds the minimum of the quadratics by assuming the least squares objective is locally quadratic. When model parameters deviate far from the optimal solution, GLM behaves as gradient descent. By contrast, GLM changes to the Gauss-Newton technique when approaching the optimal solution.

Table 2.2: The Performances of ACO_R, GLM or ROPE Compared to Other Algorithms in Various Fields

Field	method	other method	description	Reference
Test functions	ACO_R	other ants continuous	ACO_R was higher performances	[99]
Test functions	ACO_R	GA,TS ES	ACO_R was comparable	[99]
Test functions	$DACO_R$	ACO_R	$DACO_R$ outperformed ACO_R, with less time	[63]
Reservoir operation	modified ACO_R	GA	ACO_R outperformed GA increase time efficiency	[2], [91]
Petroleum Engineering	ACO_R	NA*	ACO_R outperformed NA ACO_R has higher time efficiency than NA	[200]
Electrical Engineering	ACO_R	SA	ACO_R was comparable performances with SA	[57]
Hydrology	ACO_R	GLM $DACO_R$	ACO_R was comparable $DACO_R$ & GLM $DACO_R$ was the most consistent GLM was highly time efficient	[14]
Hydrology	GLM	NSGA-II	GLM was higher time efficiency	[124]
Artificial Neural Network	ACO_R	PSO & GA	ACO_R outperformed GA & PSO	[136]
Hydrology	ROPE	GLM	ROPE outperformed GLM in small and medium flood event GLM performed better than ROPE in large flood event GLM has high time efficiency	[78]
Hydrology	ROPE	GA & IPM**	ROPE slightly outperformed GA	[173]

Note:
* NA: neighbourhood algorithm
**IPM: interior point method

The least squares problem, $\chi^2(\mathbf{x})$ in GLM can be expressed as:

$$\begin{aligned}\chi^2(\mathbf{x}) &= \sum_{i=1}^{q}\left\{\frac{y_i^h - y_i(\mathbf{x})}{\sigma_i}\right\}^2 \\ &= (\mathbf{y}^h - \mathbf{y}(\mathbf{x}))^T\mathbf{W}(\mathbf{y}^h - \mathbf{y}(\mathbf{x})) \\ &= (\mathbf{y}^h)^T\mathbf{W}\mathbf{y}^h - 2(\mathbf{y}^h)^T\mathbf{W}\mathbf{y}(\mathbf{x}) + \mathbf{y}(\mathbf{x})^T\mathbf{W}\mathbf{y}(\mathbf{x})^T,\end{aligned} \tag{2.1}$$

where y_i^h is the observed data at $t = i$ with $t = 1, \ldots, q$. y_i is the modelled response at $t = i$, given model parameter vector $\mathbf{x} = (x_1, \ldots, x_p)$. The difference $y_i^h - y_i$ is weighted by σ_i, the error measure in the observed data at $t = i$. $\mathbf{W}$ is a diagonal matrix with $\mathbf{W}_{ii} = 1/\sigma_i^2$. $\mathbf{x}$ is iteratively improved by $\Delta\mathbf{x}$ until the χ^2 objective function achieves a minimum value.

The Gradient descent method:

The gradient descent method is a parameter estimation technique that the parameter update is performed by taking steps along the negative direction of the gradient of the objective function to reach the minimum of the objective function (further information of this method can be read in [88]). Gavin [67] states that gradient descent method is reliably used to search numerous parameters in minimisation problems. However, the method provides slow convergence [117].

The gradient function of $\chi^2(\mathbf{x})$ is calculated as follows:

$$\begin{aligned}\frac{\partial\chi^2}{\partial\mathbf{x}} &= (\mathbf{y}^h - \mathbf{y}(\mathbf{x}))^T\mathbf{W}\frac{\partial(\mathbf{y}^h - \mathbf{y}(\mathbf{x}))}{\partial\mathbf{x}} \\ &= -(\mathbf{y}^h - \mathbf{y}(\mathbf{x}))^T\mathbf{W}\left\{\frac{\partial\mathbf{y}(\mathbf{x})}{\partial\mathbf{x}}\right\} \\ &= -(\mathbf{y}^h - \mathbf{y}(\mathbf{x}))^T\mathbf{W}\mathbf{J},\end{aligned} \tag{2.2}$$

where $\mathbf{J} = \frac{\partial\mathbf{y}(\mathbf{x})}{\partial\mathbf{x}}$ of size $p \times q$ is the Jacobian matrix representing local sensitivity of $\mathbf{y}$ to small changes in $\mathbf{x}$. The changes in $\mathbf{x}$, $\Delta\mathbf{x}$ in the direction of gradient descent can be expressed as:

$$\Delta\mathbf{x}_g = c\mathbf{J}^T\mathbf{W}(\mathbf{y}^h - \mathbf{y}(\mathbf{x})), \tag{2.3}$$

where c is the step size of the steepest descent direction with $c > 0$.

Gauss-Newton approach:

The Gauss-Newton method is applied to find the local minimum of the sum of squares objective function (2.1). The objective function is approximated as quadratic near the

optimal parameter values. Gauss-Newton is claimed to be faster to converge than the gradient descent method in medium dimension problems [50, 119].

In Gauss-Newton the model $\mathbf{y}$ is approximated with a first-order Taylor expansion as:

$$\mathbf{y}(\mathbf{x}+\Delta\mathbf{x}) \approx \mathbf{y}(\mathbf{x}) + (\frac{\partial \mathbf{y}(\mathbf{x})}{\partial \mathbf{x}})\Delta\mathbf{x} = \mathbf{y}(\mathbf{x}) + \mathbf{J}\Delta\mathbf{x}. \tag{2.4}$$

Substituting Equation (2.4) into Equation (2.1), we find that:

$$\begin{aligned}
\chi^2(\mathbf{x}+\Delta\mathbf{x}) &\approx (\mathbf{y}^h)^T\mathbf{W}\mathbf{y}^h - 2(\mathbf{y}^h)^T\mathbf{W}\mathbf{y}(\mathbf{x}+\Delta\mathbf{x}) + \mathbf{y}(\mathbf{x}+\Delta\mathbf{x})\mathbf{W}\mathbf{y}(\mathbf{x}+\Delta\mathbf{x}) \\
&\approx (\mathbf{y}^h)^T\mathbf{W}\mathbf{y}^h - 2(\mathbf{y}^h)^T\mathbf{W}\mathbf{y}(\mathbf{x}) - 2(\mathbf{y}^h)^T\mathbf{W}\mathbf{J}\Delta\mathbf{x} + \mathbf{y}(\mathbf{x})^T\mathbf{W}\mathbf{y}(\mathbf{x}) \\
&\quad + \mathbf{y}(\mathbf{x})^T\mathbf{W}\mathbf{J}\Delta\mathbf{x} + (\mathbf{J}\Delta\mathbf{x})^T\mathbf{W}\mathbf{y}(\mathbf{x}) + (\mathbf{J}\Delta\mathbf{x})T\mathbf{W}(\mathbf{J}\Delta\mathbf{x}) \\
&\approx (\mathbf{y}^h)^T\mathbf{W}\mathbf{y}^h + \mathbf{y}(\mathbf{x})^T\mathbf{W}\mathbf{y}(\mathbf{x}) - 2(\mathbf{y}^h)^T\mathbf{W}\mathbf{y}(\mathbf{x}) \\
&\quad - 2(\mathbf{y}^h - \mathbf{y}(\mathbf{x}))^T\mathbf{W}\mathbf{J}\Delta\mathbf{x} + \Delta\mathbf{x}^T\mathbf{J}^T\mathbf{W}\mathbf{J}\Delta\mathbf{x}
\end{aligned} \tag{2.5}$$

where $\mathbf{J}^T\mathbf{W}\mathbf{J}$ is an approximation to the Hessian of $\chi^2(\cdot)$. In order to find $\Delta\mathbf{x}$ that minimizes $\chi^2(\cdot)$, we set $\frac{\partial\chi^2}{\partial\Delta\mathbf{x}} = 0$, obtaining:

$$\frac{\partial\chi^2(\mathbf{x}+\Delta\mathbf{x})}{\partial\Delta\mathbf{x}} \approx -2(\mathbf{y}^h - \mathbf{y}(\mathbf{x}))^T\mathbf{W}\mathbf{J} + 2\Delta\mathbf{x}^T\mathbf{J}^T\mathbf{W}\mathbf{J}. \tag{2.6}$$

This provides the change in $\mathbf{x}$, $\Delta\mathbf{x}_N$, so for the Gauss-Newton method we find

$$(\mathbf{J}^T\mathbf{W}\mathbf{J})\Delta\mathbf{x}_N = \mathbf{J}^T\mathbf{W}(\mathbf{y}^h - \mathbf{y}(\mathbf{x})) \tag{2.7}$$

Gauss Levenberg Marquardt (GLM) method:

GLM combines gradient descent and Gauss Newton for updating the parameters. The parameters update in GLM is expressed as:

$$(\mathbf{J}^T\mathbf{W}\mathbf{J} + \lambda I)\Delta\mathbf{x}_{LM} = \mathbf{J}^T\mathbf{W}(\mathbf{y}^h - \mathbf{y}(\mathbf{x})) \tag{2.8}$$

where $\lambda > 0$ is the Marquardt parameter which controls the switch between Gauss-Newton and gradient descent and I is the identity matrix. When the value of λ is high, equation (2.8) becomes equation (2.3), i.e. the update of parameters follows the gradient descent method. By contrast, when λ is small, equation (2.8) changes into equation (2.7),so the parameter updates are based on the Gauss-Newton approach. When parameters are far from the optimal solution, GLM acts as gradient descent and converges slowly to the solution. When approaching the minimum, λ is gradually reduced and GLM behaves as Gauss-Newton and converges rapidly to the local minimum [50, 117, 119].

Equation (2.8) has a disadvantage when λ is large whereby the Hessian term in Equa-

tion (2.8) becomes no use. However, Marquardt [50] suggests replacing the identity matrix in Equation (2.8) with the diagonal of Hessian. Therefore, Equation (2.8) changes into:

$$(\mathbf{J}^T\mathbf{W}\mathbf{J} + \lambda diag(\mathbf{J}^T\mathbf{W}\mathbf{J}))\Delta\mathbf{x}_{LM} = \mathbf{J}^T\mathbf{W}(\mathbf{y}^h - \mathbf{y}(\mathbf{x})) \tag{2.9}$$

Using the fact that there is a proportional relationship between the Hessian and the curvature of error function ($\chi^2(\mathbf{x})$) the update (Equation 2.9) allows a large step in the direction of low curvature (nearly flat surface) and a small step in direction of the high curvature (steep curve).

The Jacobian matrix is approximated numerically using numerical forward, backward and central differences, as follows:

Forward differences:

$$\mathbf{J}_{ij} = \frac{\partial \mathbf{y}_i}{\partial \mathbf{x}_j} = \frac{\mathbf{y}_i(\mathbf{x} + \delta\mathbf{x}_j) - \mathbf{y}_i(\mathbf{x})}{||\delta\mathbf{x}_j||} \tag{2.10}$$

Backward differences:

$$\mathbf{J}_{ij} = \frac{\partial \mathbf{y}_i}{\partial \mathbf{x}_j} = \frac{\mathbf{y}_i(\mathbf{x}) - \mathbf{y}_i(\mathbf{x} - \delta\mathbf{x}_j)}{||\delta\mathbf{x}_j||} \tag{2.11}$$

Central differences:

$$\mathbf{J}_{ij} = \frac{\partial \mathbf{y}_i}{\partial \mathbf{x}_j} = \frac{\mathbf{y}_i(\mathbf{x} + \delta\mathbf{x}_j) - \mathbf{y}_i(\mathbf{x} - \delta\mathbf{x}_j)}{2||\delta\mathbf{x}_j||} \tag{2.12}$$

where $\delta\mathbf{x}_j$ is non-zero at j and equal to $dx_j(1 + |x_j|)$, and dx_j is fractional increment of x_j.

The parameter update is controlled by a gain factor, r [74] calculated according to the following expression:

$$r = \frac{\chi^2(\mathbf{x}) - \chi^2(\mathbf{x} + \Delta\mathbf{x}_{LM})}{2\Delta\mathbf{x}_{LM}^T(\lambda\Delta\mathbf{x}_{LM} + \mathbf{J}^T\mathbf{W}(\mathbf{y}^h - \mathbf{y}(\mathbf{x})))} \tag{2.13}$$

Gavin [67] states some rules for updating $\mathbf{x}$ and λ. Two of these rules are described as follows:

(i) The initial λ is specified by the user. $\Delta\mathbf{x}_{LM}$ is calculated according to equation (2.9). If $r > \epsilon_r$, $\mathbf{x} = \mathbf{x} + \Delta\mathbf{x}$. $\lambda_{n+1} = max[\lambda_n/(a\ shrinkage\ factor), 10^{-7}]$, otherwise $\lambda_{n+1} = min(\lambda_n * (an\ expansion\ factor), 10^7)$

(ii) The initial λ is specified by the user. $\lambda_0 = \lambda_u max\{diag(\mathbf{J}^T W\mathbf{J})\}$, where λ_u is a value specified by a user. $\Delta\mathbf{x}_{LM}$ is calculated according to equation (2.8). If $r > \epsilon_r$, $\mathbf{x} = \mathbf{x} + \Delta\mathbf{x}$. $\lambda_{n+1} = \lambda_n max(1/\omega, 1 - (\tau - 1)(2r_n - 1)^p)$; $\nu = \tau$ otherwise $\lambda_{n+1} = \lambda_n\nu$ and $\nu_{n+1} = 2\nu_n$ where $\nu_n = 2$ and $\omega = p = 3$.

Based on experimental tests conducted by Gavin [67], the best performance was provided by the first rule with an expansion factor of 11 and a shrinkage factor of 9. Readers

who are interested in GLM in detail may read [67]. For this study, we use the first rule for GLM. The workflow of GLM is depicted in Figure 2.4. Detailed code is presented in Appendix-Gauss Levenberg Marquardt (GLM).

2.5.2 Ant colony optimisation (ACO)

The background of ant colony optimisation

Swarm intelligence is an important concept in artificial intelligence and computer science. The idea of swarm intelligence is to design multiple simple agents leading to a desired global behaviour. The swarm intelligence takes inspiration from the collective behaviour of social insects, such as colonies of ants, bees, fire flies, or other animals, such as flocks of birds, colonies of bats or schools of fish. Collective behaviour emerges through the coordination of simple actions among members in a colony without a central control. This results in swarm intelligence being decentralised and self organising. According to Bonabeau [51], self organization involves:

- Positive feedback: simple actions of individuals are performed to build convenient structures. Recruitment and reinforcement by ants or dances by bees are considered to be positive feedback.
- Negative feedback: counterbalance of positive feedback. It helps to create stabilisation in collective behaviour. Negative feedback can be passive or active. Passive effect such as overcrowding in ant foraging behaviour. Ants on the strong recruitment route to food source may push other ants to use an alternative route. This allows the majority of ants to quickly move to another route (maybe a superior path) while foraging at current path is in progress. Active feedback is shown by honey bee. Honey bees exhibit inhibitory signals to other honey bees to reduce recruitment in dangerous foraging path. Negative feedback prevent colonies to be trapped in suboptimal decisions as a results of rapid group decision made from positive effect.
- Fluctuations: random walks, errors, random task-switching act to create new solutions.
- Multiple interactions: as self organisation, individuals in a colony should be able to make use of their own activities and other's activities by means of a form of communication system (multiple direct or indirect interactions) among individuals to build a group-level decision.

In nature these behaviours are used to solve problems such as searching for food sources, avoiding predators, attracting mates or relocating a colony. The information is

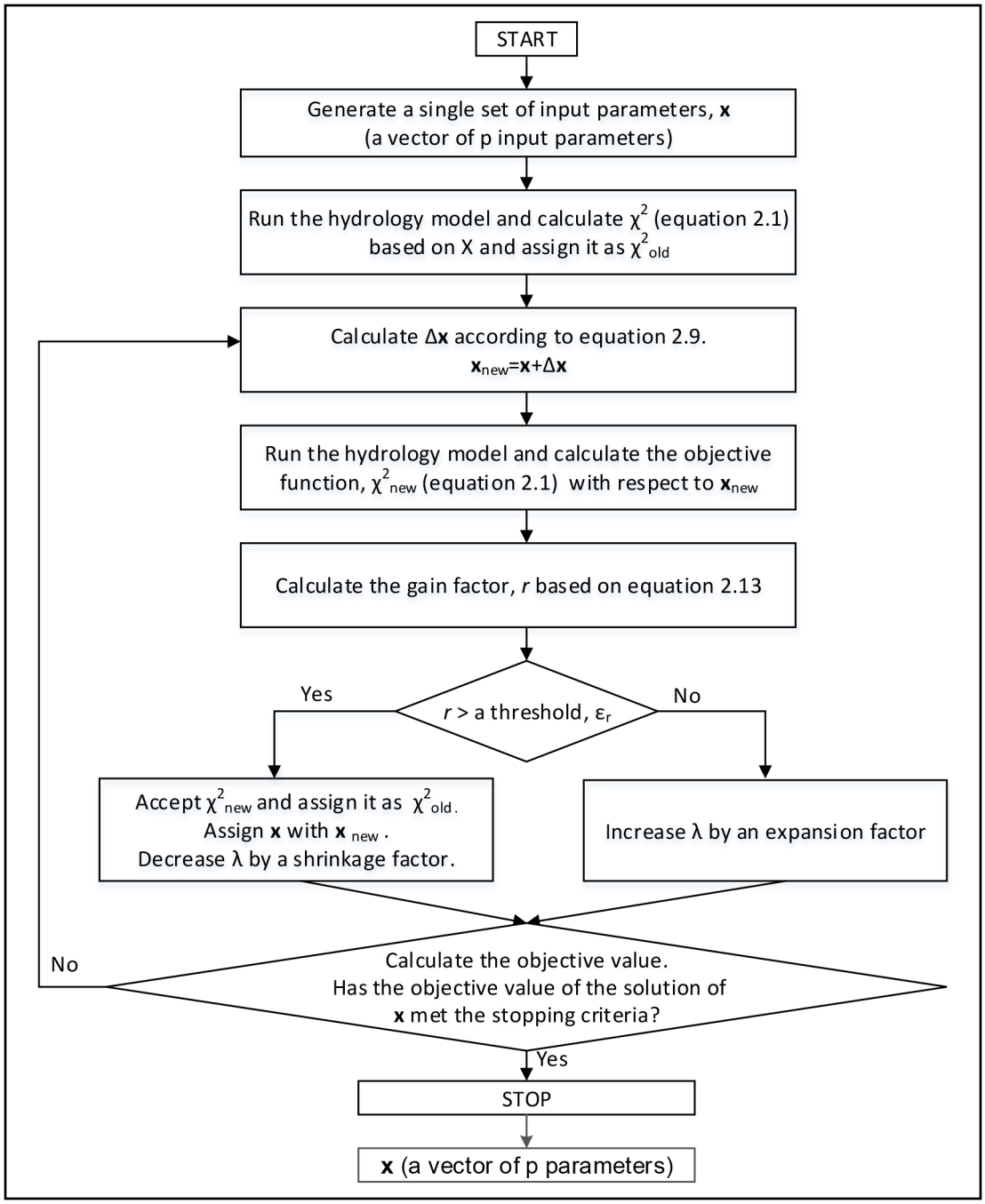

Figure 2.4: GLM workflow

stored through participants in the colony or is stored or communicated in the environment, for instance using pheromones in ants, dancing in bees, and proximity in fish and birds. The characteristics of swarm intelligence lead to a good model for algorithms dealing with increasingly complex problems, such as optimisation problems.

Ant colony optimisation (ACO) is one of the swarm intelligence algorithms. The ACO was developed by Dorigo [113] to handle difficult real world optimisation problems. The algorithm was inspired by foraging behaviour of ants for which each ant communicates indirectly using a pheromone trail to find the shortest path between an ant nest and food sources. Grasse [140] introduced the"stigmergy concept" to describe indirect communication among termites in a colony to dynamically build a structure. In the stigmergy mechanism, chemical signals, such as pheromone concentration, is used by ants to dynamically build a coherent structure.

The behaviour of ants in finding shortcuts was studied by Goss et al. [164]. The experiment used Argentine ants (*Irydomyrmex humilis*) and two paths with different lengths connecting the nest and the food source. The experiment can be simply depicted as in Figure 2.5. Initially, ants randomly walk on both paths. When the ants are moving toward the food source, they drop chemical pheromones forming a trail on the ground. Ants have the ability to smell the pheromone. The greater the concentration of pheromone on a path, the higher the probability that following ants will choose this path. As ants using the shorter path first reach the destination, when they return the probability for them to select the same path is higher. On the way home, ants also deposit pheromones resulting in stronger pheromone reinforcement for the shortest path. This attracts more ants on the shorter path. Pheromones contained on the longer path evaporate and, ultimately, all ants select the shortest path.

Modelling real ant behaviours in finding the shortest path

Real ant foraging behaviours for different lengths of paths was modelled by Goss et al. (1989) [164]. Goss conducted an experiment by building a bridge that had short (f) and long (g) paths. The ends of the bridge were numbered with $i = 1, 2$. The nest was placed near 1, while food was near 2. Ant behaviours were observed both leaving from and returning to the nest. The number of ants from each direction per second was denoted as N_a. p_{fi} denotes the probability of selecting the short path and p_{gi} for the long path. The probability of a path to be selected depended on the quantity of pheromone in each path: F_i for the short path and G_i for the long path. The time required crossing path f was based on the average speed of ants, the length of f and the time taken to deposit pheromone, which was 20 sec. Meanwhile, for path g, pheromone deposition required 20r sec, where r is the ratio of the length of g to f. The probabilistic model of ant foraging

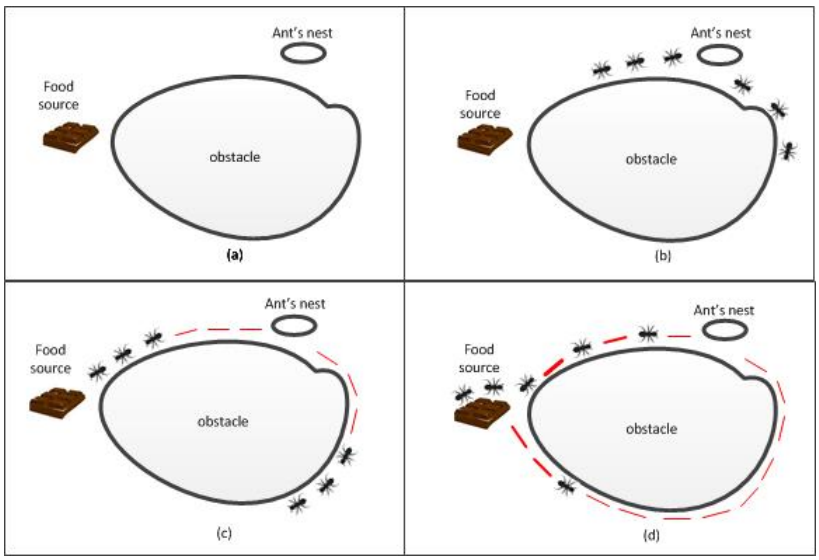

Figure 2.5: Foraging behaviour of ants in finding the shortest way to a food source

behaviours at time t is expressed mathematically as:

$$\frac{\delta F_i}{\delta t} = N_a p_{fi^*}(t-20) + N_a p_{fi}(t) \tag{2.14}$$

$$\frac{\delta G_i}{\delta t} = N_a p_{gi^*}(t-20r) + N_a p_{gi}(t) \tag{2.15}$$

where $i = 1, i^* = 2$ and $i = 2, i^* = 1$

$$p_{fi} = \frac{(a+F_i)^b}{(a+F_i)^b + (a+G_i)^b} \tag{2.16}$$

$$p_{fi} + p_{gi} = 1 \tag{2.17}$$

Equation (2.16) models the way ants to choose the paths. This function was introduced by Deneuborg et al. [84] to describe the selection between two equal paths. The parameters a and b were determined by fitting the model to observed choices of paths by the ants. Using Monte Carlo simulation, a and b were approaching 20 and 2 respectively [84]. Every ant which passed through the selected path modified the path by adding pheromone. This positive feedback led to one path being selected.

Equation (2.14) and Equation (2.15) describe how the short path became the preferable choice for the ants. At location $i = 1$, ants initially walked randomly through both paths and deposited pheromone on the paths. After 20 sec, the content of pheromone on the short path increased since ants came from both directions, i.e., ants travelling toward and returning from the food source which selected the short path. Meanwhile, the whole

long path contained pheromone after 20r sec. Between these times (20 sec and 20r sec), the short path gained more benefits as the pheromone concentration was amplified. This increased the probability that this path would be chosen.

This simple model of real ants may inspire the development of artificial ants in solving optimisation problems. Both real and artificial ants show similarities as multiple agents working together to achieve a goal. Real ants have a goal to find food, while artificial ants can be developed to find a solution for a given optimisation problem. Both real and artificial ants cooperate via stigmergy to incrementally build a solution. Real ants form pheromone trails, whilst for artificial ants, pheromones are in the form of numerical values and create a sequence known as artificial pheromone trails. Physical processes, such as evaporation, are also included in artificial ants as these occur in real ant. This enables the algorithm to remove memory of past history. Similar to real ants, artificial ants build a step by step solution on the basis of probabilistic decisions.

Ant System (AS)

The ACO algorithm was first introduced by Dorigo [113]. The first ACO variant, known as ant system (AS), was developed to solve combinatorial optimization problems, such as traveling salesman problem [5, 114, 115], quadratic problem [181] and job shop scheduling [6].

In the travelling salesman problem, the AS solution is built by constructing a fully connected graph, $G(N, E)$ where N is the set of nodes and E is the set of edges connecting two nodes. The constructed graph can be seen in Figure 2.6. Circles correspond to nodes and lines to edges. Each artificial ant, or simply "ant" for convenience, starts from node 1, 2, 3, 4, or 5. Ants move between nodes along edges on the graph to reach a destination. The movement of ants from the current node to the next node is determined probabilistically. Two nodes are connected if there is an edge $E(i, j) \in E$. Each edge $E(i, j)$ has length $d(i, j)$ and contains a pheromone trail (or simply "pheromone") $\tau(i, j)$. When ants move along the edge $E(i, j)$ they deposit pheromones ($\Delta\tau$) along the edges. The amount of pheromone deposited on the edge depends on the quality of solution developed.

Dorigo and Gambardella [114] stated AS follows two rules to construct a solution. The first rule is a probabilistic transition rule to choose the next node. This leads to selection of the shortest edge with high probability for the next transition as the short edge contains a higher amount pheromone, which results in a higher probability. The probability that ant k at node i to travel to node j can be expressed as:

$$p_k = \begin{cases} \frac{[\tau(i,j)]^a \eta(i,j)^b}{\sum_{r \in R_k(i)} [\tau(i,r)]^a [\eta(i,r)]^b} & \text{if } j \in R_k(i) \\ 0 & \text{otherwise.} \end{cases} \tag{2.18}$$

Here $\tau(i,j)$ is the pheromone on edge $E(i,j)$, $\eta(i,j) = 1/d(i,j)$ is a heuristic function, the inverse of the length of edge $E(i,j)$. $R_k(i)$ is the set of remaining nodes to be visited by ant k from node i. The a and b parameters describe the relative importance of pheromone versus heuristic function where $a, b > 0$. Equation (2.18) describes that the shorter edge resulting in higher heuristic value provide higher chance to be selected.

The second rule is global updating. When all m ants complete the journey, all nodes have been visited, and the pheromone on every edge will be updated as follows:

$$\tau(i,j) \leftarrow (1-\rho)\tau(i,j) + \sum_{k=1}^{m} \Delta\tau_k(i,j), \tag{2.19}$$

where

$$\Delta\tau_k(i,j) = \begin{cases} L_k^{-1} & (i,j) \in \text{tour of ant k} \\ 0 & \text{otherwise} \end{cases} \tag{2.20}$$

and $\rho \in (0,1]$ is a pheromone evaporation parameter. L_k is the length of tour performed by ant k. Pheromone is locally stored on edge $E(i,j)$ and used as indirect communication, or stigmergy, between ants.

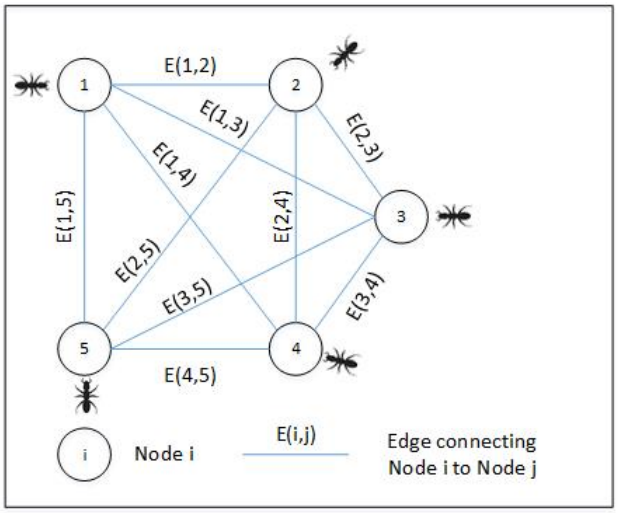

Figure 2.6: Constructed graph of traveling salesman problem

Continuous Ant Colony Optimization

The first continuous ACO technique, known as Continuous ACO (CACO), was introduced by Bilchev and Parmee [59]. The CACO method initially uses a random location, termed a nest for ants to start the journey. When they find good solutions they are stored in a set of vectors. Ants probabilistically select one of the vectors, and from the selected vector search

randomly within a radius of exploration. Update of the vector is performed when the best result is found. Another ACO approach on continuous domains is the API (the name was derived from primitive ants *Pachycondyla APIcalis*) algorithm introduced by Monmarche et al. [131]. The ants start their journeys from the same nest to search the good regions independently. Once a good solution is obtained, the nest is moved periodically. This algorithm can be used to handle both discrete and continuous optimisation problems. Another variant of ACO is the Continuous Interacting Ant Colony (CIAC) algorithm, and is used to solve continuous problems. This method was proposed by Dréo and Siarry [80]. CIAC combines indirect communication using pheromone deposits in the search space and direct communication between ants.

Socha and Dorigo [98] claimed that the above approaches have different concepts from the original ACO, thus those approaches may not be considered as extensions of ACO for continuous domains. Socha and Dorigo [99] proposed new ACO approach for handling continuous problems, known as ACO_R. This approach was considered to closely follow the ACO concept.

Continuous Ant Colony Optimization- ACO_R

The main idea of ACO_R is constructing a solution archive $\mathbf{X}$, with dimensions of k model solutions $\times$ p input parameters, as depicted in Figure 2.7. The archive also stores the objective functions $\{h(\mathbf{x}_i)\}$, and weights $\{w_i\}$. The solutions are ranked based on the quality of the objective functions. Lower values of the objective functions receive higher ranks and are stored in the top rows. The weights are computed for generating new solutions as follows [98]:

$$w_i = \frac{1}{qk\sqrt{2\pi}} exp\left\{\frac{-(i-1)^2}{2q^2k^2}\right\} \tag{2.21}$$

where q is the parameter control for selecting model in the archive of ACO_R. When q is small the best-ranked solutions are selected preferentially for the new generation. The solutions listed in the higher rows obtain higher weights. This means that the probability of these solutions being selected for generating new solutions are high. The probability of being selected is the normalised weight:

$$p_i = \frac{w_i}{\sum_{r=1}^{k} w_r} \tag{2.22}$$

The information provided in the archive is then used to dynamically build a set of probability density functions (PDFs). Every column in the archive represents a multimodal PDF, denoted $G_j(\mathbf{x})$. Thus, for the whole archive there are p multimodal PDFs. Socha and Dorigo [98] proposed a mixture of Gaussian kernels to approximate a multimodal PDF. A mixture of Gaussian kernels for input parameter j, $G_j(\mathbf{x})$, consists of a

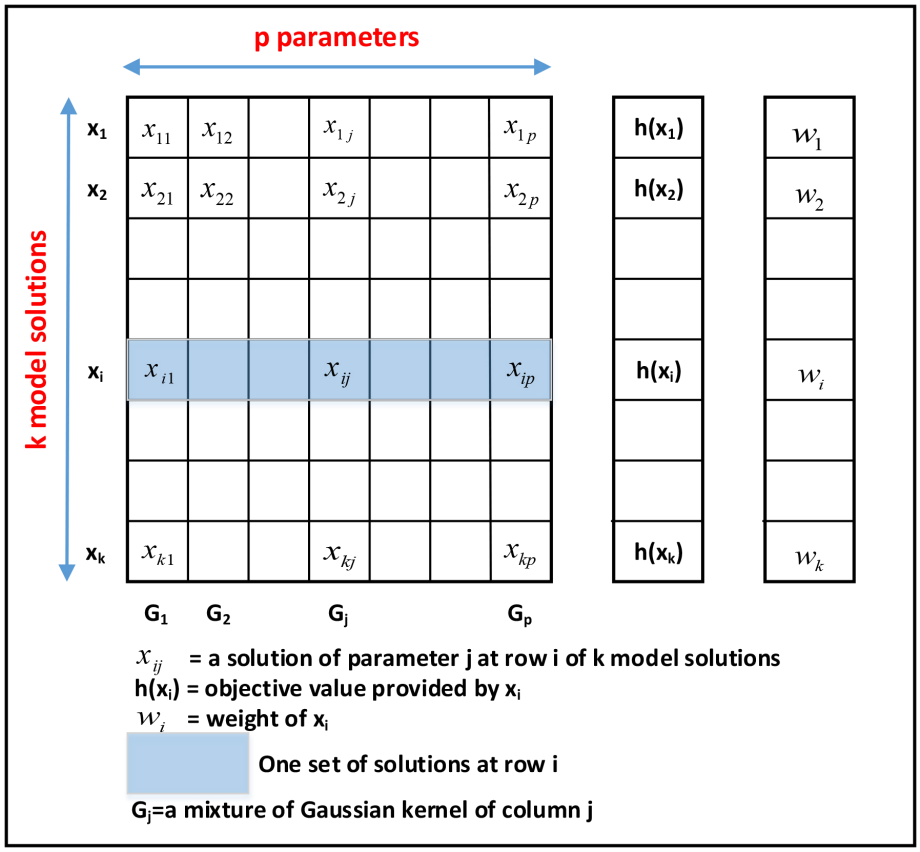

Figure 2.7: Solution archive of ACO_R

weighted sum of k one-dimensional Gaussian functions $\{g_{ij}(x)\}$, as follows:

$$G_j(\mathbf{x}) = \sum_{i=1}^{k} w_i g_{ij}(x) = \sum_{i=1}^{k} \frac{w_i}{\sigma_{ij}\sqrt{2\pi}} \exp\left\{\frac{-(x-\mu_{ij})^2}{2\sigma_{ij}^2}\right\}, \tag{2.23}$$

where $j \in \{1, ...p\}$ is the number of input parameter dimensions. The Gaussian kernel $G_j(\mathbf{x})$ is parameterised by the mean vector $\boldsymbol{\mu}_j$ and vector of standard deviations $\boldsymbol{\sigma}_j$. Each element of $\boldsymbol{\mu}_j$ is defined by the corresponding value in the solution archive, so $\mu_{ij} = x_{ij}$ for $i = 1, \ldots, k$. Each component of $\boldsymbol{\sigma}_j$ is computed as:

$$\sigma_{ij} = \frac{\xi}{k-1} \sum_{l=1}^{k} |x_{lj} - x_{ij}| \tag{2.24}$$

where $i \in \{1, \ldots, k\}$. We see that σ_{ij} is the average absolute difference between the selected solution and the other solution in column j multiplied by $\xi > 0$ (pheromone evaporation). Pheromone evaporation controls how fast the algorithm forgets worse solutions. A lower value for ξ drives the algorithm to new regions of the parameter space, which leads to faster convergence.

The workflow of ACO_R for this study is depicted in Figure 2.8. Detailed code is

presented in Appendix-Ant Colony Optimisation (ACO_R).

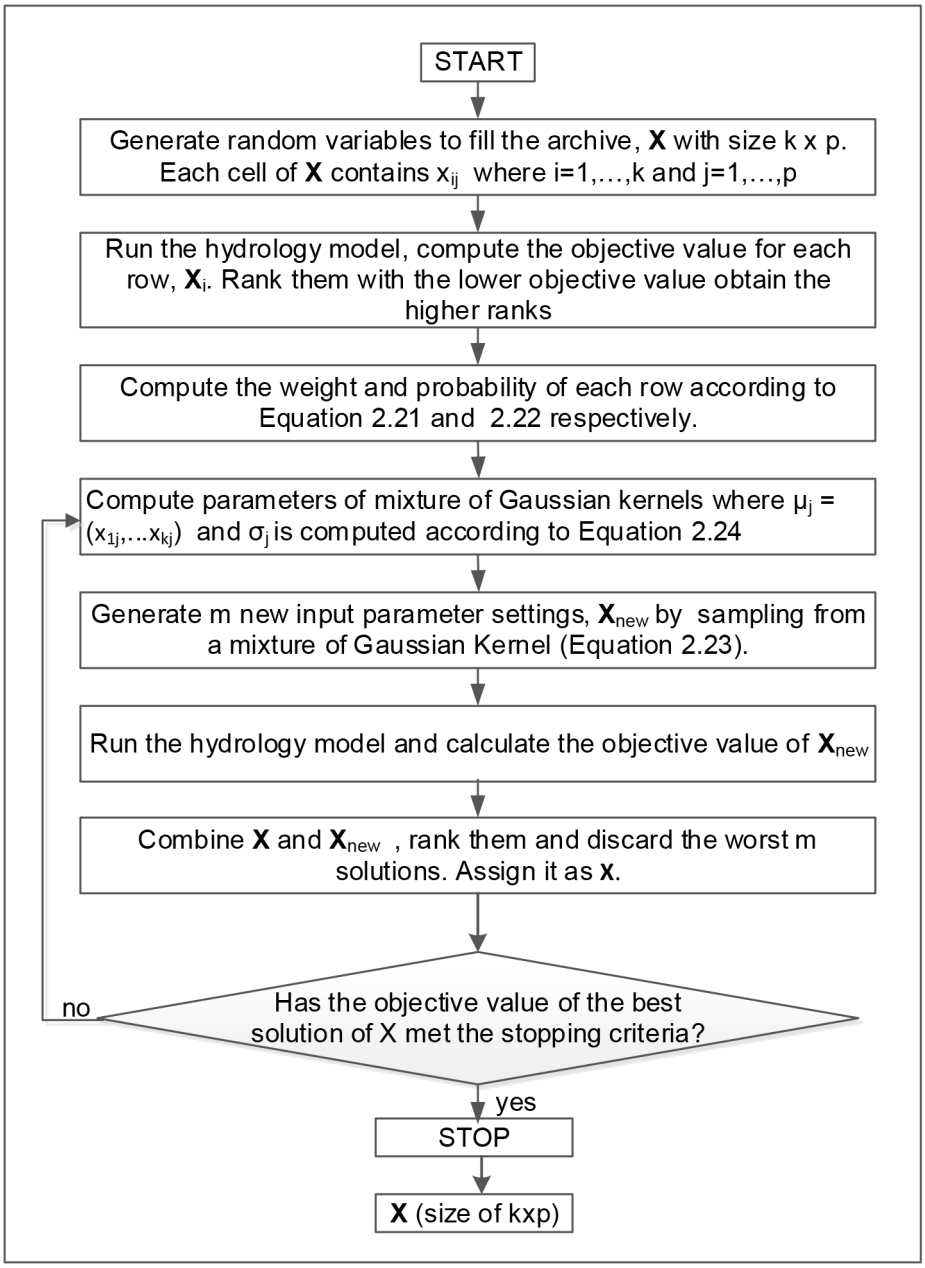

Figure 2.8: ACO_R workflow

Diversity Ant Colony Optimisation ($DACO_R$)

$DACO_R$ was proposed by Leguizamon and Coello Coello [63]. The aim of $DACO_R$ is to maintain diversity in a population of solutions, which is a common challenge encountered in large dimensional problems where ACO_R had limitations [63]. Maintaining diversity in the process to achieve the optimal solutions is important. A quick loss of diversity in solutions may result in multiple optimal regions in the input parameter space. $DACO_R$

was developed using a different kernel selection mechanism from ACO_R to generate new solutions. By applying this mechanism, the algorithm explores many more regions in the parameter space before it converges toward a local minimum.

DACO_R has the same philosophy as ACO_R for which the m solutions are generated by sampling multimodal kernels. However, the size of m in DACO_R is always the same as k. DACO_R implements two search strategies to maintain diversity in the population of solutions, namely local exploration and global exploitation. As a local explorer, the new solution at row i in $\mathbf{X}_{\text{new}}$ is generated by sampling Gaussian distributions $\{g_{i1}(\cdot), \ldots, g_{ip}(\cdot)\}$. Parameters of Gaussian distributions at row i are obtained from information at row i in $\mathbf{X}$ where $\boldsymbol{\mu} = (x_{i1}, \ldots, x_{ip})^T$ and $\boldsymbol{\sigma} = (\sigma_{i1}, \ldots, \sigma_{ip})^T$; σ_{ij} is calculated according to Equation (2.24). If the quality of the objective value of the new solution at row i improves, then the new solution is included in $\mathbf{X}_{\text{new}}$, otherwise the solution at row i in $\mathbf{X}$ occupies the corresponding row in $\mathbf{X}_{\text{new}}$.

In the second strategy, exploitation, DACO_R globally searches for the best region in input parameter space. The best solution in $\mathbf{X}$ is selected to obtain parameters of Gaussian distributions at row i where $\boldsymbol{\mu} = (x_{i_{\text{best}}1}, \ldots, x_{i_{\text{best}}p})^T$, and $\boldsymbol{\sigma} = (\sigma_{i_{\text{best}}1}, \ldots, \sigma_{i_{\text{best}}p})^T$ is calculated based on Equation (2.24). These Gaussian distributions are then sampled to generate a new solution at row i in $\mathbf{X}_{\text{new}}$. The new solution at position i is considered to be a member of $\mathbf{X}_{\text{new}}$ if there is an improvement in the objective value, otherwise the current solution at position i in $\mathbf{X}$ is placed in the corresponding position in $\mathbf{X}_{new}$. The switch between the two search strategies is determined by parameter u. u prevents the algorithm to be trapped in a certain region in input parameter space. DACO_R also uses parameter ξ which has the same role as ξ in ACO_R for calculating $\boldsymbol{\sigma}$. The workflow of DACO_R is depicted in Figure 2.9. Detailed code is presented in Appendix-Ant Colony Optimisation (DACO_R).

2.5.3 Robust Parameter Estimation (ROPE)

The ROPE technique was first proposed by Bardosy and Singh [4]. ROPE is a geometrical approach based on data depth. Data depth was first introduced by Tukey (1975) [89] to measure the degree of centrality of a point with respect to a multivariate data set. To illustrate the concept of data depth, we present Figure 2.10. The points in Figure 2.10 were obtained from sampling 30 observations from a bivariate normal distribution. The depths of the data points were calculated using the Tukey depth function [89]. The top figure depicts the depths of individual data points using shade and plotting point size. Lighter blue colours and bigger sizes reflect higher depth. Meanwhile, the bottom figure depicts different contours of the Tukey depth function, calculated using the Isodepth algorithm [77].

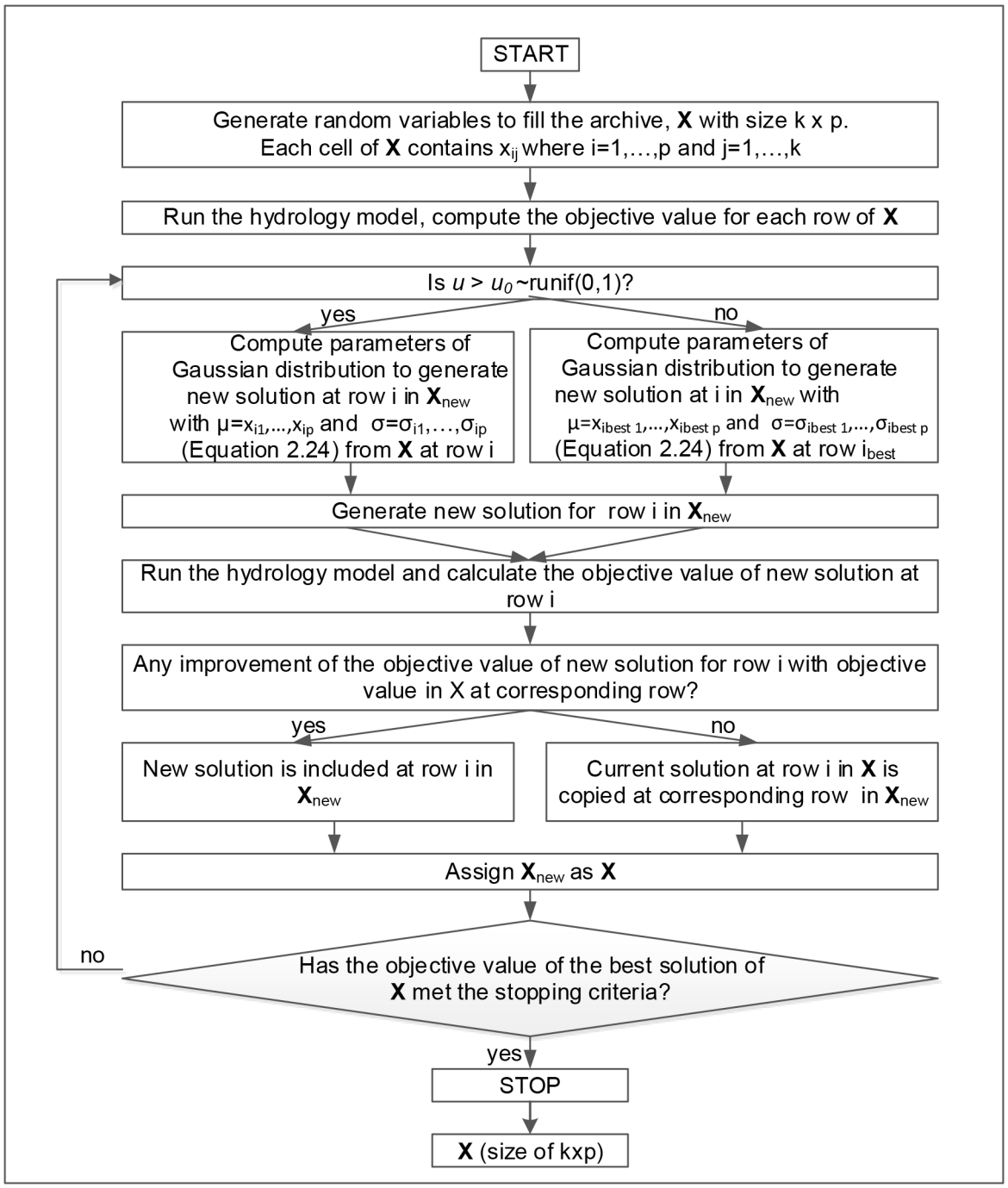

Figure 2.9: DACO$_R$ workflow

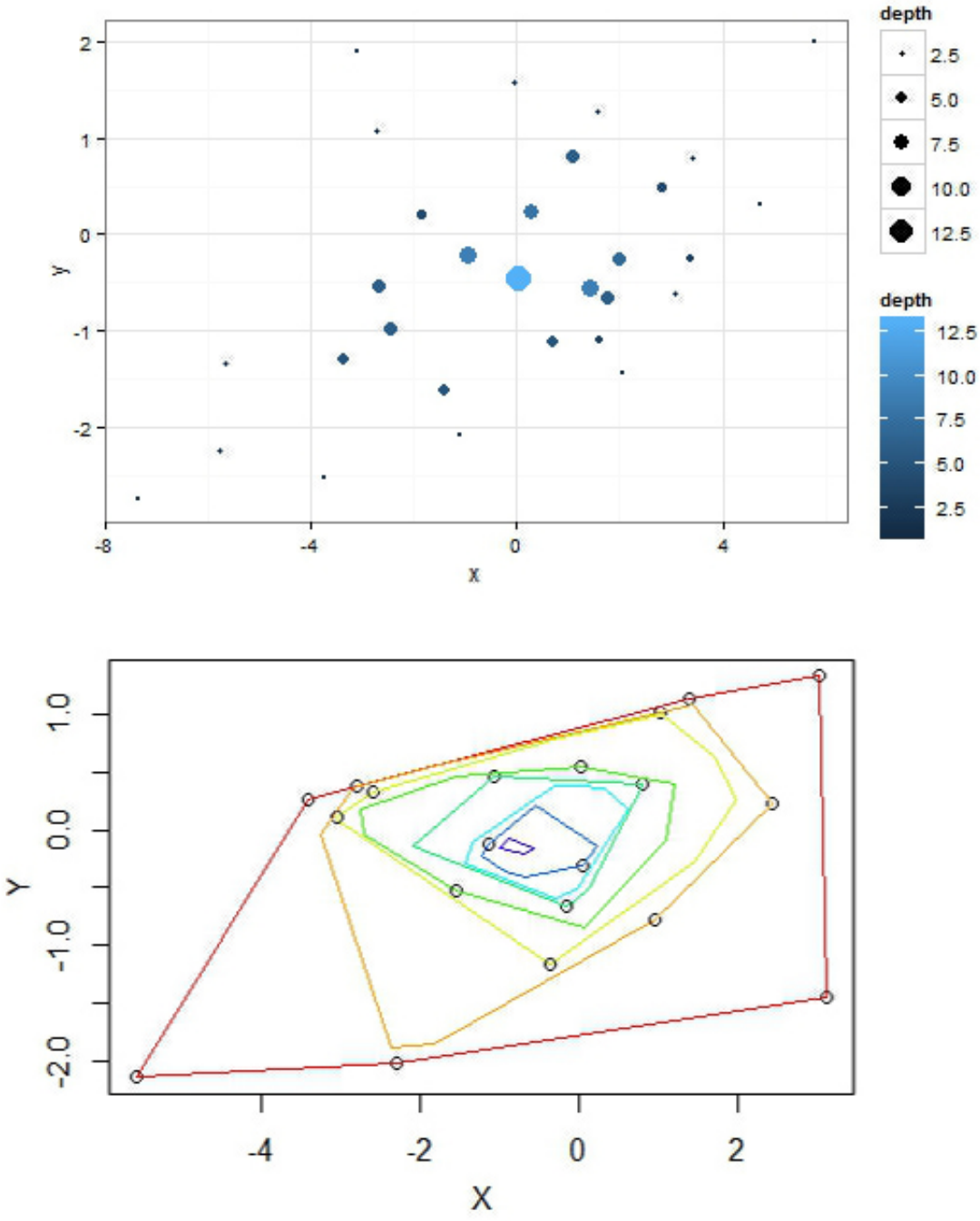

Figure 2.10: Depth of each coordinate (top) and depth contour plots of depth functions (bottom)

We see in Figure 2.10 that the data points with higher depth lie to the centre of the data. This provides a concept of "centre-outward ordering" of the multivariate data [154, 159, 167], which provides an approach to multivariate analysis [167]. The depth function is designed to measure the depth of a data point, $\mathbf{z} \in \mathbb{R}^p$ with respect to a set of p-dimensional data points, $\mathbf{X} \in \mathbb{R}^p$.

Liu [158] and Zuo [205] claimed that data depth has properties as follows:

- Affine invariance. The depth point $\mathbf{z} \in \mathbb{R}^p$ does not depend on the coordinate system.
- Maximality at centre. If a data set, $\mathbf{X}$ is symmetric at $\mathbf{z}$, then $\mathbf{z}$ has the maximal depth.
- Monotonic decreasing. The depth of a point $\mathbf{z} \in \mathbb{R}^p$ decreases as it is translated away along each ray from the deepest point.
- Vanishing at infinity. The depth of a point $\mathbf{z}$ approaches zero as $\mathbf{X} \to \infty$.

Some Types of Data Depth

Halfspace depth:

Halfspace depth, also known as Tukey or location depth, was introduced by Tukey (1975) [89]. The Halfspace depth of a point $\mathbf{z} \in \mathbb{R}^p$ with respect to a p-dimensional data set is defined as the minimum number of data points lying on a closed half-space plane with boundary of the plane through $\mathbf{z}$. The halfspace depth (DH) can be expressed mathematically as [28]

$$DH(\mathbf{z}; \mathbf{X}) = \min_{\|\mathbf{u}\|=1} \#(i; \mathbf{u}^T \mathbf{x_i} \geq \mathbf{u}^T \mathbf{z}), \tag{2.25}$$

where $\mathbf{X} = \{\mathbf{x_i} = (x_{i1}, ..., x_{ip}) : i = 1, \ldots, n)\}$. The sample version of halfspace depth is obtained by replacing $\mathbf{X}$ with $\mathbf{X}_n$.

The halfspace depth is usually normalised by the number of observations. This can be done by dividing halfspace depth with the number of points in $\mathbf{X_n}$, hence Equation 2.25 becomes:

$$DH(\mathbf{z}; \mathbf{X_n})^* = \frac{DH_p(\mathbf{z}; \mathbf{X_n})}{\#\{\mathbf{X_n}\}} \tag{2.26}$$

We present a visualization of the halfspace depth calculation for a two dimensional data set in Figure 2.11.

Figure 2.11 depicts two examples of how to calculate halfspace depths for a point $\mathbf{z}$ (blue hexagon) with respect to a set of data, $\mathbf{X_n}$ (green circles). In the first example (top figure), $\mathbf{z}$ lies inside $\mathbf{X_n}$, while in the second example (bottom figure), $\mathbf{z}$ is placed

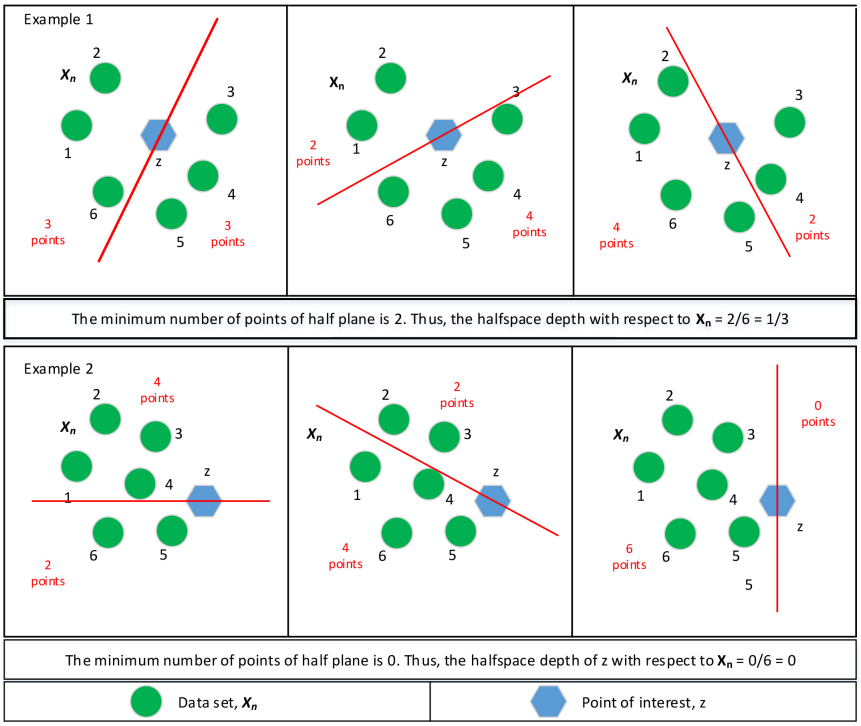

Figure 2.11: Examples to calculate the halfspace depth of a point, **z** of interest with respect a data set, **X** in two dimensional plane

a little further away from where $\mathbf{X_n}$ lies. To find the halfspace depth, a line passing through **z** is created to divide the two dimensional plane. The line is then rotated 360° clockwise to find the minimum points of $\mathbf{X_n}$ on one side of the line (or half space). The minimum number of points in $\mathbf{X_n}$ on a half plane are considered as a halfspace depth of **z** with respect to $\mathbf{X_n}$, which is usually described as a proportion. In the first example, the minimum number of points on one side of the line is 2, so the halfspace depth of point **z** relative to $\mathbf{X_n}$ is 2/6 or 1/3. In the second example the halfspace depth of **z** with respect to $\mathbf{X_n}$ is calculated to be zero since the minimum number of points in the set $\mathbf{X_n}$ on the half plane after full rotation of the line is zero. This information describes that a point located deep inside a data set has higher depth compared to a point lying near the boundary, so far from the centre of a set of data.

Tukey [90] states that depth is closely related to rank of data points. For instance, in a univariate case when a set of data is ranked, the highest and the lowest ranks are given depth one, while the second lowest and highest rank of data have depth two. Therefore, a median of data points which is a midpoint of an ordered set of data has the highest depth. In the multivariate case, a point with the maximum depth in a data set may be considered as a multidimensional median [28].

Mahalanobis depth

The Mahalanobis depth (DM) [137] of a point $\mathbf{z}$ with respect to a p-dimensional data set $\mathbf{X}$ is defined as

$$DM(\mathbf{z};\mathbf{X}) = \left\{1 + (\mathbf{z} - \boldsymbol{\mu}_{\mathbf{X}})\boldsymbol{\Sigma}_{\mathbf{X}}{}^{-1}(\mathbf{z} - \boldsymbol{\mu}_{\mathbf{X}})\right\}^{-1}, \tag{2.27}$$

where $\boldsymbol{\mu}_{\mathbf{X}}$ is the mean vector and $\sum_{\mathbf{X}}$ the covariance matrix of $\mathbf{X}$. The sample version of DM is expressed as

$$DM(\mathbf{z};\mathbf{X_n}) = \left\{1 + (\mathbf{z} - \bar{\mathbf{X}})\mathbf{S}^{-1}(\mathbf{z} - \bar{\mathbf{X}})\right\}^{-1}, \tag{2.28}$$

where $\bar{\mathbf{X}}$ and $\mathbf{S}$ are the sample mean and sample covariance matrix respectively.

Oja depth

Oja depth (DO) [71] at a point $\mathbf{z} \in \mathbb{R}^p$ is defined as the sum of the volumes of every closed simplex with a vertex at $\mathbf{z}$ and the p points of the data set $\mathbf{X}$. Liu *et al.* [158] expressed Oja depth at a point $\mathbf{z}$ with respect to $\mathbf{X}$ as:

$$DO(\mathbf{z};\mathbf{X}) = \{1 + \mathbf{F_X} V(S(\mathbf{z},\mathbf{X}))\}^{-1} \tag{2.29}$$

where $V(\cdot)$ denotes volume and $S(\mathbf{z},\mathbf{X})$ is the closed simplex with vertices $\mathbf{z}$ and p random observations $\mathbf{x}_1,\ldots,\mathbf{x}_p$ from $\mathbf{X}$. Liu *et al.* [158] further described the sample version of Oja depth where summation is selected for $\mathbf{F_X}$ as follows:

$$DO(\mathbf{z};\mathbf{X_n}) = \frac{1}{\binom{n}{p}\left\{1 + \sum_{*} V(S(\mathbf{z},\mathbf{x}_1,\ldots,\mathbf{x}_p))\right\}} \tag{2.30}$$

where $*$ is p-plets $(i_1,...i_p)$ such that $1 \le i_1 \le ... \le i_p \le n$.

To illustrate the calculation of Oja depth for a two dimensional data set, we present Figure 2.12. Oja depth in the bivariate case can be calculated by summing triangle areas formed by a point $\mathbf{z}$ and $p = 2$ other points within $\mathbf{X}_n$. The possible areas of triangles formed from four data points (1, 2, 3 and 4) are depicted in Figure 2.12. For each triangle, area is calculated according to the the length of two sides ($\mathbf{z}$ to $\mathbf{x}_i$ and $\mathbf{z}$ to $\mathbf{x}_j$ and the angle between these edges, where i and j are two points obtained from possible combinations of 1, 2, 3, and 4. The Oja depth is then calculated according to Equation (2.30).

Simplicial depth

Simplicial depth (DS) [157, 158] at a point $\mathbf{z}$ relative to a data set, $\mathbf{X} \in \mathbb{R}^p$ is defined as:

$$DS(\mathbf{z};\mathbf{X}) = P\{\mathbf{z} \in S[\mathbf{x}_1,...,\mathbf{x}_{p+1}]\} \tag{2.31}$$

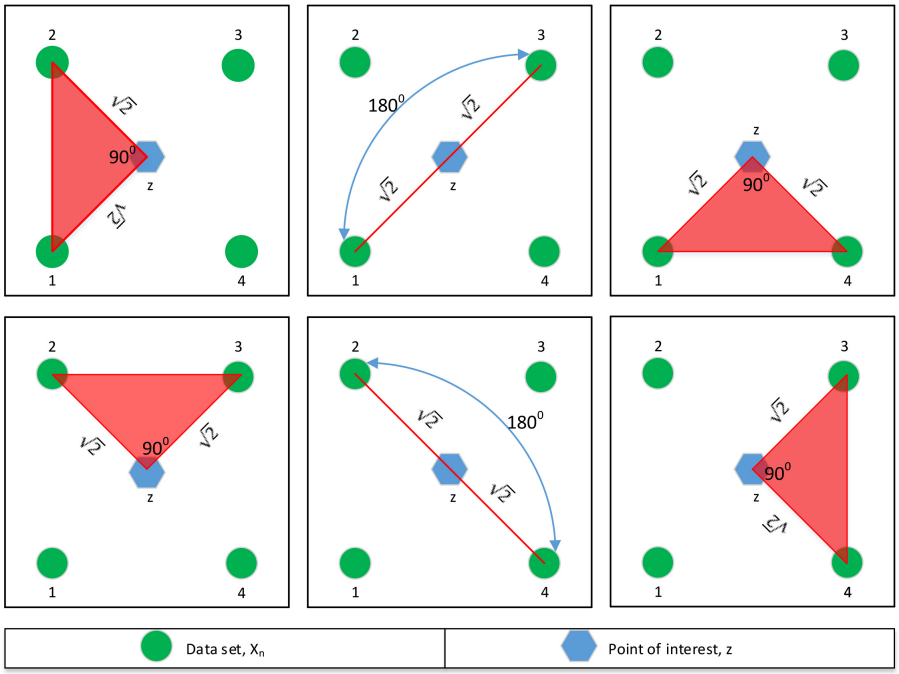

Figure 2.12: Oja depth obtained by calculating the area of combinations of triangles in the bivariate case

where $S(\mathbf{x}_1, ..., \mathbf{x}_{p+1})$ is a closed simplex formed by $p+1$ random observations $\mathbf{X}$. The sample version of the DS function based on random sample $\mathbf{x}_1, ..., \mathbf{x}_n$ from $\mathbf{X}$ on $\mathbb{R}^p$ is calculated as follows:

$$DS(\mathbf{z}; \mathbf{X_n}) = \binom{n}{p+1}^{-1} \sum_{*} I_{S(\mathbf{x}_1,...,\mathbf{x}_{p+1})} \tag{2.32}$$

where and $I(.)$ is an indicator function.

ROPE algorithm

ROPE was developed to provide a set of robust model parameters, lying deep in the good-performing model parameter vectors. This method shifts the classical concept which mainly focuses on minimizing an objective function by searching for the best set of input parameters.

Bardossy and Singh [4] state that the influence of measurement errors in hydrology may affect the resulting model parameters as well as model performance. Two parameter vectors resulting in different performances within the range of the measurement errors cannot be distinguished from one another. One of them may provide a better description of hydrological processes. Meanwhile, the best set of model parameters obtained by most HM techniques may give suboptimal estimates of the true model parameters. Therefore, providing a set of parameters which have similar performance is reasonable. These

parameters are known as good-performing parameters. Within these good-performing parameters, a set of robust model parameters can be found. The Depth function can be utilized to search robust model parameter vectors. Robust model parameters have properties as follows: [4, 78, 173]:

- Providing good performance within a selected period.
- Representing reasonable hydrological processes.
- Being insensitive to small changes in model parameters.
- Performing well in different periods of time.

The ROPE algorithm for this study is based on the concept of halfspace (Tukey) depth. We performed a modification of ROPE in order to improve the accuracy of ROPE. Within the steps in the modified ROPE algorithm we include a comparison between current model parameters (from iteration i) and previous model parameters (from iteration $i-1$). The purpose of this is to keep better-performing model parameter sets obtained from the previous iteration $(i-1)$, compared to the current parameter vectors (at iteration i). In order to clearly describe the steps of ROPE technique, we present the workflow in Figure 2.13. Detailed code is presented in Appendix-Robust Parameter Estimation (ROPE).

2.6 Challenges in Direct History Matching

The application of multiple solution-based approaches in HM is challenging in terms of computational time. Based on our experience, the use of the ROPE algorithm to estimate RRM parameters for the Dandalups took up to three days to provide a set of robust parameters with a sample size of 1000, using an PC with an Intel (R) Core (TM) i5 CPU at 3.3 GHz, 8 Gb of RAM. It can be imagined that the HM problem becomes much worse if the computer-based model is highly spatially resolved, such as 3D model which commonly applied in oil reservoir modelling. A 3D oil model may require up to a week to obtain a single output value for a given set of input parameters [11]. In these situations it is desirable to have surrogate models to replace the full computational simulators, providing reasonably accurate results with much less computational time.

2.7 Indirect History Matching

In indirect history matching (indirect HM), HM methods are executed to minimise the differences between observed and surrogate model response. The surrogate model is an

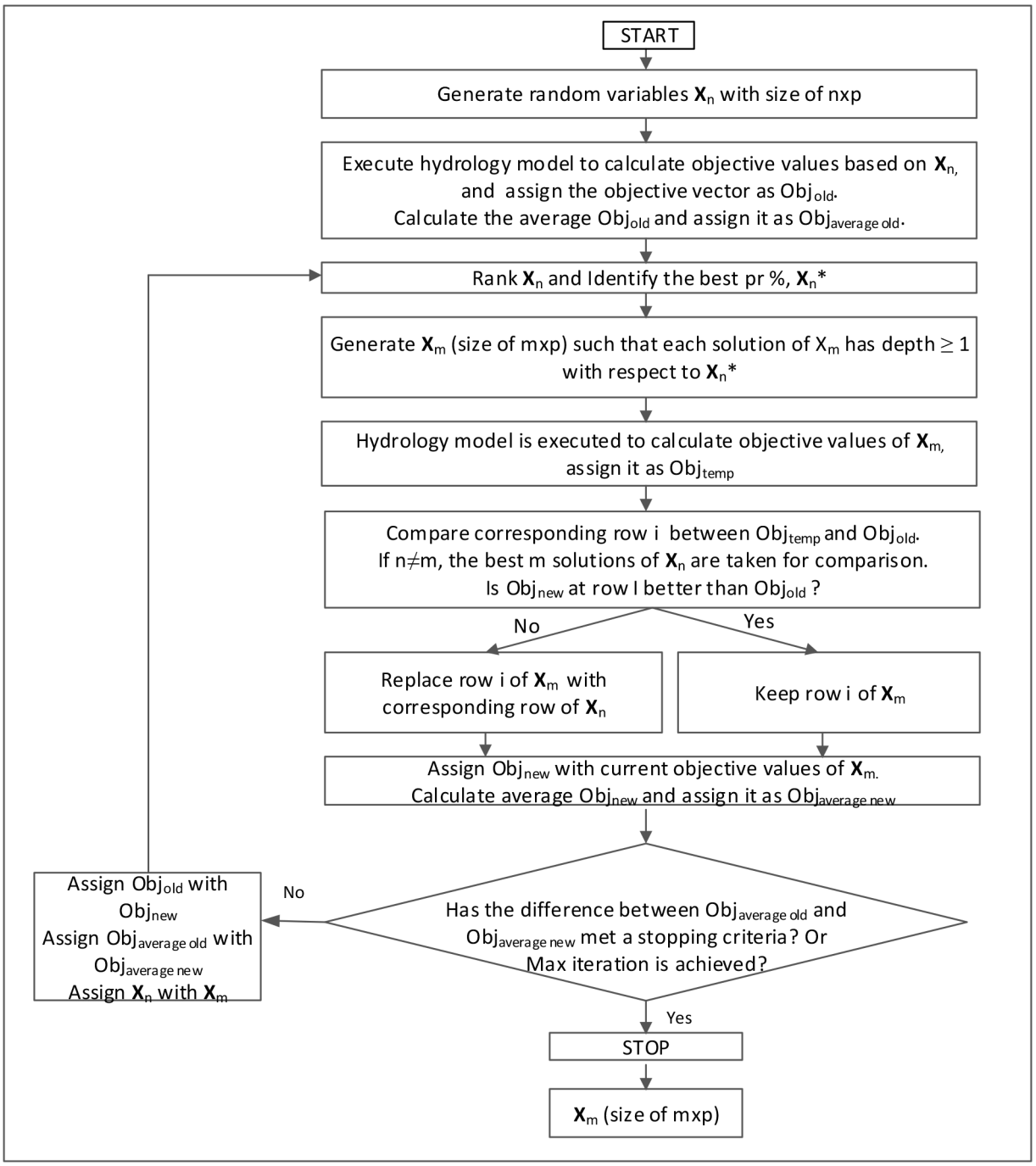

Figure 2.13: ROPE workflow

approximate model, developed to substitute the role of the RRM for providing model responses in HM. The approximate model can be represented by the following functional form:

$$y^s = g(\mathbf{x})$$

$$y = y^s + \epsilon \tag{2.33}$$

where $g(\cdot)$ is a surrogate model of p input parameters $\mathbf{x} = (x_1, x_2, ...x_p)$ and ϵ is the approximation error. Given a set of input parameters $\mathbf{X}$ and a set of responses $\mathbf{Y}$, the function $g(\cdot)$ can be estimated. In the case of replicating a RRM (or developing a surrogate model for a RRM), a set of responses $\mathbf{Y}$ is generated by executing RRM on a set of input parameters $\mathbf{X}$. $g(\cdot)$ can then be developed using a statistical technique. The workflow of the development of a surrogate model for a RRM is described in Figure 2.14.

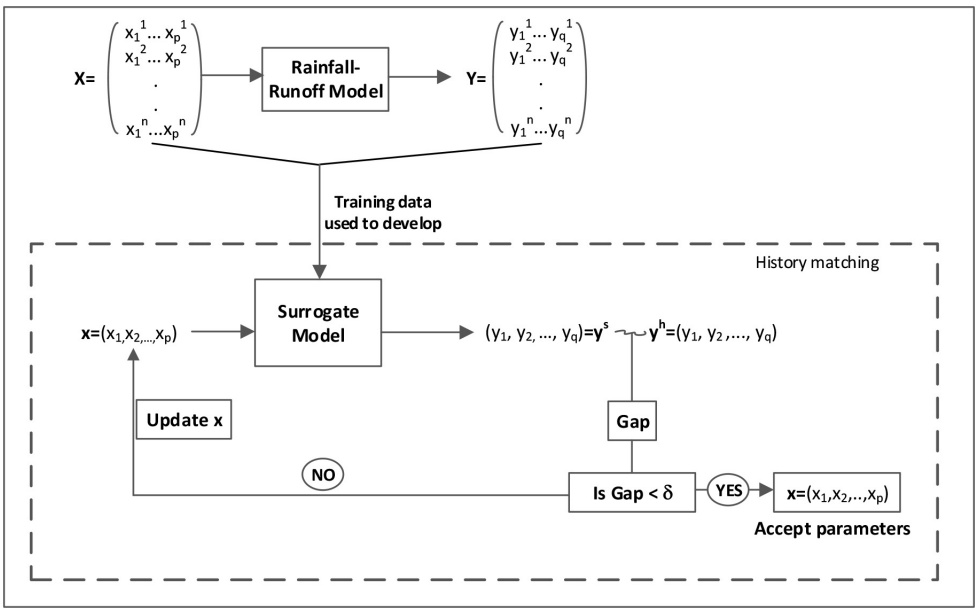

Figure 2.14: The workflow for the development of a surrogate model to approximate a hydrological model in HM.

In 2.14, the RRM is executed on a set of input parameter values, $\mathbf{X}$, where $\mathbf{x}^i = (x_1^i, x_2^i, \ldots, x_p^i)$ where i refers to i^{th} row of $\mathbf{X}$ and $i = 1, 2, \ldots, n$, to generate the output responses $\mathbf{Y}$. $\mathbf{Y}$ contains a set of responses with $\mathbf{y}^i = (y_1^i, y_2^i, ..., y_q^i)$ for $i = 1, \ldots, n$. The input parameter values, $\mathbf{X}$, and the responses $\mathbf{Y}$ become a set of data to train q surrogate models $\mathbf{Y}^s = g(\mathbf{X})$. The surrogate models are then used to replace the RRM in HM. For HM, the surrogate models are run on a set of input parameter values $\mathbf{x} = (x_1, x_2, \ldots, x_p)$ to produce the output response, $\mathbf{y}^s = (y_1, y_2, \ldots, y_q)$. With the availability of historically

observed values of $\mathbf{y}^h = (y_1^h, y_2^h, \ldots, y_q^h)$, the values of input parameter set of $\mathbf{x} = \mathbf{x}^h$ such that $\mathbf{y}^h = g(\mathbf{x}^h + \epsilon)$ are estimated. Different statistical models are used to develop surrogate models are described in the following sections.

2.7.1 Multidimensional Kriging Model

The Kriging method has been widely applied in geostatistics, and its application to computer simulated experiment was first proposed by Sacks et al. [85]. The Kriging model can be expressed as:

$$y(\mathbf{x}) = \sum_{i=1}^{k} \beta_i f_i(\mathbf{x}) + Z(\mathbf{x}) \tag{2.34}$$

where $f_i(\mathbf{x})$ is a known polynomial function of $\mathbf{x}$, the $\{\beta_i\}$ are unknown parameters. $Z(\mathbf{x})$ is a normally distributed Gaussian random process with mean zero and covariance matrix

$$Cov(Z(\mathbf{x}_i), Z(\mathbf{x}_j) = \sigma^2 \mathbf{R}(R(\mathbf{x}_i, \mathbf{x}_j)), \tag{2.35}$$

where σ^2 is the process variance, $\mathbf{R}$ is an $n \times n$ correlation matrix, and $R(\mathbf{x}_i, \mathbf{x}_j)$ is the correlation function between a pair ($\mathbf{x}_i$, $\mathbf{x}_j$) at n design runs ($\mathbf{x}_1, ..., \mathbf{x}_n$). The correlation function $R(\mathbf{x}_i, \mathbf{x}_j)$ has several forms. In this study we use a common form, the Gaussian correlation. The Gaussian correlation function for p input parameters can be expressed as:

$$R(\mathbf{x}_i, \mathbf{x}_j) = \prod_{k=1}^{p} exp(-\theta_k |x_{ik} - x_{jk}|^2) \tag{2.36}$$

where $\{\theta_k > 0 : i = 1, \ldots, p\}$ are unknown correlation parameters to fit the model.

The Kriging predictor, $y^s(\mathbf{x})$ of the response y($\mathbf{x}$) at a testing input parameter $\mathbf{x}$ is expressed as:

$$y^s(x) = \hat{\boldsymbol{\beta}} + \mathbf{r}^T(\mathbf{x})\mathbf{R}^{-1}(\mathbf{y} - \mathbf{f}\hat{\boldsymbol{\beta}}) \tag{2.37}$$

where $\mathbf{y}$ is a vector with length n which comprises the corresponding response of n design runs (training input parameter data). Here $\mathbf{r}(\mathbf{x})$ is an $n \times 1$ correlation vector, defined as $\mathbf{r}(\mathbf{x}) = [R(\mathbf{x_1}, \mathbf{x}), \ldots, R(\mathbf{x}_n, \mathbf{x})]^T$ where $\mathbf{x_1}, \ldots, \mathbf{x}_n$ are vectors of training input parameter data and $\mathbf{x}$ is a vector of testing data; $\mathbf{f}$ is a column vector of ones when $f(\mathbf{x})$ is considered to be a constant [11,175]. The two terms on the rhs in Equation (2.37) are not correlated, and the second term on the right can be interpreted as a smooth adjustment of the residuals.

The parameter estimates $\hat{\boldsymbol{\beta}}$ (Equation 2.37), $\hat{\sigma^2}$ (Equation 2.35), and $\hat{\theta}$ (Equation 2.36), can be obtained from the log-likelihood function for the Gaussian process, given by:

$$-\frac{1}{2}[nln(\sigma^2) + ln|\mathbf{R}| + (\mathbf{y} - \mathbf{f}\beta)^T\mathbf{R}^{-1}(\mathbf{y} - \mathbf{f}\beta)/\sigma^2] \tag{2.38}$$

By calculating the derivative of Equation (2.38), with respect to β and σ^2, we find that the Maximum Likelihood Estimator (MLE) of β:

$$\hat{\beta} = (\mathbf{f}^T\mathbf{R}^{-1}\mathbf{f})^{-1}\mathbf{f}^T\mathbf{R}^{-1}\mathbf{y}, \tag{2.39}$$

and the MLE of σ^2

$$\hat{\sigma^2} = \frac{1}{n}(\mathbf{y} - \mathbf{f}\hat{\beta})^T\mathbf{R}^{-1}(\mathbf{y} - \mathbf{f}\hat{\boldsymbol{\beta}}) \tag{2.40}$$

and Equation (2.38) becomes:

$$-\frac{1}{2}(nln\hat{\sigma^2} + ln|\mathbf{R}|). \tag{2.41}$$

The MLEs of θ_k where $k = 1, \ldots, p$ in Equation (2.36) can be obtained by maximizing Equation 2.41. However, this is complex, requiring multivariate optimisation. Welch et al. [184] proposed an algorithm to estimate Kriging model parameters. The detailed steps of this algorithm are described by Na-Udom [11].

The multidimensional Kriging model has been well recognized in the engineering field. Zhaoyang et al. [10] used Kriging to model cantilever beam structure and leading edge stiffener for aircraft. A Kriging model was also used as a surrogate model for an optimisation problem on the injection system of automotive compressed natural gas engines [52] and in the process of clinching joint material [62].

2.7.2 Artificial Neural Network

The Artificial Neural Network (ANN) method is inspired by the successful parallel working of neurons in the brain. How do these neurons work in brain? This can be described as follows. Each neuron transfers electrochemical signals to the connected neurons. Incoming signals are collected by dendrites via synapses which then transfer electrical information to the nuclei of the neurons. This information, comprising activating and inhibiting signals, is accumulated by a nucleus and as soon as a threshold value is exceeded the nucleus transmits an electrical voltage along the axon to the neurons connected to it. Figure 2.15 illustrates the components of neurons.

In ANN, artificial neurons are structured into several layers. The connections between neurons in subsequent layers is provided by synapses. The input layer contains nodes related to the number of predictor variables, while the output layer has nodes containing responses. The layers between the input and output are layers known as hidden layers. Input and hidden layers also take into account a constant node as intercept synapses which this is not directly related to the predictors, and is known as bias. Figure 2.16 describes two pictures of ANN: one is the simplest ANN model, with p neurons in the

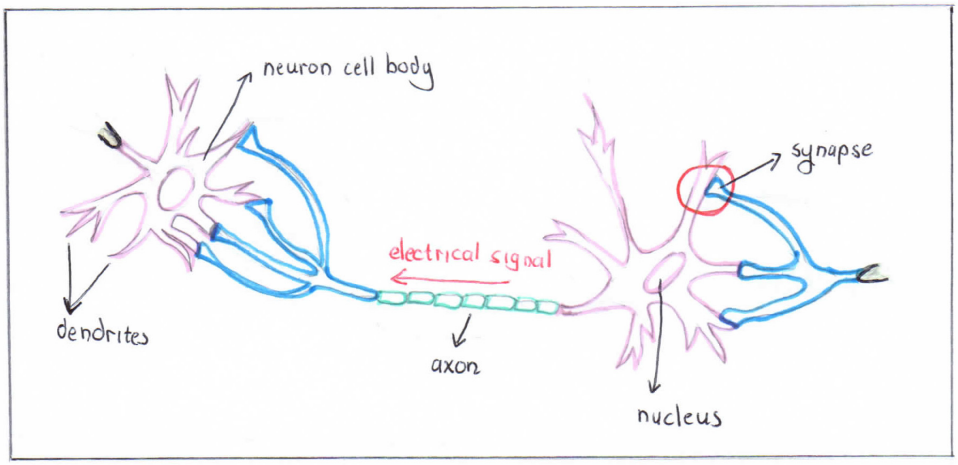

Figure 2.15: Neuron diagram

input layer and one neuron in the output layer. The other model includes three neurons in the hidden layer, two neurons in the input layer and one neuron in the output layer. The neurons in subsequent layers are connected by sypnases.

Each synapse is initially assigned random weights, usually sampled from a normal distribution. A vector of input parameters in $\mathbf{x}$, are then processed into the ANN in two steps, namely passing the signals through the integration function, then converting them into the outputs of the nodes using the activation function.

For the simplest ANN (left picture of Figure 2.16) the integration and activation functions can be expressed as:

$$v = w_0 + \sum_{j=1}^{p} w_j x_j \tag{2.42}$$

$$o(\mathbf{x}) = A(v), \tag{2.43}$$

where x_j denotes a neuron or an input parameter in input layer j with $j = 1, \ldots, p$. w_0 is the intercept (linking 1 to the output neuron) and w_j is a sypnatic weight linking x_j in the input layers with the output neuron. $o(\mathbf{x})$ is the output of the ANN model and $A(\cdot)$ is an activation function.

For the more complex ANN model, with p neurons in the input layer, one hidden layer with q neurons and an output neuron, the integration and activation functions are formulated as:

$$v = w_0 + \sum_{j=1}^{q} w_j A(w_{0j} + \sum_{i=1}^{p} w_{ij} x_i) \tag{2.44}$$

$$o(\mathbf{x}) = A(v) \tag{2.45}$$

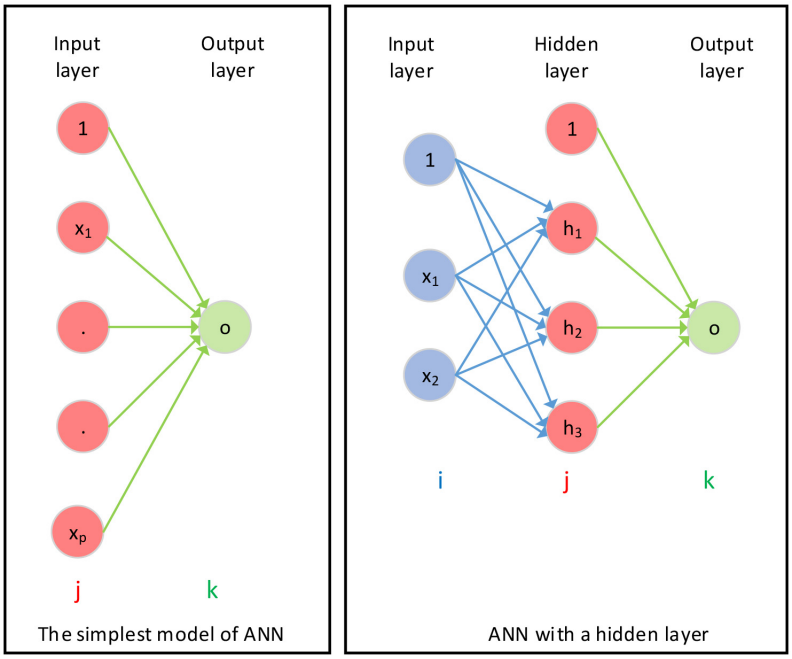

Figure 2.16: Examples of ANN

where p is the number of neurons in the input layers, w_{ij} is a synaptic weight linking x_i in input layers to h_j neuron in hidden layer, w_j is a synaptic weight linking the h_j neuron in the hidden layer to the output neuron; $o(\mathbf{x})$ is the the output of ANN model and $A(\cdot)$ is an activation function.

In general, the relationship between input neurons and output neurons in ANN can be expressed as:

$$o(\mathbf{x}) = A(V(\mathbf{x})), \tag{2.46}$$

where V is integration function commonly defined as :

$$V(\mathbf{x}) = w_0 x_0 + \sum_{j=1}^{p} w_j x_j = w_0 + \mathbf{w}^T \mathbf{x} \tag{2.47}$$

and the activation A(.) usually has some particular characteristics, e.g., a bounded non-decreasing and differentiable function, such as the hyperbolic tangent $A(v) = \frac{2}{1+e^{-2v}} - 1$ or the logistic function $A(v) = \frac{1}{1+e^{-v}}$.

A vector of predicted outputs with length n ($\mathbf{o}(\mathbf{x})$) is generated by the ANN for a given matrix of input parameters ($\mathbf{X}$) with size $n \times p$ and its corresponding weights. The difference between predicted output and actual training response is commonly measured by the sum of squared errors as expressed in the following equation:

$$E = \frac{1}{2}\sum_{l=1}^{n}\{o(\mathbf{x})_l - y_l\}^2 \tag{2.48}$$

where n denotes number of responses, $o(\mathbf{x})_l$ and y_l denote the l^{th} predicted output and actual training response, respectively.

The ANN model is trained through a learning process, for which the synaptic weights are iteratively updated to minimize the error function. Common learning algorithms that have been widely applied are backpropagation (backprop) and resilent backpropagation (Rprop). We present descriptions of both algorithms in the following section.

Backpropagation

Backpropagation is based on a gradient descent algorithm to find the local minimum of the error surface. In gradient descent methods, the weights are adjusted in the opposite direction of the partial derivative of the gradient of the error surface with respect to the weights ($\frac{dE}{d\mathbf{w}}$). When the gradient of the error surface is negative, the value of the weight is increased. On the contrary, the weight is reduced when a positive sign is provided by gradient. Figure 2.17 illustrates the concept of the gradient descent method for finding the local minimum for single weight parameter.

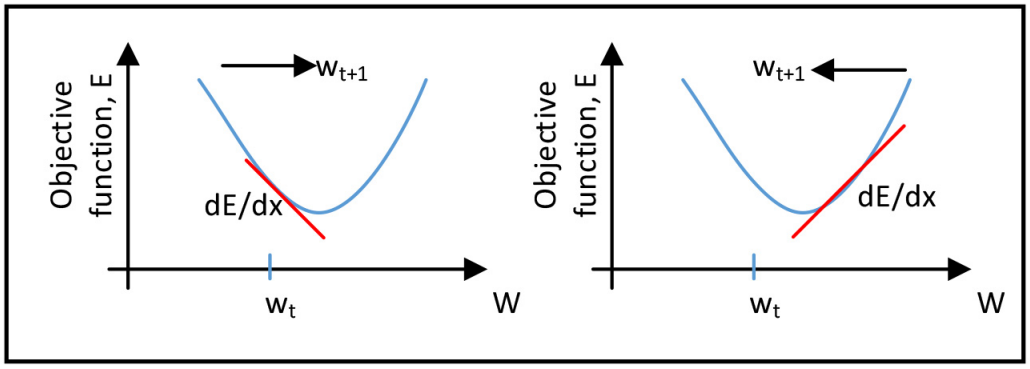

Figure 2.17: The update of a parameter in gradient descent method

The derivative of the integration and activation functions of the ANN using gradient descent to update the weights is performed by using the chain rule for partial derivatives. The detailed information can be obtained from Rojas [149].

A common problem usually found in the gradient descent technique is a non-uniform size of the derivative (gradient) across the learning process. This means that the weights which are located far from the output layer, for instance near the input layer, receive slower learning/improvement due to the limited influence of slope provided by the activation function [121]. To overcome this problem Miller proposed an alternative algorithm, resilient backpropagation which will be described in the next section.

Resilient backpropagation algorithm

Resilient Backpropagation

Resilient backpropagation, or Rprop, uses the sign provided by the gradient of the error function $\frac{\partial E}{\partial w}$ rather than the magnitude of the gradient of the error function as performed by the backpropagation algorithm. This sign is used to provide the direction of the weight updates [121]. The size of weight updates, meanwhile, are controlled by Δ_{jk}. When the sign of two consecutive partial derivatives of the error function with respect to the weight remain the same, the Δ_{jk} is increased by the learning rate of η^+ to speed up the convergence in shallow regions and weight. However, when the two consecutive partial derivative change in sign, it indicates that the last adjustment is too large and the Rprop skips the local minimum, the Δ is reduced by the learning rate of η^-, and the current weight returns to the previous weight prior the last adjustment. The weight w_{jk} is then updated according to the opposite sign of the current gradient error with respect to corresponding w_{jk} multiplied by Δ_{jk}. The steps are described in the following Algorithm 1. Detailed work flow of the Rprop algorithm can be obtained from [123].

Algorithm 1 Resilient Backpropagation Algorithm

for all weights, w_{jk} **do**
  if $\frac{\partial E(t-1)}{\partial w_{jk}} * \frac{\partial E(t)}{\partial w_{jk}} > 0$ **then**
    $\Delta_{jk}(t) = minimum(\Delta_{jk}(t) * \eta^+, \Delta_{max})$
    $w_{jk}(t) = w_{jk}(t) - sign(\frac{\partial E(t)}{\partial w_{jk}}) * \Delta_{jk}(t)$
    $\frac{\partial E(t-1)}{\partial w_{jk}} = \frac{\partial E(t)}{\partial w_{jk}}$
  else if $\frac{\partial E(t-1)}{\partial w_{jk}} * \frac{\partial E(t)}{\partial w_{jk}} < 0$ **then**
    $w_{jk}(t) = w_{jk}(t) + sign(\frac{\partial E(t-1)}{\partial w_{jk}}) * \Delta_{jk}(t)$
    $\Delta_{jk}(t) = maximum(\Delta_{min}, \Delta_{jk}(t) * \eta^-)$
    $\frac{\partial E(t-1)}{\partial w_{jk}} = 0$
  else
    $w_{jk}(t) = w_{jk}(t) - sign(\frac{\partial E(t)}{\partial w_{jk}}) * \Delta_{jk}(t)$
    $\frac{\partial E(t-1)}{\partial w_{jk}} = \frac{\partial E(t)}{\partial w_{jk}}$
  end if
end for

ANN has been widely applied in many fields of study, such as science and engineering [16, 43, 125, 148], economics [102], medical science [204]. Applications of surrogate models in modelling rainfall-runoff processes are listed Table 2.3.

Table 2.3: Several applications of surrogate models in RRMs

Model input	response	model	catchment	ref.
7-8 inputs current and antecedent flow	hourly runoff at $t = 12$ hr	ANN with cascade correlation	River Trent catchment UK	[33]
18 inputs current and antecedent rainfall	daily runoff at t	ANN multiple linear regression	Sungai Liu Malaysia	[165]
6-38 inputs current and antecedent rainfall	daily runoff at t	ANN	7 catchments India, US Ireland, Colombia Japan, Philippine	[127]
15 inputs current and antecedent rainfall antecedent runoff	daily runoff at t	ANN	Ourika Basin Morocco	[170]
16 site descriptors	10, 20, 30 year flood events	ANN	Britain, Northern Ireland Isle of Wight	[36]
5 inputs, current and antecedent rainfall data (at $t, t-1, t-2$) antecedent runoff (at $t-1, t-2$)	daily runoff at t	ANN	Kentucky River US	[172]
up to 4 inputs, antecedent runoff ($t-1$; $t-1$ and $t-2$; $t-1, t-2$ and $t-3$; $t-1, t-2, t-3$ and $t-4$)	monthly runoff at t	ANN	Colorado River US	[8]
15 inputs Rainfall data for 5 gauging station	daily runoff	ANN	Kangsabati reservoir catchment, India	[3]
6 input of soil	pesticide leaching	Kriging	Europe river	[76]
up to 17 inputs antecedent rainfall, runoff,baseflow	daily runoff	ANN	Anseghmir river Morocco	[126]

2.8 Concluding remarks

HM is an important process in the computer-based modelling paradigm. Input parameters of a model are calibrated to develop a reliable model which is then used for some purpose, such as forecasting model response, studying natural behavior, patient treatment, etc. Numerous HM methods have emerged to solve HM problems since their advent in 1960. These methods are classified into single and multiple solution-based methods. In this study, we proposed HM methods from both classes, namely one single method (GLM) and three multiple solution-based method (ACO_R, $DACO_R$, and ROPE). The selection of these algorithms was based on consideration as follows:

(i) ACO has been one of the famous multiple solution approaches to solve optimization problems. The performance of ACO_R were comparable with other well known algorithms, and in some applications ACO_R demonstrated even better performance than other algorithms.

(ii) GLM has been used widely by the hydrology community. This algorithm showed comparable accuracy with other methods and provided an advantage in computational time.

(iii) The ROPE algorithm has been shown to have comparable performance with other algorithms and has been used for optimisation some RRMs.

We applied two HM procedures in this study, namely direct HM and indirect HM. In direct HM, the RRM is directly used to provide model responses and, subsequently, is used for calculating the objective function measuring the gap between modelled and observed response, which we seek to minimise using HM methods (GLM, ACO_R, $DACO_R$ and ROPE). In indirect HM, we introduced surrogate models for the RRM in providing the model response. Two statistical models chosen were multidimensional Kriging and ANN. All models have been well recognized and successfully applied in many modelling fields. Indirect HM is performed by GLM, ACO_R, $DACO_R$ and ROPE using the outputs of surrogate models.

In the next chapter we describe the hydrological model used for this study. The results and discussion of direct HM and indirect HM will be provided in Chapter 4.

CHAPTER 3

Hydrological Modeling Background, Study Areas and the LUCICAT Model

3.1 Introduction

This chapter presents the background on hydrological modelling, the study catchments explored in this study and the Land Use Change Incorporated CATchment (LUCICAT) model. The study areas explored are The Dandalups, Bates and Lewis catchments. All these catchments lie in the the Peel region of Western Australia between Serpentine and Pinjarra (about 90 km south of Perth). Bates and Lewis are experimental catchments within The Dandalups. The Dandalups is protected water resource catchment since the two reservoirs within this catchment supply about 20% Perth' annual water consumption [187, 188]. The Dandalups area is also rich in mineral resources (bauxite in particular). Therefore, managing water resources in The Dandalups is vital to maintain sustainability of future water resources in terms of both quantity and quality, while maintaining extraction of valuable mineral resources. Lewis is a bauxite mining area, while Bates is a control catchment developed for understanding the impact of mining activities on the catchment.

We apply the LUCICAT model [108] to mimic the natural rainfall runoff system in these catchments. The LUCICAT model is a distributed conceptual hydrological model that propagates rainfall into final streamflow and salinity into a water reservoir. The model was developed to represent the impact of land use and climate change on streamflow and salinity. This models has been used by decision makers to manage water resources in Western Australia [106]. By modelling The Dandalups, Bates, and Lewis, all interested parties (hydrologists, government agencies, mining company) are able to understand the impact of land use, such as mining, forest thinning and rehabilitation, and climate change on catchment water yields in The Dandalups and Bates, and water quantity (runoff) and quality (salinity) in Lewis. The outcomes of modelling scenarios (mining, rehabilitation and/or climate scenarios) of Bates and Lewis may influence management options for the whole Dandalups catchment.

The objectives of this chapter are:

(i) To describe the rainfall runoff generation processes and hydrological modeling.

(ii) To describe the background of the LUCICAT model and model parameters that play important roles in the generation of streamflow and total salt.

(iii) Describe the characteristics of the catchments explored in this study.

(iv) To describe the application of LUCICAT to the study catchments.

3.2 Hydrological Processes

3.2.1 Hydrological Cycle

The Hydrological cycle, also known as the water cycle, is a continuous process of water movement between the atmosphere, earth's surface and subsurface. The hydrological cycle is illustrated in Figure 3.1. The sun, as the main driver in the cycle, emits energy to evaporate water from the earth's surface, shallow groundwater wicks, and water bodies (oceans, lakes, rivers, etc). Plants also release water to the atmosphere through transpiration. The water vapours rise in the atmosphere and condense forming microscopic droplets of water in clouds. These tiny droplets join together to form bigger drops and when they are heavy enough, they return to the earth as precipitation in the form of rain, snow or hail. Several hydrological processes occur following precipitation. Some water may be intercepted by vegetation. Some of it may flow across surfaces as surface runoff, and the remaining water may infiltrate to groundwater, eventually transferring back to water bodies. Finally water is released back to the atmosphere through evaporation and transpiration.

3.2.2 From Precipitation to Streamflow

In a catchment, precipitation as an input undergoes several processes to form the output of streamflow. Streamflow, described as the water flow in channels, is also known as runoff, stream discharge, or catchment yield [106]. In a catchment, rainfall events occur for some duration, and are spatially distributed. Conversely, streamflow is a continuous output series, providing rapid flow following rainfall events and relatively steady discharge throughout the dry season. Based on this fact, Ward and Robinson [145] suggested there must be a nonlinear relationship between rainfall inputs and streamflow outputs.

Streamflow is generally formed by three components, namely surface flow, interflow and baseflow [144]. Surface flow is water flow on the earth's surface due to saturated soils. Interflow is water flow under the soil surface. Baseflow is water discharge from groundwater to a stream channel. The processes of generating streamflow may be classified as quickflow and slowflow. Quickflow is characterized by a rapid process between the precipitation event and streamflow generation. Surface flow and interflow may be categorized as quickflow. Meanwhile, under slowflow, a continuous stream of water is provided through dry periods. Slowflow may include interflow and baseflow.

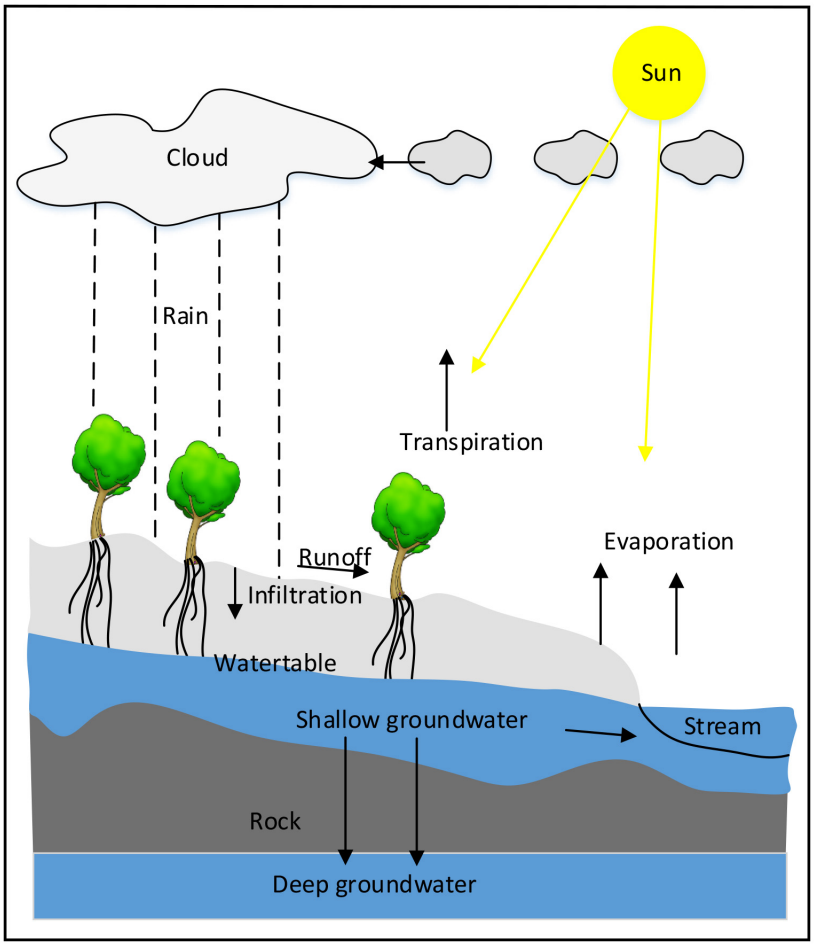

Figure 3.1: Hydrological cycle

The soil profile underneath the surface can be classified into the unsaturated and the saturated zones, connected with the capillary fringe. The unsaturated zone, where the soil pores are not entirely filled with water, is located in the upper layer of soil. By contrast, the saturated zone containing soil pores full of water is located beneath the water table. This saturated zone is known as groundwater.

When rainfall reaches the ground, some fraction creates surface runoff and the remainder infiltrates to the soil matrix. The surface runoff emerges due to the capacity of water storage under the surface being exceeded. Prolonged rainfall, regardless of intensity, leaving no empty soil pores, leads to a saturated condition. This inhibits infiltration of surface water leading to the generation of infiltration excess runoff. The rate of infiltration depends on several factors, such as water content in the soil, the soil's hydraulic conductivity, the viscosity of water, the distribution of pore size, surface roughness and tortuosity of the path between pores [146].The infiltrated water may be evaporated from the upper layer of soil, transpire back by vegetation to the atmosphere, percolate downward or may result in interflow.

Interflow is a lateral movement of water under the soil surface. This occurs when the soil

permeability limits the amount of water that flows downwards. Interflow is contributed from shallow and intermittent groundwater following rainfall recharge, that often directly discharges to streams. Interflow can be classified as a quickflow or slowflow mechanism. This classification depends on the geological characteristics of the catchment. With high permeability of soils, and steep slopes or catchment topography, interflow may become quickflow. In other types of geology, where a time-lag of streamflow generation may be present, interflow may contribute to prolonged duration of streamflow after rainfall.

Percolation is the movement of water downward from the unsaturated zone into the saturated zone. This leads to water recharge of shallow and deep groundwater. Groundwater flows into the stream channel that intersects the saturated zone, creating baseflow. Baseflow maintains streamflow during long dry periods between rainfall events.

3.2.3 Input Factors of Streamflow Generation

The input factors which influence streamflow generation can be classified into climatic and catchment characteristics. Rainfall, evaporation and temperature are components of climatic inputs. The frequency, duration and distribution of rainfall determines the amount of rainfall to infiltrate below the surface, and the excess of water for runoff. Increased temperature increases the water holding capabilities of the atmosphere, leading to an increase in evaporation and transpiration. This results in a reduction in catchment yield.

Three major components that characterise a catchment in relation to streamflow generation, are topography, geological structure and vegetation. Topography entailing elevation, slope, location of lakes, channels of a catchment contributes to the knowledge of flow direction and aggregation at stream channels within a catchment. Geological structures include top soil depth, types of soil, the depth of the water table affecting the infiltration and percolation rates. Meanwhile, vegetation effects the amount of rainfall intercepted before it reaches the groundwater, and also the transpiration rate. Both interception and transpiration are influenced by the types of vegetation, measured via Leaf Area Index (LAI). LAI is the ratio of one-sided green leaf area to ground surface area.

3.2.4 Dryland and Stream Salinity

Stream and land salinity have become key environmental problems in Australia. Of all the States, Western Australia has the greatest salt-affected land area, contributing 80% towards the national salinity problem [133]. The generation of dryland and stream salinity is closely related to the hydrological processes.

The Australian landscape has large deposits of salts in the soil column. These salts came from the ocean, evaporated into the atmosphere and fell on the ground surface within rainfall and dust. Over long processes of evaporation, transpiration and weathering, saltfall has infiltrated and percolated to the soil profile and deposited in the unsaturated and saturated zones. The concentration of salt contained in the upper groundwater zone is relatively low due to high flow

that removes most of the salt in soils. Unlike in the upper groundwater zone, the flow in the lower groundwater zone is slow, resulting in slow mobilisation of salt and thus generating high salinity in an aquifer [144].

The replacement of deep-rooted vegetation with shallow-rooted annual agriculture crops and pasture species has led to an increase in saline aquifer recharge and, eventually, has resulted in rise in the groundwater table. When the groundwater table rises, it dissolves salts stored in soils and gradually brings salts to the surface soil. Salinization at the surface, eventually, has created dryland salinity. A similar process also occurs in stream salt generation. A rise of the groundwater table increases baseflow, so increasing saline water in streamflow. In addition to salt intake from baseflow, streamflow receives salt from surface runoff from a saline catchment. The concentration of salt in rivers tends to be high during the dry season due to less dilution by fresh runoff. Meanwhile, in the wet season the rising water table increases saline groundwater discharge and thus an increase in stream salinity.

3.2.5 Hydrological Modelling

The need for managing water resources due to population growth, rapid urbanisation, increasing agricultural activities and climate change inspires scientists to study the natural rainfall-runoff system. This is done by developing models for delineating the rainfall and runoff relationships. These models, referred to as hydrological models, are mathematical equations describing the hydrological processes and a time series of response balances. With hydrological models, water management planners may be able to make decisions regarding management of water resources and land use.

The oldest and simplest rainfall-runoff model was developed by Mulvaney in 1851 [176]. This model, known as the Rational method, was aimed at calculating peak discharge (Q) from a catchment as follows:

$$Q = ciA \tag{3.1}$$

where c, i and A are the rational method runoff coefficient, rainfall intensity and catchment area respectively. This simple model, although not very accurate, forms the basis for more complex models. In the era of rapidly growing computing technology over the past 50 years, the complexity of hydrological models has increased to more realistic mathematical forms for hydrological processes.

The structure and application of these models can be described by the following mechanisms:

(i) The basic algorithms used in the models.

(ii) The spatial representation, i.e. lumped versus distributed models.

(iii) The"downward" or "upward" approaches in models.

(iv) The stochastic or deterministic approaches for model output.

In terms of the basic algorithms, there are three types of approaches which can be adopted, namely empirical, conceptual and process-based models. Empirical models, also known as

"black-box" models, are based on the relationships between inputs and outputs without any attempt to describe the processes behind this relation. Conceptual models consider the processes in the catchment, water storage capacity of soils, vegetation, groundwater, stream channels, lateral flow, flow aggregation and the presence of evapotranspiration. Under this setup, the theoretical model structure is specified and model parameters are calibrated until reasonable matching with the historical responses is found. Unlike the previous classified models, process-based models are built based on the fundamental principles of physics implementing equations for water flow over and through soil and vegetation. As the process-based models typically use a computational grid for which each grid cell represents physical hydrological processes, the models are able to provide spatial distribution of climate conditions and physical parameters, such as soil porosity [1]. However, the process-based models are computationally expensive compared to the conceptual models and require large input data on a gridded basis [68]

The next model mechanism is based on spatial representation, i.e., lumped versus distributed models. Lumped models treat a catchment as a single unit and average the values of rainfall, evaporation, runoff and infiltration over the unit. These models are suitable for catchments with homogeneous areas. By contrast, in distributed models, a catchment is divided into discrete units allowing spatial variability. The distributed models can include several input factors, such as land use and climate changes that lead to improve forecasting [174].

Hydrological models can be developed through a downward or upward approach. In the downward approach, known as a "top-down" model, the model is built from large spatial and temporal scales to gradually finer scales [180]. In contrast to downward approaches, upward approaches or "bottom-up", implement laboratory scale models at larger scales [180].

Finally, the results of the models are generated in deterministic or stochastic manners. Deterministic approaches uses a single set of input parameter values to provide a single set of output. Most hydrological models use deterministic approaches. Unlike deterministic approaches, stochastic approaches rely on the statistics of observations and a range of input and output values. This model then produces a probability distribution for model responses.

3.3 Study Areas

In this thesis we explore three catchments, namely The Dandalups, Bates and Lewis catchments. Description of these catchment are provided in the following sections.

3.3.1 The Dandalups

The Dandalups is located 90 km south of Perth, Western Australia. The catchment is about 700 km^2 with most of the area on the Darling Scarp. The elevation varies between 10 AHD (Australian Height Datum) and 500 AHD, with average slope ranging from 1.2% to 16.2% . The outlet of the catchment is to flat land. The soil types within the catchment are mainly duricrust, sandy soils and gravels.

Figure 3.2 depicts the location of The Dandalups, surface water bodies and gauging stations (some of them belong to the Department of Water). The catchment has two reservoirs, the North and South Dandalup reservoirs. In this region three water streams, namely the North Dandalup river, the South Dandalup river and Conjurunup Creek flow into the Dandalup River and eventually to the Murray River (to the west). The two reservoirs are among eight dams that supply surface water for the city of Perth, providing 20% of Perth's annual water consumption [187, 188]. The surface water contributes 33% of total water consumption (groundwater and desalination supply 40% and 27% respectively) to the Perth Metropolitan Region [35]. North Dandalup, which has a capacity of 75 GL, contributes 10% of Perth's annual water supply (19 GL) [187]. South Dandalup, with capacity of 208.2 GL, provides 17.9 GL per annum of water for consumption in Perth [188].

The catchment has a Mediterranean climate characterized by a hot and dry summer season, with occasional cyclonic storms, and a mild, wet winter season. The mean annual precipitation is about 1,100 mm (based on the period 1960-2007). Approximately 80% of annual rainfall occurs during the winter season (May-October). Seasonal inundation occurs in less than 5% of the catchment. Around a quarter of the catchment (the western area) is at high risk of phosphorus leaching to water channels. The western part of the catchment has been cleared for mainly agriculture activities, such as stock grazing. In contrast areas in the eastern part of the catchment remain relatively undisturbed. Most of the remaining forest is dominated by Jarrah (*Eucalyptus marginata*), Marrri (*Eucalyptus calophylla*) and Wandoo (*Eucalyptus wandoo*) species.

The Darling Plateau, encompassing the Dandulups region, experiences low catchment yields due to the high rate of evapotranspiration in forest within the catchment [82, 129], and low rainfall since 1975 [34].

In addition to water supply, The Dandalups contain bauxite mines. Alcoa World Alumina Australia (Alcoa) has held the license to operate bauxite mining since 1963, under agreement with the State Government. However, as the area has limited water supply, Alcoa is required to maintain the catchment water yields similar to those generated prior to commencement of mining activities. Considering that The Dandalups is an important area, it is essential to maintain water resources management plans that sustain the quantity and quality of the catchment beyond mining activities.

3.3.2 Bates and Lewis Catchments

The Bates and Lewis catchments are located within The Dandalups region. Bates catchment has an area of 2.23 km^2, with elevation varying from 255 to 355 AHD. The average slope is 0.059. The catchment has annual mean rainfall of 1130 mm. The area of Lewis catchment is slightly smaller than Bates at 2.01 km^2. Lewis elevation varies from 272 to 356m with average slope of 0.074. The Lewis catchment also reports the same annual mean rainfall (1130 mm).

These two sub-catchments are highlighted by Government agencies and ALCOA as being of significant interest as they intend to explore the impact on water quantity and quality of adjacent water bodies due to mining activities.

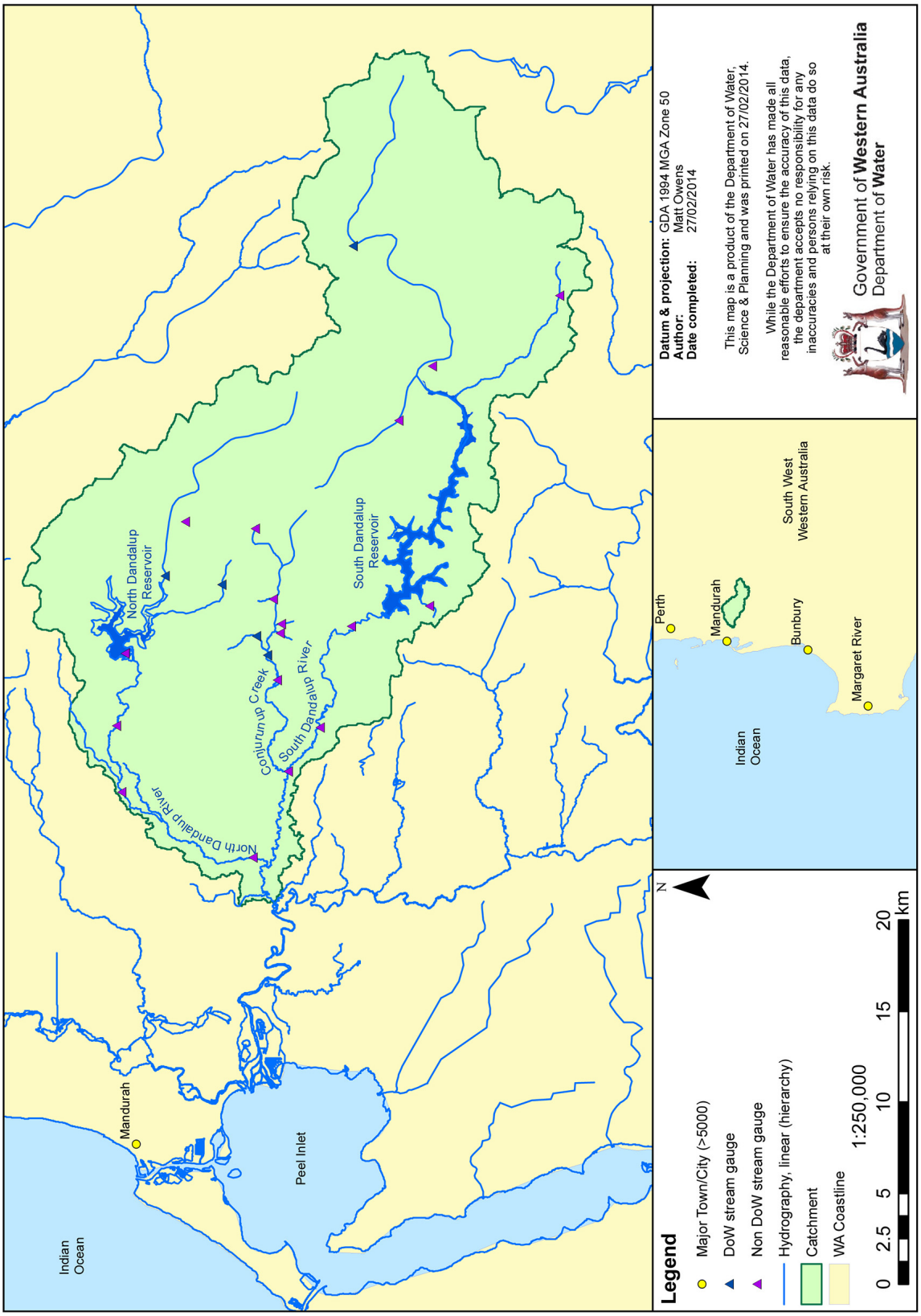

Figure 3.2: The Dandalups [122]

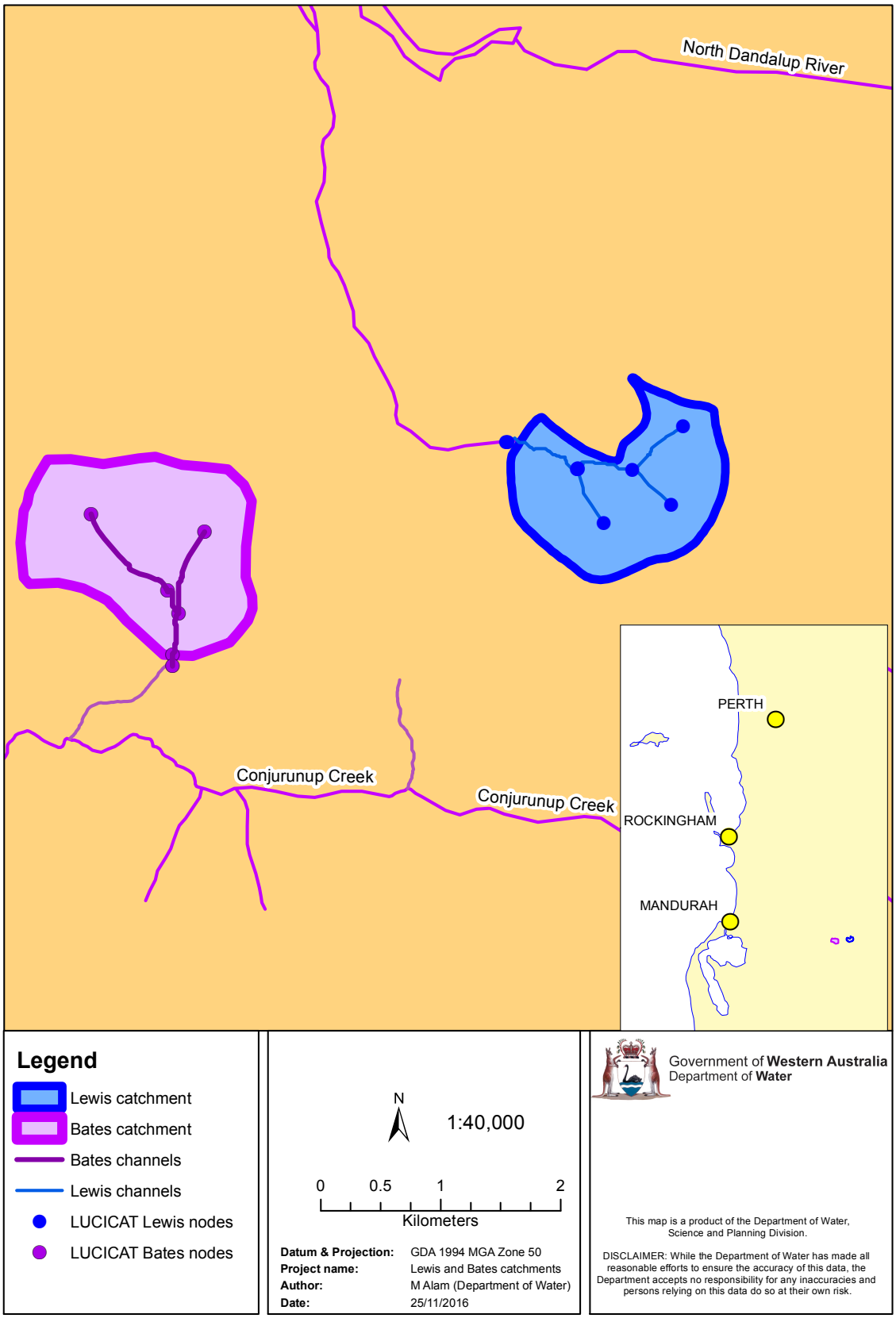

Figure 3.3: Bates and Lewis catchments [128]

3.4 The LUCICAT Model

The LUCICAT background, preparation for the three catchments and model parameter for calibration are described in the following sections.

3.4.1 Background on the LUCICAT Model

The Land Use Change Incorporated CATchment model (LUCICAT) was developed by Bari [108] for the Department of Water to model daily water and salt balances as influenced by land use and climate changes. This model mimics the natural hydrological and geological processes and properties of catchments. The model contains complex physical processes described using mathematical equations, based on pragmatic assumptions and laws of known Physics and estimated input parameters. Figure 3.2.1 depicts the interface of LUCICAT Live.

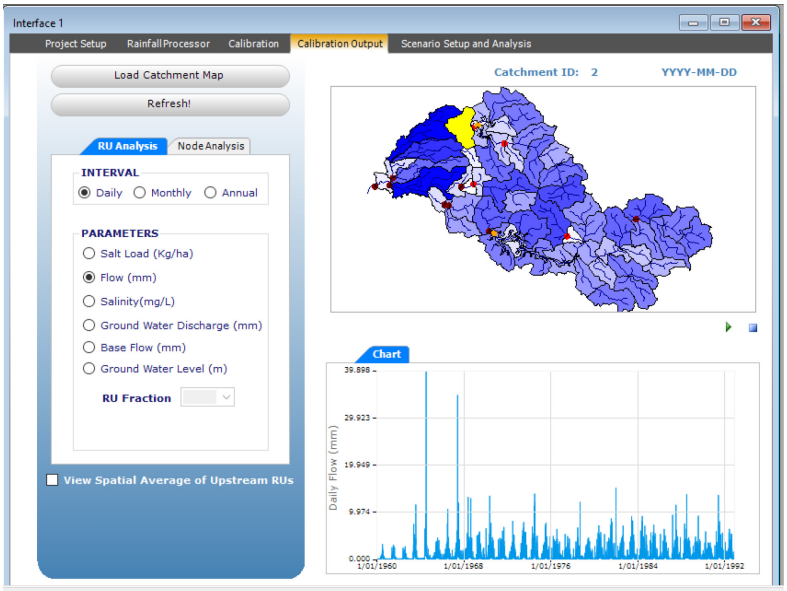

Figure 3.4: The LUCICAT live interface [112]

The LUCICAT model was built using a downward approach, taking large spatial and temporal scales and then complexity was added to describe the finer scales [107]. The model is a distributed conceptual model [112] considering physical processes to transform rainfall to streamflow and to calculate the salt movement in the soil profile and streamflow.

The LUCICAT model has been specifically adjusted to the condition of Western Australian catchments, and applied in small to large catchments. Among these are the Wight and Salmon catchments of the Collie River catchment [107], the Ernies and Lemon sub-catchments of the Collie River catchment [108], the Wungong water supply catchment [109], the Stirling River [120], Kent River [110], Serpentine River [9] and Ord River [111] catchments. For these catchments the LUCICAT model was used for predicting catchment behaviour and streamflow due to land use change, deforestation and/or climate change.

3.4.2 Theory behind LUCICAT

In the LUCICAT model, a catchment is split into several subcatchments known as Response Units (RUs). Each RU represents one building block of the model. Bari and Smettern [107] described the building block as an open book approach as can be seen in Figure 3.5. In the building block model there are five interconnected stores, namely the Upper Store (Dry and Wet Stores), Subsurface Store and the Groundwater Store. These stores are responsible for generating the streamflow. Figure 3.5 depicts the hydrological processes in each building block. Some of the rainfall is intercepted by vegetation leaves and the rest infiltrates to the Upper Store. In the Upper Store, surface runoff (Q_{r1}) and interflow (Q_i) are generated. The remaining water in the Upper Surface percolates to the Subsurface Store (I) and then penetrates deeply to the Groundwater store. When the groundwater level reaches the stream bed due to clearing activities, groundwater forms the Streamzone Store, i.e., a transient zone due to induction of groundwater and generates baseflow to the stream (Q_b). The Streamzone Store contributes additional surface runoff (Q_{r2}). The model includes evapotranspiration occurring in each store. All discharges (Q_{r1}, Q_{r2}, Q_i and Q_b) create the total generation of streamflow of a RU. All RUs are integrated by streamflow routing control to build a whole hydrological process model for a catchment.

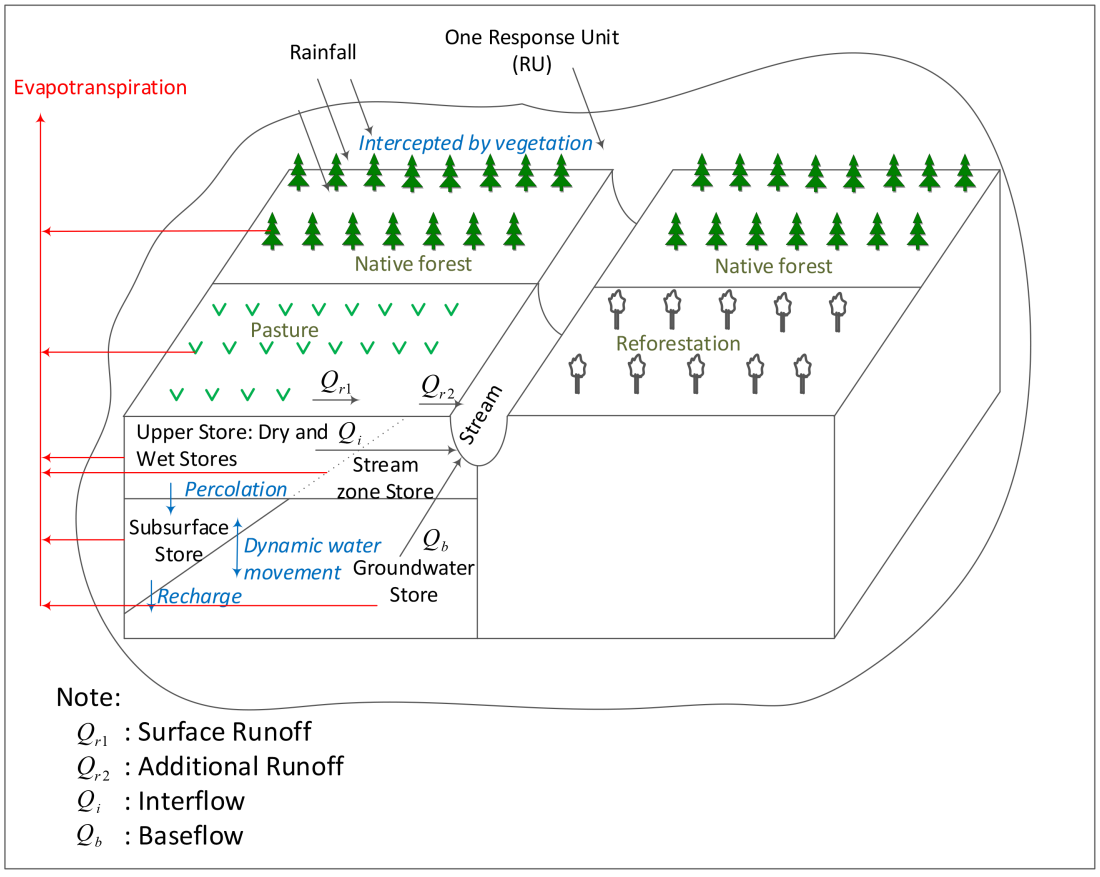

Figure 3.5: LUCICAT building model

Details of the LUCICAT model are presented in [107], [108], and [47]. In the remainder of this section we describe the physical processes modeled in LUCICAT. Equations are provided to describe the role of input parameters that are calibrated in this study. The model description and equations are adapted from LUCICAT manual provided by the Department of Water [47].

The LUCICAT model considers three components of evapotranspiration. The first component is interception of rainfall by vegetation. Interception depends on daily rainfall, Leaf Area Index (LAI) and forest canopy storage [47]. Some salt in rainfall is also intercepted and removed in the subsequent rainfall event. The second component is evaporation from soils, which occurs in the Upper and Streamzone Stores. Evaporation is a function of the moisture content of soil and LAI [47]. Transpiration is the last component of evapotranspiration, which depends on the relative root volume and soil moisture content in different stores, LAI, and residual potential evaporation [47].

The remaining rainfall that reaches the ground is known as effective rainfall (RE, mm). RE can either infiltrate to underneath soil or generate runoff. It is, however, highly possible that water RE penetrates into soil as the Dry Store has high infiltration capacity. The amount of water held in Dry Store (W_d, mm) is determined by field capacity (θ_f) and soil depth (d', mm) and can be expressed as follows: [47]

$$W_d = d'\theta_f \tag{3.2}$$

Some part of RE is retained in the Dry Store and the remainder penetrates to Wet Store (R_f, mm). The equations of these processes are described as follows [47], and there are three cases to consider:

Case 1: If $0 < RE < (b+1)W_{dmx}(1 - \frac{W_d}{W_{dmx}})^{\frac{1}{b+1}}$, then we have

$$Rf = \left(\frac{A_t - A_i}{A_t}\right)\left[RE - W_{dmx} + W_d + W_{dmx}\left\{(1 - \frac{W_d}{W_{dmx}})^{\frac{1}{b+1}} - \frac{RE}{(b+1)W_{dmx}}\right\}^{b+1}\right] \tag{3.3}$$

Case 2: If $RE > (b+1)W_{dmx}(1 - \frac{W_d}{W_{dmx}})^{\frac{1}{b+1}}$, then we have

$$Rf = \left(\frac{A_t - A_i}{A_t}\right)\{RE - (W_{dmx} - W_d)\}. \tag{3.4}$$

Case 3: If $W_d < W_{dmn}$ then $Rf = 0$.

In the above, W_d is water content in Dry Store (mm), W_{dmx} and W_{dmn} refer to maximum and minimum water retention respectively (mm) in Dry Store. A_t is total surface area (mm^2), A_i is impervious area (mm^2) and b is Dry Store soil moisture exponent (dimensionless). In addition to infiltrated water of RE, Dry Store also receives rainfall salt and salt retained in the top soil.

When the moisture content of Dry Store exceeds that of the field capacity, the concentration of salt is released to Wet Store (C_{rf}, mg/l) with chemical processes formulated as [108]:

$$C_{rf} = C_u C_d, \tag{3.5}$$

where C_u is a parameter related to salt release from Dry to Wet Store and C_d is salt concentration of Dry Store (mg/L). The salt transported (S_{rf}, mg/mm^2) in excess water to Wet Store is expressed as [108]:

$$S_{rf} = C_{rf} Rf \tag{3.6}$$

The excess water between field capacity and soil saturation level penetrates down to Wet Store. The Wet Store plays an important role in generating total surface runoff. The total surface runoff is generated by pervious (Q_{r1}, mm) and impervious surface runoff (Q_{r2}, mm) and is expressed as follows [47]:

$$Q_r = Q_{r1} + Q_{r2}, \tag{3.7}$$

where
Case 1: $0 < Rf < (c+1)W_{wmx}(1 - \frac{W_w}{W_{wmx}})^{\frac{1}{c+1}}$, then we have

$$Q_{r1} = [1-(1-\frac{W_d}{W_{dmx}})^{\frac{b}{b+1}}][Rf - W_{wmx} + W_w + W_{wmx}\{(1-\frac{W_w}{W_{wmx}})^{\frac{1}{c+1}} - \frac{Rf}{(c+1)W_{wmx}}\}^{c+1}] \tag{3.8}$$

Case 2: if $Rf \geq (c+1)W_{wmx}(1 - \frac{W_w}{W_{wmx}})^{\frac{1}{c+1}}$, then we have

$$Q_{r1} = [1 - (1 - \frac{W_d}{W_{dmx}})^{\frac{b}{b+1}}][Rf - (W_{wmx} + W_w)] \tag{3.9}$$

where W_w is water content in Wet Store (mm). W_{wmx} is maximum capacity of Wet Store (in mm) and c is the exponent of Wet Store soil moisture (dimensionless). Next we have

$$Q_{r2} = \frac{A_i}{A_t} RE \tag{3.10}$$

The Wet Store also contributes to lateral flow (interflow) generation and vertical flow (percolation) to Subsurface Store, driving the movement of dissolved salt. Interflow is formed when water content in Wet Store exceeds W_{wmn}. The physical process of interflow (Q_i, mm) is expressed by the following equation [47]:

$$Q_i = K_{ul}(\frac{W_w - W_{wmn}}{W_{wmx} - W_{wmn}})^{ia} x, \tag{3.11}$$

where K_{ul} is Wet Store lateral hydraulic conductivity (mm/day), W_{wmn} is a threshold value for interflow generation (mm). ia is interflow exponent (dimensionless) and x is ratio of the pervious area of soil surface which exceeded the field capacity (dimensionless).

The remaining water (I, mm) percolates to the Subsurface Store formulated with the following equation [108]:

$$I = K_{uv}\left\{1 + pb\left(1 - \frac{W_l}{W_{lmx}}\right)^{pa}\right\}\left(\frac{W_w}{W_{wmx}}\right)x, \tag{3.12}$$

where K_{uv} is vertical conductivity of Wet Store (mm/day), pb and pa are parameters (dimensionless) related to vertical soil conductivity. W_l is water holding capacity of subsurface elementary area (mm) and W_{lmx} is maximum capacity of Subsurface Store (mm). The salt concentration of percolated water is less than that contained in Wet Store.

The Subsurface Store has a deep unsaturated soil profile. This store connecting Upper Store and Groundwater Store receives percolated water and salt from Upper Store to recharge Groundwater [107]. The Subsurface Store also gains water and salt from Groundwater when the groundwater level increases. Two mechanisms of groundwater recharge occurring in Subsurface Store are excess flow from the soil matrix and flow created from preferred pathways [47].

Water from Subsurface Store recharges the Groundwater Store. The capacity of groundwater storage depends on the location of the conceptual level of groundwater. When the groundwater level reaches the stream base, groundwater creates induced saturated areas, known as Streamzone Store. The permanent groundwater level also discharges direct flow to the stream through baseflow. Baseflow discharges (Q_b, mm)to Streamzone Store can be mathematically expressed as [47]:

$$Q_b = K_{ll} L |d_s - d_g| tan\beta, \quad d_g < d_s, \tag{3.13}$$

or

$$Q_b = 0, \quad d_g > d_s. \tag{3.14}$$

where K_{ll} is lateral hydraulic conductivity of Subsurface Store (mm/day). L is catchment-wide average stream length (mm). d_s and d_g represent stream depth (mm) and average depth to groundwater level (mm) respectively; $\beta = \alpha/2$, where α is average ground slope in the catchment. Salt contained in Groundwater Store is also discharged to the Streamzone and the stream

Streamzone Store is transient and can expand or contract depending on the amount of water contained in Dry and Wet Store and discharged by Groundwater Store. The RE on this area becomes surface runoff (Q_{r2}). The surface runoff, interflow and baseflow generate a total streamflow (Q_t) of a RU that can be expressed as [108]:

$$Q_t = Q_r + Q_i + Q_b \tag{3.15}$$

The generation of salt in streamflow is influenced by each component of total streamflow. The salt generation in stream highly depends on the concentration of salt in each store and also, the flow occurs in the store. For example, salt is transported to the stream via surface runoff (Q_{r1}) and interflow are expressed [47]:

$$S_{qr1} = C_w Q_{r1} \tag{3.16}$$

$$S_{qi} = C_w Q_i \tag{3.17}$$

where S_{qr1} and S_{qi} are salt load of surface runoff from previous area (mg/mm^2) and salt load of interflow (mg/mm^2) respectively, and C_w is concentration of salt in Wet Store (mg/L). Likewise total streamflow, the total salt in the stream is also combination of salt contained in surface runoff, interflow and baseflow.

3.4.3 LUCICAT Model Preparations for The Dandalups, Bates, and Lewis Catchments

Preparation of input files of LUCICAT models for The Dandalups, Bates, and Lewis catchments was conducted by the Department of Water. The first preparation was to divide a catchment into subcatchments or RUs. The size of RUs depends on the availability of rainfall stations, stream network, topography, soil type and land-use history [47]. Digital elevation models (DEMs) in a GIS package are used to define the possible boundaries of RUs and stream channels network in RUs. These boundaries were edited to provide stream channels and one single outlet in each RU. All RUs were numbered to represent the location of subcatchments in a catchment. The flow direction, the centroid coordinates, rainfall, geological structures and topography of each RU are described in an attribute file (_Atr.dbf). For The Dandalups catchment, the area was divided into 55 RUs with area varying between 3.11 and 25.55 km^2 (Figure 3.6). Meanwhile, Bates and Lewis only have one RU each since both catchments are part of The Dandalups with area less than 2.5 km^2.

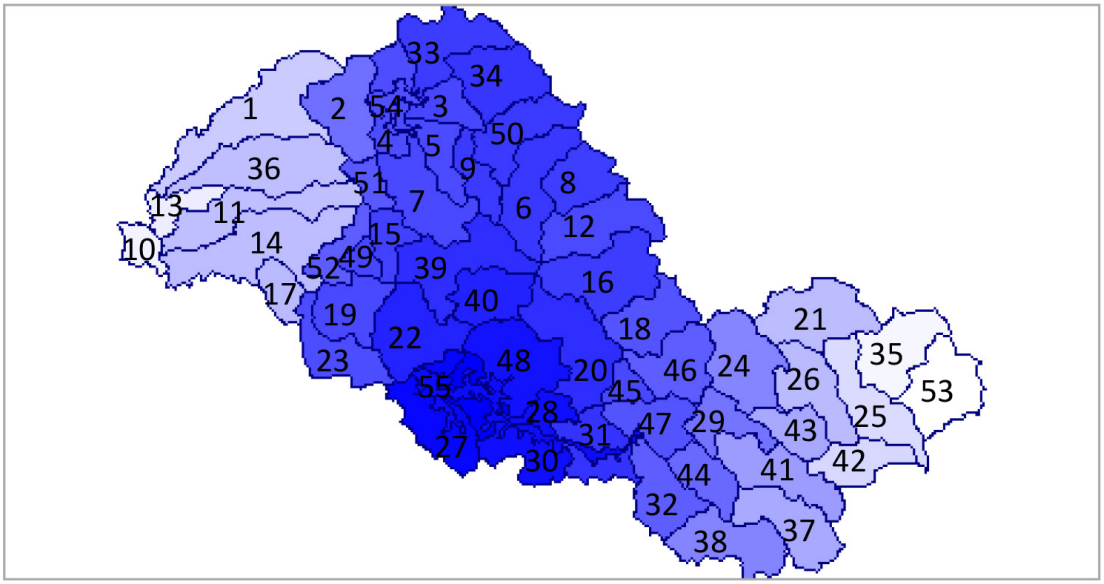

Figure 3.6: Response Units in The Dandalups [112]

The stream channel network was developed after defining the RUs. The river channels were designed to have continuity. These channels were then divided into channel segments. Each end of segment was identified with a node. Nodes were also placed at intersections between

channels, in and out of the lakes, the channels passing through the boundaries of RUs, at a gauging station. The information regarding channels including the width, length, top and bottom nodes and manning coefficients (0.08) was described in a channel file (_Channel.dbf). These processes resulted in 353 nodes for Dandalup catchment and one node each for Bates and Lewis due to small catchment areas. Figure 3.7 depicts some nodes in The Dandalups.

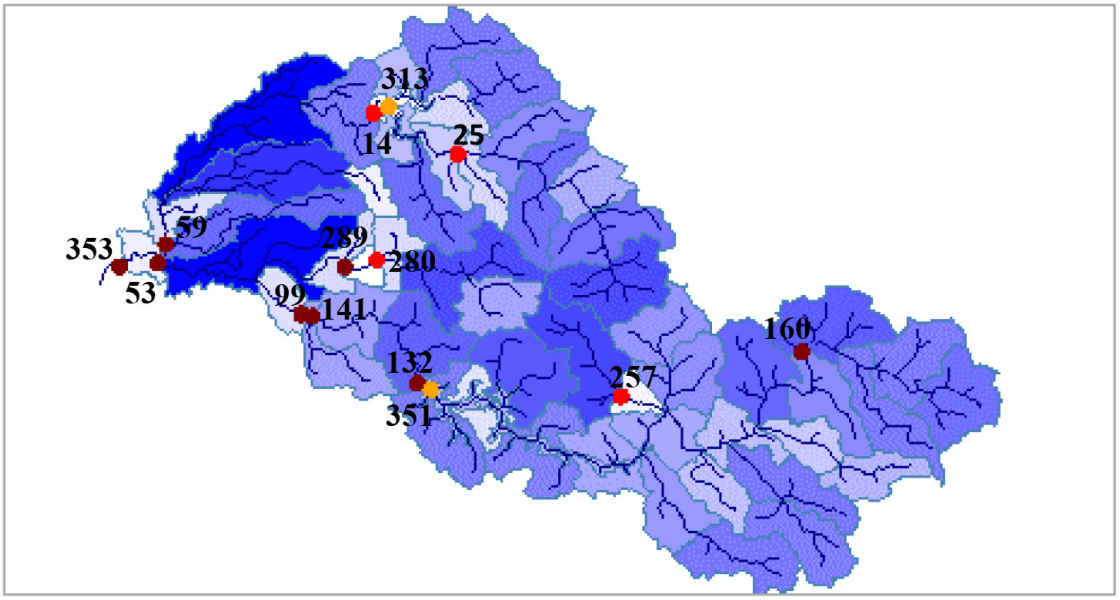

Figure 3.7: Nodes in The Dandalups [112]

The next step was to define the land use in each RU. Land use was divided into functional units which are represented in a percentage of the RU area. The functional units were then split into one or more land use units. The area of each land use unit was described as a percentage of the functional unit area. Each land use unit represented one type of vegetation that varies over time. The LUCICAT model currently has five vegetation options, i.e., native forest, annual pastures and three different perennial pastures. The information on vegetation was listed in a vegetation file (_Veg.dbf). This file included information on LAI over time and the root depth of vegetation.

Global parameters that applied for all RUs were prepared and provided in the lucicat.par file. Among global parameters were the hydraulic conductivity in different stores, exponent parameters for Dry and Wet Store and interflow, a salt release factor, and biological factors for maximum and minimum rainfall.

Other input files that were also prepared are rain.dat and pan.dat to describe rainfall and pan evaporation of each RU and lake_character.dat for lake or reservoir information. The Dandalups contains lake_character.dat to describe North and South Dandalup dams. Meanwhile, Bates and Lewis do not have lake information as there were no lakes found in these catchments.

3.4.4 Calibration of LUCICAT Model Parameters

We focus on six global LUCICAT model parameters for calibration. The choices of model parameters were adopted from the study conducted by the Department of Water [47]. These model parameters, their roles in the model and equation numbers, and the ranges of initial model parameters are listed in Table 3.1. Parameters of ia, K_{uv}, b, K_{ll} and c play important roles in streamflow and stream salt generation. However, parameter C_u only has a pronounced effect on salt balance. Therefore, for single output history matching, i.e. based on streamflow, we consider the first five input parameters to be calibrated. Meanwhile, for history matching of streamflow and salt, we adjust six global input parameters.

Table 3.1: Six sensitive global parameters of LUCICAT

Parameter	Role	In equation	Range
ia	interflow generation	(3.11)	1.8-3.1
K_{uv}	vertical conductivity of top soil	(3.12)	15.3-27.2
b	dry water soil moisture exponent	(3.3), (3.8) and (3.9)	0.156-0.56
K_{ll}	baseflow generation	(3.13)	400-1000
c	wet water soil moisture exponent	(3.8)	0.156-0.56
C_u	salt release from Dry to Wet Store	(3.2.1)	0.0063-0.0263

The Dandalups has numerous nodes. However, only four nodes have historical streamflow data. These nodes are number 14, 25, 257 and 280 as they can be seen in Figure 3.7. There are no historical stream salt provided by these nodes. The length of historical streamflow data for each node is listed in Table 3.2.

Table 3.2: Historial streamflow provided nodes in Dandalup catchment

Node	historictal streamflow	number points
14	01/01/1960-13/11/1992	12006
25	01/01/1984-31/12/2007	8766
257	01/06/1988-23/01/1998	3524
280	25/05/1967-14/05/1992	9122

Since node 14 has the longest historical data compared to the other nodes and node 14 provides the most sensitive history matching results for the region, while the computational time using all nodes will be prohibitive, node 14 is decided to represent The Dandalups catchment.

HM for model parameters of Bates and Lewis is based on historical data provided by one node located in each catchment. In Bates, the node provides the historical streamflow period between 25/06/1988 and 31/12/2010. Meanwhile, in Lewis, the historical streamflow and salt generation is available within period 13/08/1992 and 31/12/2009.

3.5 Concluding Remarks

This chapter presented the overview of three study areas in Western Australia, namely The Dandalups (700 km^2), Bates (2 km^2) and Lewis (2 km^2) catchments. This chapter also described some mathematical equations in the LUCICAT hydrological model to understand the role of the input parameters in the LUCICAT model. Six input parameters, i.e., ia, K_{uv}, b, K_{ll}, c and C_u will be calibrated to match historical streamflow and salt generation and five input parameters, i.e., ia, K_{uv}, b, K_{ll} and c to match observed streamflow. The results of HM will be presented in Chapter 4. The calibration of The Dandalups model parameters will be also performed by coupling HM methods with surrogate models, these results will be presented in Chapter 5.

CHAPTER 4

Application of Direct History Matching Procedures in the Calibration of Hydrological Model Parameters

4.1 Introduction

In this chapter, we present the results of implementing direct history matching methods (HM) for calibrating parameters of hydrological models for The Dandalups (700 km^2), Bates (2 km^2) and Lewis (2 km^2) catchments in Western Australia. The hydrological model used is LUCICAT, described in chapter 3, along with the catchments. We employ four HM methods, described in Chapter 2, namely Ant Colony Optimization based on continuous domain (ACO_R and $DACO_R$), Gauss Levenberg Marquardt (GLM) and Robust Parameter Estimation (ROPE).

The chapter is divided into five major parts. The first part, comprising the first few sections, presents an objective function and algorithm variable settings for the HM methods. An overview of HM methods used for the catchments is provided. Next, results are presented for the Dandalups based on annual peak streamflow, and a comparison of methods and applications. The third part describes performance of HM techniques for The Dandalups catchment based on daily streamflow and a comparison of results. The fourth part presents the performance of HM methods for Bates catchment. Finally, the performance of HM methods for multiple responses is presented; i.e., to match observed streamflow as well as salt generation for Lewis catchment. The chapter ends with some conclusions. The objectives of this chapter are summarised as follows:

(i) To study the sensitivity of tuning variables for HM methods for accuracy and time-efficient application.

(ii) To apply HM methods for the calibration of a hydrological model for each catchment based on available stream flow data.

(iii) To compare the performance of various HM methods in terms of accuracy, consistency, computational time and reduction of uncertainties in initial input parameters.

(iv) To apply HM methods for the calibration of hydrological models for multiple responses using LUCICAT.

4.2 An Objective Function for History Matching

HM methods aim to minimize the difference between the observed and model responses. This difference is quantified by using a root mean square error (RMSE) expressed as follows:

$$\text{RMSE} = \sqrt{q^{-1}\sum_{j=1}^{q}(y^h_{\text{resp}_j} - y_{\text{resp}_j})^2}, \tag{4.1}$$

where $y^h_{\text{resp}_j}$ and y_{resp_j} are the j^{th} observed and modelled response (streamflow in m^3/s or salt generation in tonnes/day) respectively. The modelled response is generated via the LUCICAT model based on a set of input parameters $\mathbf{x}$.

When the two model responses, streamflow and salt generation, are available for history matching we use a weighted objective function (weighted RMSE) expressed as follow:

$$\text{WRMSE} = \omega\sqrt{q^{-1}\sum_{j=1}^{q}\left\{\frac{y^h_{\text{flow}_j} - y_{\text{flow}_j}}{stdev(\text{flow}_o)}\right\}^2} + (1-\omega)\sqrt{q^{-1}\sum_{j=1}^{q}\left\{\frac{y^h_{\text{salt}_j} - y_{ts_j}}{stdev(\text{salt}_o)}\right\}^2} \tag{4.2}$$

where $0 < \omega < 1$, $y^h_{\text{flow}_j}$ and $y^h_{\text{salt}_j}$ are the j^{th} observed streamflow and salt generation respectively; y_{flow_j} and y_{salt_j} are the j^{th} modelled streamflow and salt generation, respectively, generated based using a set of input parameter $\mathbf{x}$. The standard deviations of observed streamflow and salt generation are denoted by $stdev(\text{flow}_o)$ and $stdev(\text{salt}_o)$, respectively.

4.3 Implementation of the Direct HM Method

This section presents details of the settings used to apply the direct HM methodology for calibration of parameters for the catchments. This includes sensitivity analysis of tuning variables used, the generation of initial solutions, the stopping criteria and criteria used for evaluating goodness of fit of the HM methods.

4.3.1 Case Studies and Objective Functions

In this chapter we illustrate four applications of direct HM methodology. Direct HM is aimed to match the historical time series of streamflow for The Dandalups and Bates and both streamflow and salt generation for Lewis. For The Dandalups, firstly the series of annual daily peak streamflow is matched to calibrate the five parameters, $(ia, K_{uv}, b, K_{ll}, c)$ of The Dandalups hydrological model. Next, the same parameters of The Dandalups are calibrated based on daily time series of streamflow. HM methods are then applied for calibration of the input parameters, ia, K_{uv}, b, K_{ll}, and c, of the hydrological model for the Bates catchment. Finally, for the Lewis

catchment, HM is performed to calibrate the six parameters, ia, K_{uv}, b, K_{ll}, c, and C_u, based on both daily steamflow and salt generation, with variable weighting. Further details of historical data duration is presented in Table 4.1.

Table 4.1 presents all scenarios performed in this study.

Table 4.1: Scenarios of HM in This Study

Catchment	Data for History Matching			Number of Points	Validation Period
	Calibration Period	Type	Frequency		
Dandalups	1960-1992	Streamflow	Annual Peak	33	1960-1992
Dandalups	1960-1992	Streamflow	Daily	12,006	1960-1992
Bates	1988-2000	Streamflow	Daily	4,573	2001-2010
Lewis	1992-2004	Streamflow, Salt Generation	Daily	4,524	2005-2009

The objective function used in HM for The Dandalups and Bates catchments are presented in Equation 4.1. For Lewis catchment, HM requires minimising Equation (4.2), where ω values are set to be 0, 0.25, 0.5, 0.75, and 1. All six model parameters (ia, K_{uv}, b, K_{ll}, c, and C_u) for Lewis are calibrated when the salt is present in the objective function, i.e., when $\omega \neq 1$. However, when $\omega = 1$, so that that HM is based solely on streamflow, only the five former input parameters are adjusted.

4.3.2 Initial Solution, Stopping Criterion and Measures of Goodness

The initial solutions for ACO$_R$ and DACO$_R$ HM method are obtained using a Latin Hypercube Design (LHD) of order 50×5 , on $[0, 1]^5$ based on the S-optimality criterion. This criterion ensures uniform coverage of the design space by maximising the mean distance between design points. The generated LHD is then transformed to the scale of the HM parameters before being passed into the LUCICAT for generating the final response. The ROPE method for HM requires an LHD of size 1000×5. In view of the excessive computational time for generating a LHD with the S-optimatliy criterion, here we use a minimax criterion. This criterion ensures uniform coverage of design space by maximising the minimum distance. Meanwhile, the initial solution for GLM is obtained by randomly selecting a row of the S-optimal LHD. LHDs are generated via the lhs.design function in the DoW.wrapper R package [178].

The stopping criterion for ant colony optimisation-based methods is a difference of less than 0.005 between the best RMSE of four consecutive iterations. The termination criterion for ROPE is difficult to determine since the tolerances defined due to the observation errors are arbitrary values [173]. Therefore, for this study, it was decided to use a stopping criterion of the mean

RMSE of two consecutive iterations being less than 0.005 for a maximum of five iterations [173]. Meanwhile, the HM process for GLM stops when the RMSE obtained from four consecutive iterations have improvement of less than 0.005 [160]. For each algorithm the HM process is repeated six times using different initial settings to understand the stability of algorithms in providing results.

The performances of all algorithms are evaluated in terms of accuracy, consistency and computational effort. Accuracy and consistency are quantified by the average and standard deviation of RMSE for six repetitions of the algorithm. RMSE is computed on historical data for The Dandalups catchment and on the validation period (post-historic period) for the Bates and Lewis catchments. Computational effort is measured by computational time and the number of model runs. As the modelling objective of the study is to measure the volume of available water resource, using calibrated parameters of LUCICAT, we also compare methods for their ability to reduce parameter uncertainty, correlation between parameters and non-uniqueness of parameter settings.

4.3.3 Sensitivity Analysis Test for Tuning Variables

4.3.3.1 ACO_R

The four variables that control ACO_R performance are k, m, ξ, and q. As widely recommended in the literature [2, 63, 99, 200], we set the values for k at 50, m at 50 [63, 200]. The values of k and m were considered feasible for a manageable computational time. The tuning variables ξ and q were selected by searching for the best combination of ξ and q providing a low average RMSE and is consistent (measured via standard deviation). Two possible values, 0.5 and 0.8, were considered for ξ and the four values, 0.05, 0.1, 0.2, and 0.5 for q. ACO_R was then executed for all possible combinations of ξ and q to calibrate the five model parameters of The Dandalups (ia, K_{uv}, b, K_{ll} and c), in order to match 33 annual daily peaks. The HM process was repeated three times for each tuning variable combination using different initial settings. The results of each combination in terms of average RMSE values and consistency as indicated by error bars (standard deviation) are presented in Figure 4.1. Results of a two-way ANOVA and Levene's test are also presented in Figure 4.1.

It is observed in Figure 4.1 that the pair of $\xi = 0.8$ and $q = 0.2$ marginally outperforms the other combinations. This pair provides the lowest average RMSE across the selected combinations. However, this combination has high variation in RMSE values, shown by the wide error bars. By comparison, the pairs of $\xi = 0.5$, $q = 0.1$ and $\xi = 0.5$ and $q = 0.2$ show higher consistency but provide poorer average performances (higher average RMSE values) compared to other combinations. Nevertheless, ANOVA results indicate no significant interaction ($p = 0.642$) between ξ and q, even at the 10% level of significance, p-value still indicates there is no significant different. No significant main effects of ξ and q is found. Levene's test also indicates equality of variance ($p = 0.953$) across the combinations of ξ and q. These results indicate that all selected tuning variables have comparable performance in terms of accuracy and consistency. Since the

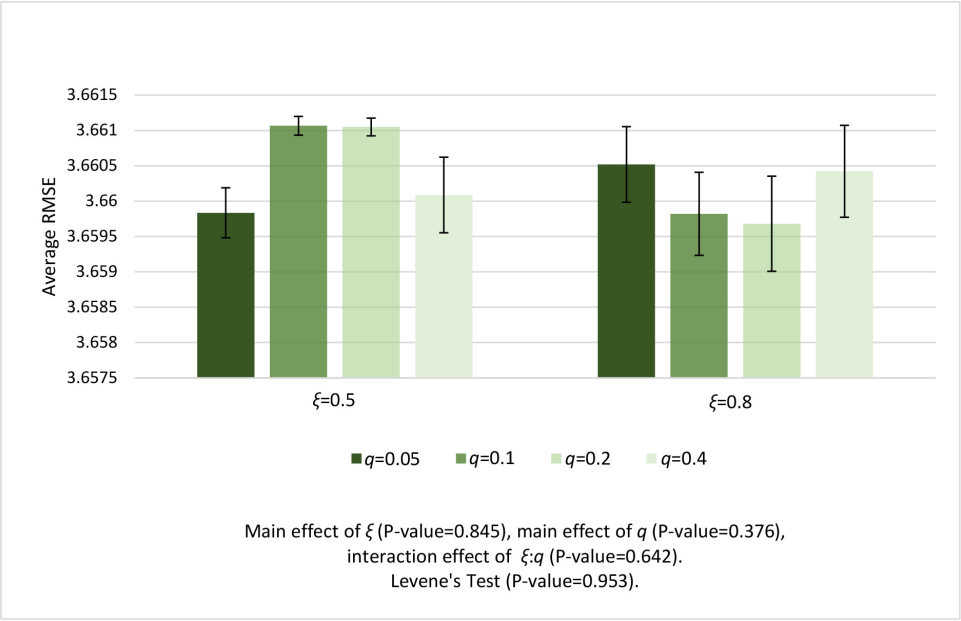

Figure 4.1: Sensitivity Analysis Test for ACO_R Tuning Variables

performance of all tested combinations are similar, we can use any of these combinations as ACO_R tuning variables to calibrate model parameters in different scenarios and catchments.

4.3.3.2 $DACO_R$

Similar to ACO_R, we set the variables of k and m to 50, as in the previous study [63]. The sensitivity test was then focused on finding the best values for the tuning variables ξ and u. Three possible values were considered for ξ and u. The values selected for ξ were 0.6, 0.75, and 0.9; values for u were 0.1, 0.2, and 0.4. $DACO_R$ was run based on all possible combinations of ξ and u to match 33 annual daily peaks of streamflow for The Dandalups. The purpose of HM was to calibrate the five model parameters mentioned earlier for ACO_R. The performance of each combination in terms of mean RMSE and consistency (error bars) are presented in Figure 4.2. Two-way ANOVA and Levene's tests were performed for $DACO_R$ results and the results of these tests are included in Figure 4.2.

Figure 4.2 depicts that for $u = 0.1$, coupled with all ξ values, provides slightly lower RMSE values as well as higher consistency (shorter error bars) compared to other pairs. ANOVA results indicate no significant interaction ($p = 0.725$) between ξ and u, even at the 10% level of significance and no significant main effects (ξ: $p = 0.233$, u:$p = 0.172$). Levene's test also indicates equality of variance ($p = 0.584$) across the combinations of ξ and u. These results demonstrate that all selected pairs of tuning values for $DACO_R$ perform equally, hence any of the tested parameter setting values may be used for HM.

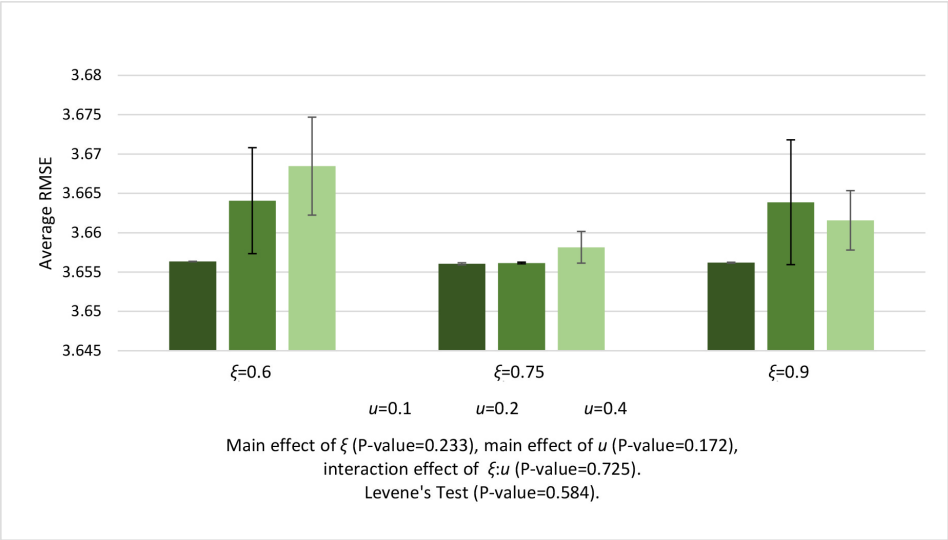

Figure 4.2: Sensitivity Analysis Test for DACO$_R$ Tuning Variables

4.3.3.3 ROPE

The performance of ROPE is controlled by three variables, which are the initial number of solutions (n), the number of new solutions generated (m) and the percentage of good performing parameters (pr). The sensitivity test for ROPE was focused on n only, with pr was set at 10% [4] and m was set to be equal to n. We selected four different values of n, namely 1000, 2000, 2500, and 5000. For each n, ROPE was run to match 33 annual daily peaks of streamflow in order to calibrate the values of five optimal model parameters for The Dandalups. The process was repeated three times for each n using different initial settings. The performance of ROPE was then evaluated via RMSE. The average and standard deviation of RMSE was compared across the values for n. These results, including the results obtained from one-way ANOVA and Levene's tests, are depicted in Figure 4.3.

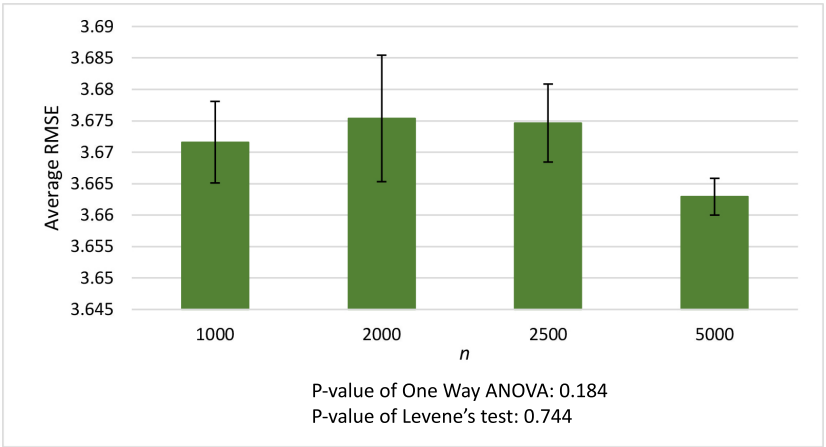

Figure 4.3: Sensitivity Analysis Test for ROPE Tuning Variables

As clearly depicted in Figure 4.3, ROPE with $n = 5000$ shows the best performance, providing the lowest RMSE as well as the lowest variation. However, using a sample size of 5000 requires much longer computational time than using a sample size of 1000. For our current experiment, ROPE for $n = 5000$ completed the HM process in more than two weeks, whereas only a quarter of this time was required for $n = 1000$. There are no significant differences among the performances of different choices of n in terms of mean RMSE (the p-value of one way ANOVA is 0.184) and variation (p-value of Levene's test is 0.744). Therefore, we selected ROPE with $n = 1000$ as the optimal value in this study in view of computational time.

4.3.3.4 GLM

The sensitivity test for GLM was focused on two tuning variables, the gain factor ρ, controlling the update input parameter, and the perturbation factor for Jacobian matrix, dx. The remaining variables, initial lambda (λ_0), expansion (λ_e) and shrinkage (λ_s) factors were chosen from Gavin's study [67], where $\lambda_0 = 1e-2$, $\lambda_e = 11$, and $\lambda_s = 9$. Three values were selected for dx ($1e-2$, $1e-3$, and $1e-4$) and four values were chosen for ρ ($1e-4$, $1e-6$, $1e-8$, and $1e-10$), leading to a total of 12 combinations. GLM was then run to perform HM using 33 annual daily peaks of streamflow in order to calibrate the five model parameters for The Dandalups. This process was repeated three times using different initial solutions for each combination. The results of this test are depicted in Figure 4.4. Two-way ANOVA test and Levene's test results are also incorporated into Figure 4.4.

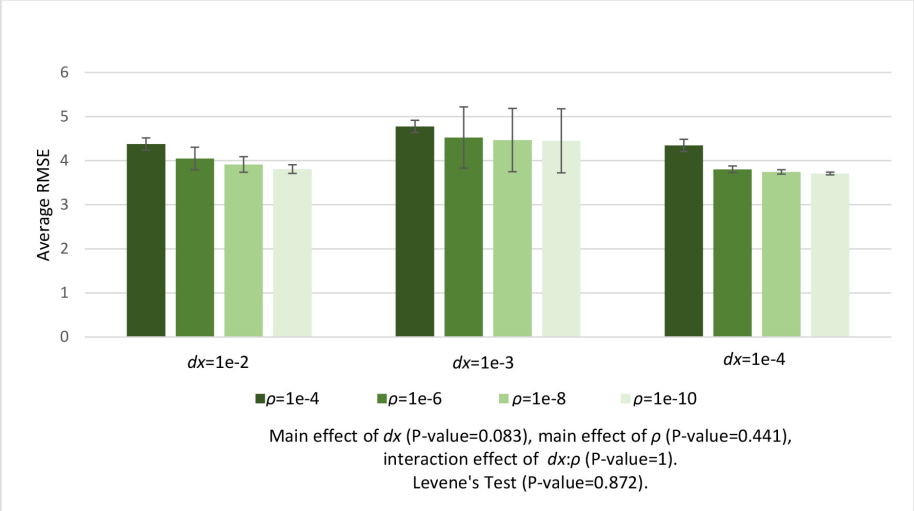

Figure 4.4: Sensitivity Analysis Test for GLM Tuning Variables

Figure 4.4 depicts that GLM provides marginally better performances (average RMSE) among the selected combinations when a tuning variable of $dx = 1e-4$ is used (for different ρ values). GLM with $dx = 1e-4$ also provides better consistency as indicated by shorter error bars compared to those obtained from other combinations. Although GLM performances provided by $dx = 1e-4$ are superior compared to others, all combinations show comparable

performances from a statistical point of view. The p-value for the interaction effect of dx and ρ and each of the main effects is greater than 5%. Based on Levene's test, the variances of the combinations are equal (p-value > 0.05) and show similar consistency. These results imply that all selected tuning variables of GLM have similar performance. Hence any of these can be used as GLM tuning variables to perform HM in calibrating model parameters.

The tuning variables used by each algorithm for HM for each catchment are summarised in Table 4.2 below.

4.4 Results for Direct History Matching

4.4.1 The Dandalups: Annual Daily Streamflow Peaks

Direct HM for The Dandalups was first performed to match 33 annual daily streamflow peaks to calibrate the five model parameters, namely the interflow exponent (ia), vertical conductivity of wet and subsurface store (K_{uv}), the dry water exponent (b), lateral hydraulic conductivity (K_{ll}) and the wet water exponent (c). This approach was based on the consideration that peaks are most influential in the choice of model parameters. The results and discussion are provided in the following sections.

4.4.1.1 The Accuracy and Consistency of HM Methods

Each of the HM methods listed in the previous section was implemented 6 times, and the RMSE for the final solution recorded. Table 4.3 presents summary of performance of HM methods along with results of one-way ANOVA and Levene's test.

As shown in Table 4.3, DACO$_R$ marginally outperforms the other two HM techniques, having the lowest average RMSE. According to the one-way ANOVA test (p-value = 0.026) average RMSE for at least one method is different from the others. According to Tukey's test on the average performance of ACO$_R$, DACO$_R$ and ROPE are comparable, whilst mean RMSE provided by GLM is statistically different from that obtained from DACO$_R$ at the 5% significance level. In spite of this, GLM performance is comparable with the ACO$_R$ and ROPE methods.

In terms of variation within results (Table 4.3), DACO$_R$ is the most consistent method, whilst GLM is the most inconsistent providing nearly a twenty times higher standard deviation to those obtained by ACO$_R$ and ROPE and more than 700 times that obtained from the DACO$_R$ results. The Levene's test results presented in Table 4.3 indicate that at least one pair of algorithms does not have equal variance at a significance level of 0.05. Further exploring for pairwise differences, using Levene's test with a Bonferroni correction, reveals that GLM has unequal variances compared with all multiple solution-based methods at a significance level of 0.008($= 0.005/6$). The performances of ACO$_R$ and ROPE are comparable. In conclusion, the results of GLM in performing HM using annual daily peaks for The Dandalups are not as consistent as those of multiple solution-based methods.

Table 4.2: Parameters of HM Methods for all catchments.

HM of Annual Daily Peaks for The Dandalups					
HM Methods	k	m	ξ	q	
ACO_R	50	50	0.5	0.05	
HM Methods	k	m	ξ	u	
$DACO_R$	50	50	0.9	0.1	
HM Methods	λ_0	λ_e	λ_s	dx	rho
GLM	$1e-2$	11	9	$1e-4$	$1e-10$
HM Methods	n	m	pr		
ROPE	1000	1000	10%		
HM of Daily Time Series for The Dandalups					
HM Methods	k	m	ξ	q	
ACO_R	50	50	0.5	0.05	
HM Methods	k	m	ξ	u	
$DACO_R$	50	50	0.6	0.1	
HM Methods	λ_0	λ_e	λ_s	dx	rho
GLM	$1e-2$	11	9	$1e-3$	$1e-10$
HM Methods	n	m	pr		
ROPE	1000	1000	10%		
HM of Daily Time Series for Bates Catchment					
HM Methods	k	m	ξ	q	
ACO_R	50	50	0.8	0.1	
HM Methods	k	m	ξ	u	
$DACO_R$	50	50	0.6	0.1	
HM Methods	λ_0	λ_e	λ_s	dx	rho
GLM	$1e-2$	11	9	$1e-3$	$1e-10$
HM Methods	n	m	pr		
ROPE	1000	1000	10%		
HM of Daily Time Series for The Lewis Catchment					
HM Methods	k	m	ξ	q	
ACO_R	50	50	0.5	0.05	
HM Methods	k	m	ξ	u	
$DACO_R$	50	50	0.9	0.1	
HM Methods	λ_0	λ_e	λ_s	dx	rho
GLM	$1e-2$	11	9	$1e-3$	$1e-10$
HM Methods	n	m	pr		
ROPE	1000	1000	10%		

Table 4.3: Average RMSE for HM methods- methods with the same subscript represents pairs with equal means at the 5% level of significance based on Tukey's multiple comparison test. Variance is based on Levene's test with Bonferroni correction.

	ACO_R	$DACO_R$	ROPE	GLM
average	3.665_{ab}	3.656_{a}	3.675_{ab}	3.840_{b}
stdev	$1.246e-2_{a}$	$2.983e-4_{a}$	$1.208e-2_{a}$	$2.201e-1$
p-value for one-way ANOVA				0.026
p-value for Levene's test				0.000

4.4.1.2 Computational time and number of models for HM

Table 4.4 presents the comparison of average computational time and number of model runs required by each method. The average computational time across methods is statistically significantly different (p-value is 0 to 3dp). It is obviously from Table 4.4 that GLM is the most efficient technique, with average time required less than one hour with a much smaller number of model runs required for convergence among HM methods applied in this study. Conversely, ROPE needs a calibration time nearly nine times longer than ACO_R. It is observed that $DACO_R$ and ACO_R require comparable numbers of models to complete the HM process.

Table 4.4: Computational Time and Number of Model Runs of All HM Methods.

	ACO_R	$DACO_R$	ROPE	GLM
Computational time (hour)	9.167	6.406	81.965	0.872
Number of model runs	550	517	6000	53
p-value of one-way ANOVA test				0.000

4.4.1.3 Reduction of Input Parameter and Streamflow Uncertainties

The performance of HM methods to reduce the initial ranges of parameter uncertainties are depicted in Figure 4.5. The optimal ranges of GLM solutions were obtained from six final input parameters of six experiments. Meanwhile, the ranges of input parameters for multiple solution-based methods were provided from six combinations of an input parameter set that corresponds to the best RMSE obtained from each experiment. Figure 4.5 shows that ACO_R, $DACO_R$ and ROPE are capable of identifying a narrow optimal region in parameter space for most input parameters, providing a reduction of more than 75% compared to initial uncertainties of parameters *ia*, K_{uv}, *b* and *c*. ACO_R and ROPE, however, can only narrow down the uncertainty of initial K_{ll} ranges to 36.8 % and 46.2 % respectively, while $DACO_R$ and GLM can reduce to

less than 2% for the same input parameter. On the other hand, GLM can only provide reduction of the initial parameter ranges of ia, K_{uv} and c by at the most 2/3. High variation in GLM results for ia, K_{uv} and c may result from a tendency of GLM to become locally trapped at points near the optimum, or proximity to the initial starting points as is commonly found for gradient-based methods [151].

Unlike the other input parameters, all methods show agreement for parameter b, which converges to the upper boundary (0.56). Despite variation in the uncertainty ranges obtained from HM results, all HM methods are able to reduce the initial ranges of input parameters, identifying similar final parameter regions in parameter space. With smaller uncertainty of input parameter ranges, the hydrological model leads to less uncertain future streamflow predictions.

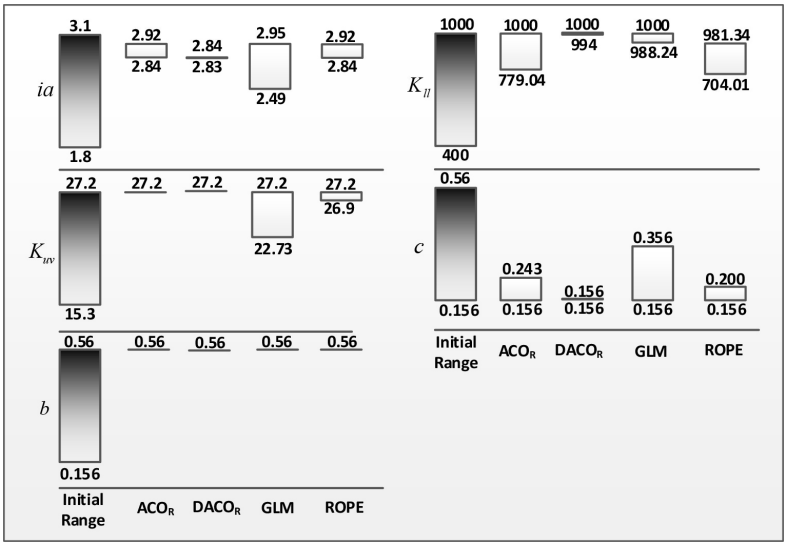

Figure 4.5: Comparison of parameter uncertainties of initial ranges of input parameters and HM results

How much does the final range of input parameters affect uncertainty reduction of streamflow? In order to analyse it, we selected six sets of input parameter results from ACO$_R$. Each set was obtained from a set of input parameters that provided the best RMSE in each run. The LUCICAT model was executed based on these six sets of input parameters to provide six streamflow outputs. Each streamflow output was then averaged to obtain average streamflow per year (m^3/year). To compare these results, we tried all possible combinations of the initial upper and lower boundaries of each of the input parameters. Since the calibration was focused on five input parameters, and each input parameter contained initial upper and lower values, there would be $2^5 = 32$ combinations, i.e., the first set was 1.8,15.3,0.156,400,0.156; the second set was 1.8, 15.3, 0.156, 400, 0.56, and so on. These 32 combinations of input parameters became inputs for the LUCICAT model to generate 32 sets of streamflow. Average streamflow from each set was then computed. The collection of average streamflow of ACO$_R$ results and upper and lower boundaries were compared to provide streamflow uncertainty of streamflow

through boxplots presented in Figure 4.6.

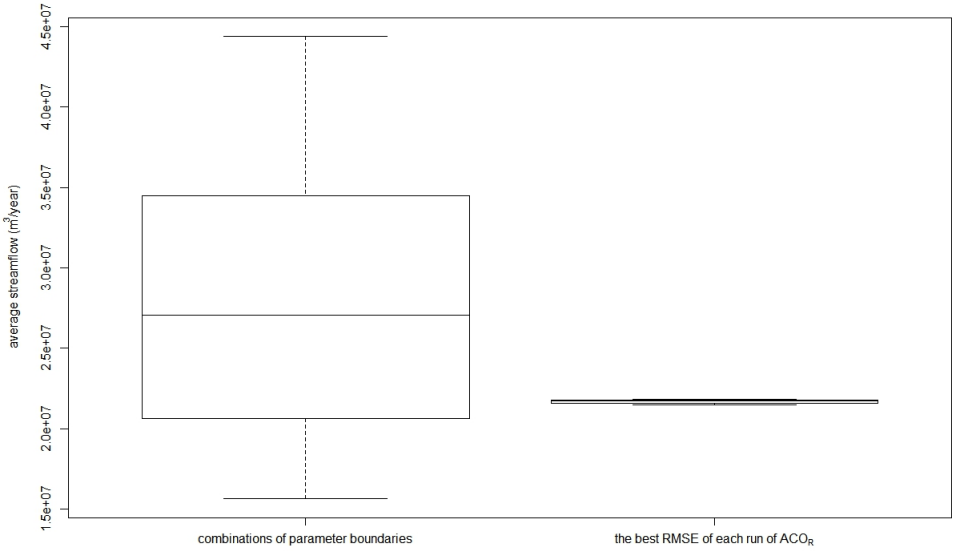

Figure 4.6: Comparison of parameter uncertainties of initial ranges of input parameters and HM results

As depicted in Figure 4.6, the average streamflow obtained from solutions provided by ACO_r are concentrated in the small region around 22 million m^3/day. This region lies between the 30^{th} and 40^{th} percentiles of the range of average streamflow provided by the initial parameter boundaries. This result indicates that ACO_R is able to reduce the uncertainties of average streamflow by 99% of the ranges obtained from combinations of initial boundaries of input parameters.

4.4.1.4 Non-Uniqueness in HM Results

HM commonly provides non-unique solutions [200], which means that many combinations of input parameters can provide similar HM results. In order to present the non-uniqueness in our results, we extracted the results obtained from ACO_R. There were 300 combinations of input parameters provided by executing ACO_R on six different initial sets of input parameters (since ACO_R provided 50 combinations for each set). Out of these combinations, we first selected the input parameter set of the best solution (solution with the lowest RMSE). We then set a tolerance of 0.0001 to find other sets of input parameter sets that provided similar performances (RMSE). Based on this tolerance, another 20 sets of input parameter sets were selected. Cluster analysis was next performed to see which input parameter sets are similar. For this purpose, we used the expectation maximization (EM) algorithm in the Mclust R package [29]. The results are shown in Figure 4.7, which separate input parameter sets into two clusters (indicated by circles and triangles) .

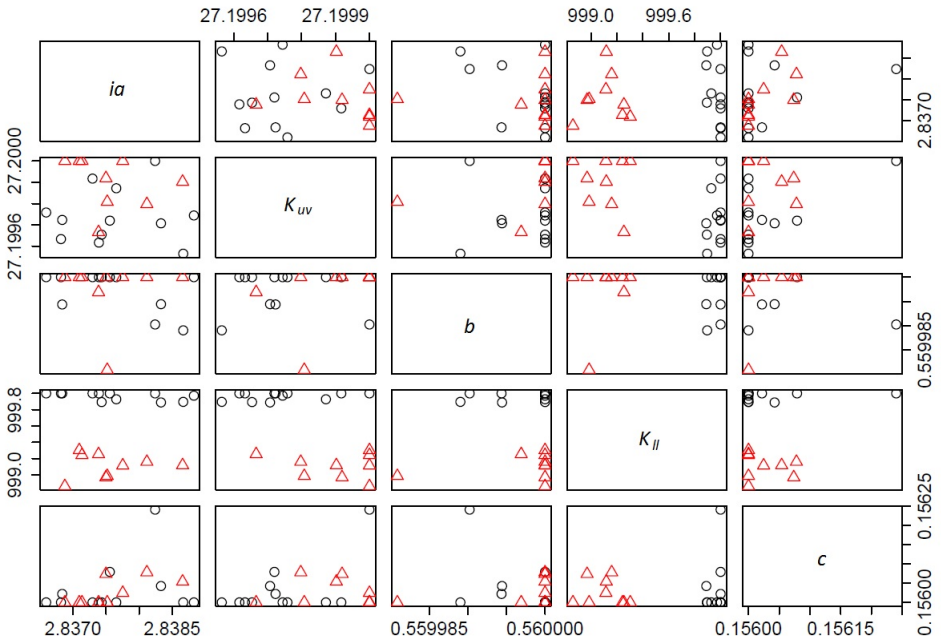

Figure 4.7: Two clusters formed of twenty one combinations of ACO_R

Figure 4.7 clearly depicts that the two groups of ACO_R results are mainly separated by input parameter K_{ll}. This indicates that similar RMSE values can be represented by two marginally different groups due to K_{ll} values. How much does this effect streamflow generation? To answer this, the LUCICAT model was run on two sets of input parameters obtained from the best performing input parameter in each cluster. This resulted in a streamflow difference of only 264 m^3/year.

4.4.1.5 Correlation between Input Parameters

In order to obtain correlations between input parameters, Pearson correlation coefficients were computed for the 21 solutions provided by ACO_R. The results (correlation values/p-values of a pair of input parameters) are presented in Table 4.5.

Table 4.5: Correlation of input parameters of 21 solutions of ACO_R

	Correlation/p-value				
	ia	K_{uv}	b	K_{ll}	c
ia	1	-0.095/0.681	-0.232/0.311	0.034/0.884	0.350/0.119
K_{uv}		1	0.160/0.488	-0.532/0.013	0.300/0.186
b			1	0.039/0.866	-0.197/0.392
K_{ll}				1	0.061/0.793
c					1

Table 4.5 shows that most of the input parameter pairs do not show significant correlation, except for input parameter K_{uv} and K_{ll} (p-value = 0.013). However, since the correlation value is around 0.5, it can be considered that K_{uv} and K_{ll} have no strong correlation.

4.4.1.6 The Graph of the Best Model Parameters Obtained HM Methods

The graphs of streamflow of the best input parameters provided by ACO_R, GLM, ROPE and observed data are depicted in Figure 4.8. Observing Figure 4.8, the hydrographs provided by the HM methods cannot capture the extreme points (due to wet and dry years) of observed streamflow well, which occurred in the first half of study period. However, after 1980 all the observed annual daily peak points are captured quite well. Figure 4.8 shows depicts that all results show agreement in HM results.

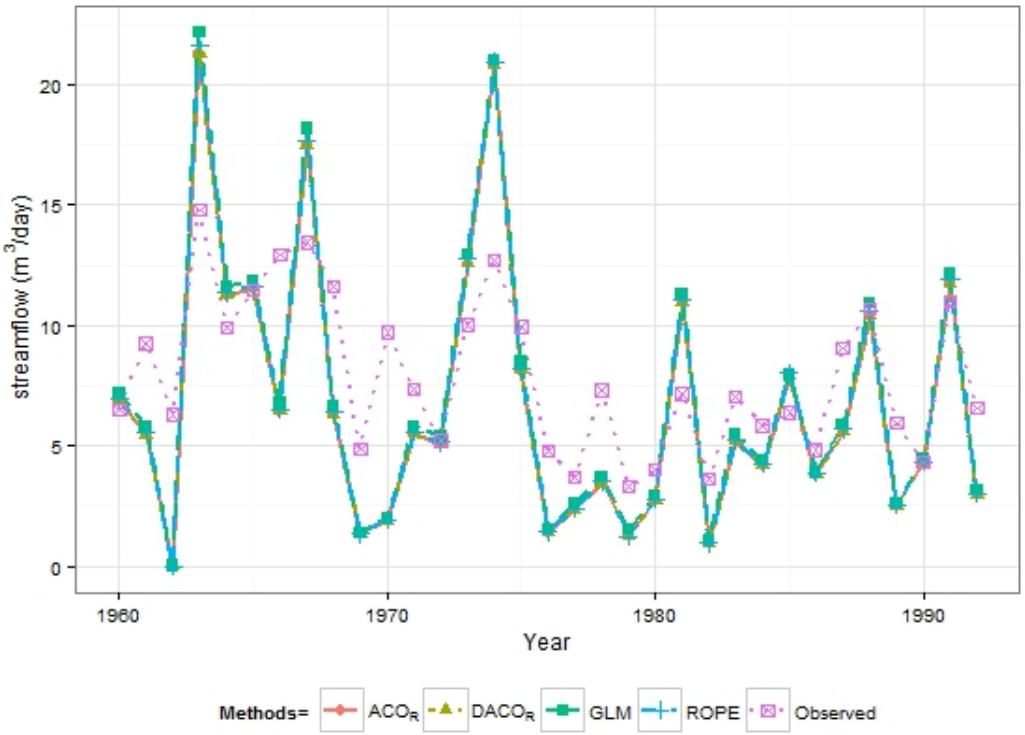

Figure 4.8: Hydrographs of annual daily peak of the best input parameter obtained from ACO_R, GLM adn ROPE for The Dandalups periods between 1960 and 1992

In order to understand the effect of HM points on HM method performance we present the HM based on daily streamflow for The Dandalups in the following section.

4.4.2 The Dandalups: Daily Time Series of Streamflow

The second scenario of direct HM is to calibrate the five model parameters of The Dandalups (ia, K_{uv}, b, K_{ll} and c) using daily streamflow in the period 1960 to 1992. The results of each HM method are discussed in the following sections.

4.4.2.1 The Accuracy and Consistency of HM Methods

Table 4.6 presents the comparison between average RMSE and consistency (standard deviation) of ant colony optimization methods, ROPE, and GLM, along with the best so far model performance (RMSE) provided by industry.

Table 4.6: Performances of ACO_R, $DACO_R$, ROPE and GLM.

	ACO_R	$DACO_R$	ROPE	GLM	estimated by industry [42]
average RMSE	1.372	1.372	1.376	1.410	1.616
stdev of RMSE	$7.93e-4$	$4.28e-4$	$2.67e-3$	$5.20e-2$	
p-value for one-way ANOVA			0.055		
p-value for Levene's test			0.3822		

It is observed in Table 4.6 that the performances of ACO_R and $DACO_R$ are comparable and slightly outperform the local deterministic-based approach (GLM) and ROPE. However, based on the p-value of a one-way ANOVA test, all HM methods have similar average performances at a significance level of 0.05 (p-value = 0.055). In terms of consistency, GLM shows the highest variation within its results among HM methods. Conversely, $DACO_R$ is the most consistent in providing results. Nevertheless, the p-value of 0.3822 for Levene's test indicates that the algorithms have equal variances at a significance level of 0.05. Furthermore, compared with the best estimate reported by industry, all of the methods show an improvement of at least 12% on RMSE.

4.4.2.2 Computational Time and Number of Models for Convergence

The average computation time and number of models of six experiments for each algorithm are listed in Table 4.7. At least one pair of algorithms show significant difference in mean computational time (p-value = 0.000). Among the algorithms implemented in this study, GLM is a very time-efficient method providing results on average in one and half hours, with the least number of LUCICAT models run. According to Tukey's comparison test, both ant colony based-methods require comparable computational time for convergence. The ROPE method is computationally expensive, requiring more than ten times longer in stopping time compared to ant colony-based approaches.

Table 4.7: Computational time and number of model runs of all HM Methods. Methods within the same subscript represents pairs with equal means at the 5% level of significance based on Tukey's multiple comparison test.

	ACO_R	$DACO_R$	ROPE	GLM
Computational time (hour)	6.972_a	7.645_a	82.718	1.51
Number of model runs	383	417	6000	93
p-value of one-way ANOVA test				0.000

4.4.2.3 Reduction in Model Parameter Uncertainties

Reduction in initial uncertainty for the five input parameters calibrated by ACO_R, $DACO_R$, ROPE and GLM are depicted in Figure 4.9. The uncertainty ranges were generated from six values of the best RMSE obtained from each run for multiple solution-based methods, and six RMSE values collected from all runs for GLM. Figure 4.9 illustrates that all algorithms show a consistent direction in locating optimal regions for most input parameters. For parameter *ia*, GLM is observed to concentrate in different regions from those obtained from ant colony-based methods and ROPE. This may be result from GLM's restriction to locally search the optimal regions near the initial points [124], which is common in most gradient-based methods [160]. If the number of initial input parameters are increased, GLM may have a chance to provide better solutions [124].

Both ant-based optimization methods and ROPE are able to reduce the initial input parameter ranges of *ia* to less than 5%, while GLM reduces to approximately 31% of the same initial uncertainties. Moreover, for parameter *b*, it is observed that GLM demonstrates the highest variation in its results compared to the other HM methods, reducing the initial input parameters to about 20%. ROPE is able to reduce to less than 5% of initial ranges of *b*.

4.4.2.4 Model Parameter Correlation

Table 4.8 presents Pearson correlation coefficients between calibrated input parameter pairs and their corresponding p-values. The correlation coefficients were computed based on 250 sets of calibrated input parameters, whose corresponding RMSE is within a 0.001 tolerance of the best-performing solution set, selected from 300 available solutions provided by ACO_R (since we run six runs for ACO_R and each run provides 50 solutions).

Table 4.8 shows that the correlation of all pairs of parameters are weak, although some pairs demonstrate significant relationships, such as the pairs *ia* and *b*, *ia* and *c*, and *b* and *c*. These results show agreement with those obtained from HM on annual daily peak flow in the first scenario of this study, where model parameters of The Dandalups are not strongly correlated.

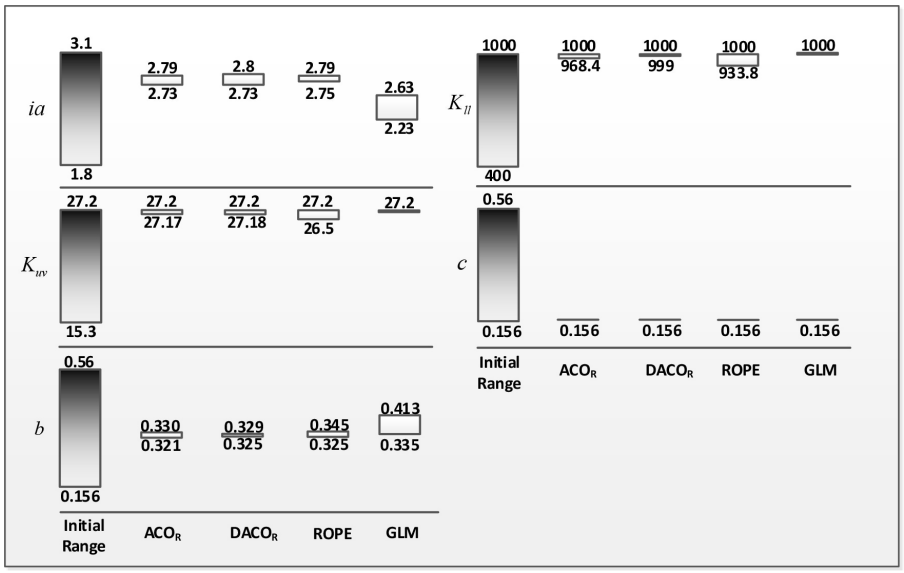

Figure 4.9: Comparison of reduction initial model parameter uncertainties performed by ACO$_R$, DACO$_R$, ROPE, and GLM.

Table 4.8: Correlation of input parameter of 250 solutions of ACO$_R$.

	Correlation Coefficient/p-value				
ACO$_R$					
	ia	K_{uv}	b	K_{ll}	c
ia	1	-0.086/0.173	-0.432/0	0.050/0.434	0.333/0
K_{uv}		1	0.110/0.082	-0.106/0.096	-0.050/0.428
b			1	-0.098/0.121	-0.255/0
K_{ll}				1	0.092/0.147
c					1

4.4.3 Comparison of HM Methods on Annual Daily Peaks and Daily Time Series of Streamflow

In order to understand the sensitivity of the selection of the number of points for HM on the performance of these methods, we compared the results provided by ACO$_R$, DACO$_R$, ROPE and GLM from Section 4.4.1 HM based on annual daily peaks, and the previous section on daily streamflow. Since we used different numbers of points to calculate RMSE for annual daily peaks and daily streamflow, we rerun the LUCICAT model based on the model parameters obtained from HM of annual daily peaks to predict daily streamflow, and then compute the corresponding RMSE. In this setup the performance of model parameters for both scenarios (annual daily peaks and daily time series) can be compared. The results are listed in Table 4.9. A t-test analysis was performed for comparing mean RMSE and mean computational time, while F-test analysis was

conducted for equality of two variances. The results of these analyses are presented by p-values listed in Table 4.9.

Table 4.9: Average RMSE, standard deviation and computational time of HM methods based on HM annual daily peak and daily streamflow.

	ACO_R			$DACO_R$		
	peak	daily	p-value	peak	daily	p-value
mean RMSE	1.439	1.372	0.007	1.419	1.372	0.000
stdev	0.0364	0.0008	0.000	0.0001	0.0004	0.005
computational time (hours)	9.167	6.972	0.000	6.406	7.645	0.110
	ROPE			GLM		
	peak	daily	p-value	peak	daily	p-value
mean RMSE	1.453	1.375	0.000	1.528	1.410	0.053
stdev	0.021	0.003	0.000	0.1135	0.0520	0.112
computational time (hours)	81.965	82.718	0.011	0.872	1.51	0.005

It is observed in Table 4.9 that the average RMSE values for HM on daily streamflow for all multiple solution-based methods are significantly different compared to the those provided by HM on peak points at significance level 0.05. By contrast, average RMSE for GLM based on HM of daily time series or annual daily peaks (p-value = 0.053) are comparable at the 5% significance level. All algorithms show better performance when the number of HM points is increased. These results may reveal that the selection of 33 annual daily peaks may not be sufficient to search for "a good representation of input parameters of a model". However, a large number of data points for calibration may make the processes computationally prohibitive.

All multiple solution-based methods which use HM on daily streamflow provide significantly smaller variation within their results, compared to those provided by annual daily peak model parameters. This indicates that stability of algorithms in searching for the optimal region within the input parameter space increases with the number of HM points. This could also be an artifact of computation of RMSE. Moreover, from the t-test results, it is observed there are significantly different average computational times required for all algorithms. HM time required by $DACO_R$, ROPE, and GLM methods is higher when more calibration points are included in the HM process.

Figure 4.10 compares the optimal region provided by ACO_R, $DACO_R$ and GLM for HM based on annual daily peaks (indicated by -p) and daily time series (indicated by -d). Observing the results depicted in Figure 4.10, all HM methods lead to solutions encompassing similar model parameter regions for K_{uv}, K_{ll} , and c for both scenarios. Conversely, the optimal regions of parameter ia obtained from matching annual daily peaks do not overlap the regions obtained

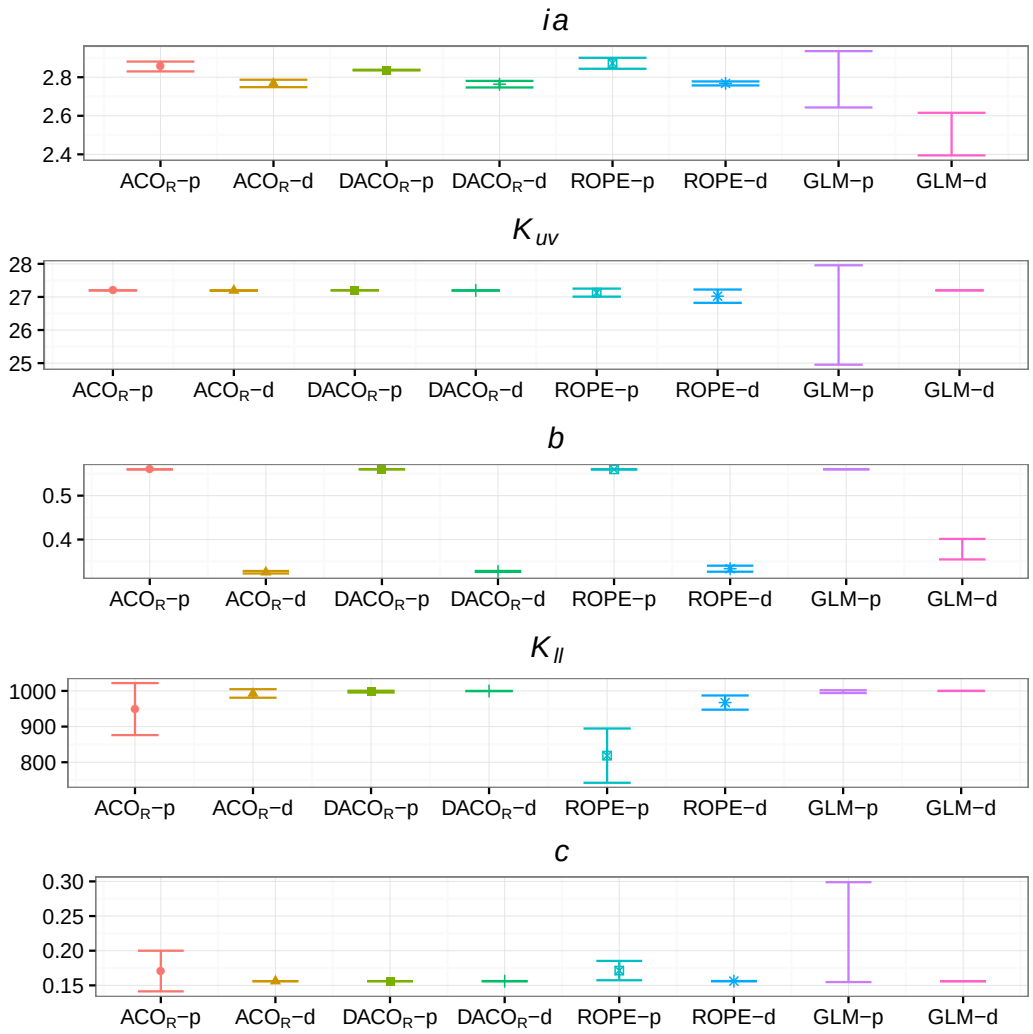

Figure 4.10: Comparison of optimal model parameter region performed by all methods based on HM of annual daily peaks and daily streamflow.

by matching daily streamflow. Even the model parameter b estimates are located in different optimal regions (parameters from HM of annual daily peaks are located in the upper boundary and from daily time series lie around the lower boundary of initial input parameter range). This means that the choice of the number of HM points greatly influences the obtained model parameters.

4.4.4 Bates Catchment: Daily Time Series of Streamflow

Next we apply HM methods to calibrate parameters (ia, K_{uv}, b, K_{ll} and c) for the much smaller Bates catchment. Calibration is performed by matching the observed daily streamflow from 1988 to 2000 via methods ACO$_R$, DACO$_R$, ROPE and GLM. For each method, the resulting set of calibrated input parameters are also used to forecast streamflow for the years 2001 to 2010. The quality of the forecast is accessed by comparing observed and forecast series through

RMSE. This process is repeated 6 times.

4.4.4.1 The Accuracy and Consistency of HM Methods

The accuracy and consistency of HM results are presented in Table 4.10 along with the best estimate obtained by industry until now. The table also presents the results of ANOVA and Levene's test.

Table 4.10: Average RMSE for history matching methods
Methods within the same subscript represents pairs with equal mean RMSE at the 5% level of significance based on Tukey's multiple comparison. Variance is based on an F-test for pairwise comparison with Bonferroni correction.

	Calibration Period			
	ACO_R	$DACO_R$	ROPE	GLM
average RMSE	$1.158e-2_a$	$1.158e-2_a$	$1.158e-2_a$	$1.181e-2$
stdev RMSE	$2.63e-5$	$4.28e-5$	$2.27e-5$	$2.20e-5$
p-value for one way ANOVA	0.000	(for four methods)		
p-value for Levene's test	0.6143			
industry estimates	$2.022e-2$			
	Validation Period			
	ACO_R	$DACO_R$	ROPE	GLM
average RMSE	$4.986e-3_a$	$5.022e-3_a$	$4.906e-3_a$	$5.393e-3$
stdev of RMSE	$3.36e-4_b$	$2.43e-4_b$	$7.65e-5_a$	$1.47e-4_{ab}$
p-value for one way ANOVA	0.006			
p-value for Levene's test	0.018			
industry estimates [42]	$1.362e-2$			

During the calibration period, ANOVA results in Table 4.10 show average RMSE values for all algorithms are not the same (p-value = 0 to 3dp) even at the 1% level of significance. This is due to the average RMSE of GLM being nearly 20% higher than the ant optimization based methods and ROPE. In contrast to the previous findings for The Dandalups, $DACO_R$ shows the least consistency in its results compared to the other algorithm in the calibration period. However, according to Levene's test (p-value = 0.6143), all algorithms have equal variances in RMSE.

During the validation period, ANOVA results in (Table 4.10) show that average RMSE for all algorithms are not the same (p-value = 0.006), even at the 1% level of significance. Average RMSE for GLM is higher than the other methods. In regards to the variance in RMSE, homogeneity of variance cannot be established (p-value = 0.018). The ROPE and GLM methods have comparable and smaller variance in RMSE compared to the other methods. In spite of

the varied performance among HM methods, all methods are able to improve the accuracy of industry estimates by more than 41% in the calibration period and by more than 60% in the validation period.

The RMSE values obtained from calibration period in (Table 4.10) for all HM methods are greater than those obtained from the validation period. This is largely due to the fact that the calibration period had more wet days, with higher intensity of rain, compared to the validation period. This may have led to the model output based on the obtained model parameters overestimating some peaks in the calibration period, resulting in a higher deviation between the model output and observed streamflow than in validation period.

4.4.4.2 Computational Time and Number of Models for HM

Table 4.11 presents the computational time and number of LUCICAT models required to achieve convergence. Computational time for converge (p-value = 0.000) for all algorithms is not the same. GLM is very time-efficient, requiring less than an hour and about 186 LUCICAT models to complete HM. On the other hand, ROPE requires the most computational time (nearly a day) due to the requirement for a large population size. Meanwhile, ACO_R and $DACO_R$ have comparable computational time for convergence.

Table 4.11: Computational time and number of model runs of ACO_R, $DACO_R$, ROPE and GLM,

	ACO_R	$DACO_R$	ROPE	GLM
Computational time (seconds)	6,647	6,347	78,913	2,972
Number of model runs	400	400	6000	186
p-value		0.000		

4.4.4.3 Reduction in Model Parameter Uncertainties

The reduction of initial input parameter ranges for Bates is presented in Figure 4.11. The optimal ranges were collected from the six best input parameter settings obtained from each run of ACO_R, $DACO_R$, ROPE and GLM. Figure 4.11 shows that all algorithms have a consistent direction in locating optimal region in parameter space. The input parameter regions of ia and K_{uv} obtained from GLM, however, slightly deviate from those obtained from ant colony-based optimization and ROPE techniques. Meanwhile, all HM methods converge to the same regions for the remaining model parameter. ROPE and GLM are able to narrow down uncertainty in initial input parameters by at least 85%, whereas ACO_R and $DACO_R$ are able to reduce uncertainity by at least 65%.

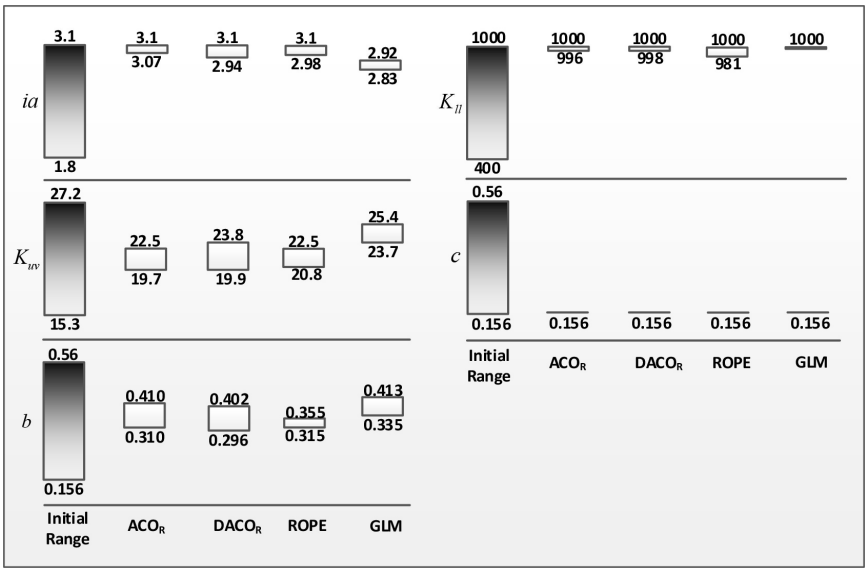

Figure 4.11: Comparison of the reduction initial model parameter uncertainties performed by ACO$_R$, DACO$_R$and GLM.

4.4.4.4 Non-Uniqueness in HM Results

We selected a set of input parameters of six experiments performed by ACO$_R$. These input parameters provide the RMSE values within a tolerance of 0.0001 with respect to the best RMSE within the ACO$_R$ results. Out of 300 combinations of input parameters, there were 211 combinations within this tolerance. We then performed cluster analysis. Five clusters were created by using the Mclust R package [29]. Each input parameter in each cluster was averaged and normalized with respect to the corresponding initial input parameter range. We plot the cluster analysis results in Figure 4.12. Each line represents each cluster.

Observing Figure 4.12, these clusters are mainly different for parameters K_{uv} and b (marked by two ellipses). In order to understand the impact of these clusters on streamflow generation during the calibration period, we selected the best combinations of input parameters, corresponding to the lowest RMSE, from each cluster. For these parameters, the LUCICAT model was executed to obtain streamflow. The results are presented in Table 4.12. This shows the differences of streamflow between the best input parameters of clusters ranging between 25,000 and 42,000 $m^3/year$. According to the Water Corporation [186], the average water consumption per person is 127 $m^3/year$ (in 2015-2016). With these differences, the water can be supplied to between 197 and 331 people or equivalent to 35% and 60% population of North Dandalup respectively based on census 2011 [143]. These results indicate that similar performances (within a tolerance of $1e-4$) of different input parameter values can provide different streamflow generation. This shows non-uniqueness of HM results.

Table 4.12 shows the differences of streamflow between the best input parameters of clusters ranging between 25,000 and 42,000 $m^3/year$. These results indicate that similar performances

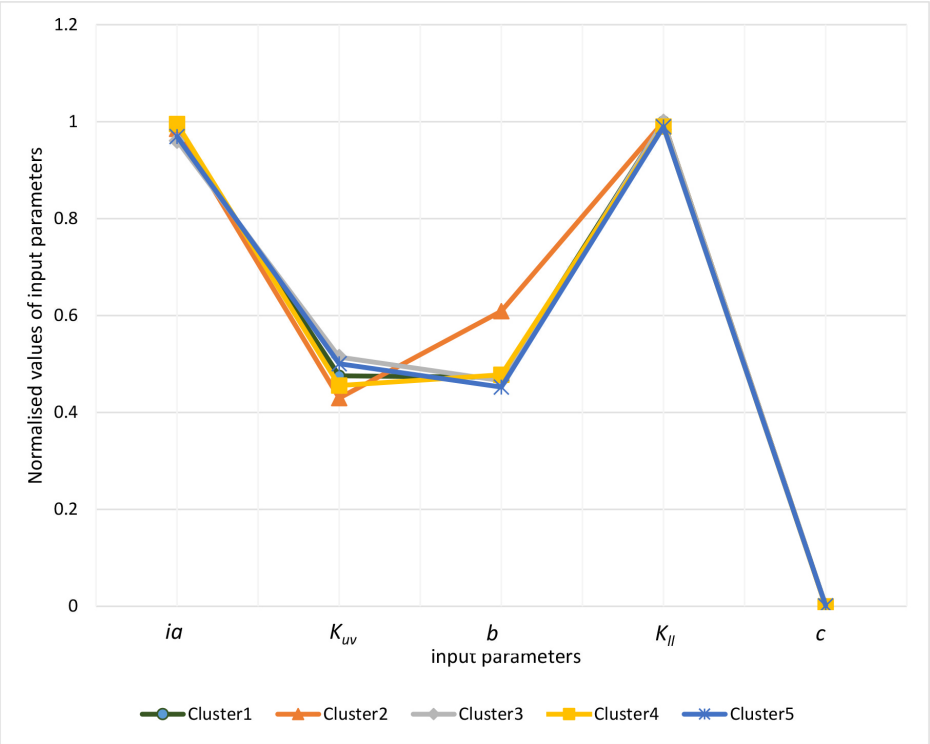

Figure 4.12: ACO_R clusters formed from 211 input parameter settings.

(within tolerance of $1e-4$) of different input parameters values can provide different streamflow generation. This shows non-uniqueness of HM results.

4.4.4.5 Correlation in Bates Model Parameters

The correlation of input parameters was computed using the previous 211 combinations of input parameters, obtained from ACO_R having RMSE within the tolerance $1e-4$ from the best RMSE obtained in the ACO_R results. The Pearson correlation coefficients of pairs of input parameters and their corresponding p-values are listed in Table 4.13.

Table 4.13 shows ia and K_{uv}, and K_{uv} and b have moderate correlation (correlation coefficient values less than -0.5 and p-value < 0.01). Meanwhile, the remaining pairs are observed to have weak relationships. The presence of parameter correlation may cause poor identification of model parameters which may lead to unsuccessful parameter calibration [160]. Pragmatically, this means that the HM process may result in parameter settings that may not be feasible, for instance a low value of both ia and K_{uv}. If correlation between the parameters is expected and known, than it can be incorporated in designing initial settings. In some cases it may be possible to collate correlated parameters for HM into one parameter. However for adopting any such approach, a prerequisite is the quantification of parameter correlation, which may be difficult to achieve.

Table 4.12: The effect on the best input parameter of each cluster on streamflow generation

Cluster	ia	K_{uv}	b	K_{ll}	c	RMSE ($x1e-2$)	streamflow ($x1000m^3/year$)
1	3.068	20.830	0.352	1000	0.156	1.156	397
2	3.100	19.873	0.441	999.829	0.156	1.163	422
3	3.009	22.167	0.348	999.623	0.156	1.159	394
4	3.100	21.122	0.317	999.187	0.156	1.158	380
5	3.100	20.802	0.349	995.751	0.156	1.158	391

Table 4.13: Model parameter correlations for ACO_R.

	Correlation Coefficient/p-value				
ACO_R					
	ia	K_{uv}	b	K_{ll}	c
ia	1	-0.737/0	0.275/0	-0.125/0.069	0.063/0.363
K_{uv}		1	-0.611/0	0.133/0.054	-0.142/0.040
b			1	-0.0182/0.7931	0.117/0.089
K_{ll}				1	0.076/0.2693
c					1

4.4.4.6 The Hydrographs of the Best Model Parameters

Figure 4.13 and Figure 4.14 depict streamflow for the model parameters obtained from the best HM match case. The figures also present observed streamflow in the corresponding period. The top left pictures in Figure 4.13 and Figure 4.14 describe the streamflow for the calibration and validation periods. The remaining pictures demonstrate the streamflow behaviours for particular years (calibration years: 1990, 1998 and 2000 and validation years: 2002, 2006 and 2009).

Figure 4.13 illustrates that the hydrographs of streamflow provided by the model parameters via HM methods show a good fit with the observed streamflow curve. Only a few peaks were not captured well by the model parameters. The model parameter hydrographs also indicate a good match in the validation period (Figure 4.14).

4.4.5 Lewis catchment: Daily Time Series of Streamflow and Salt Generation

In this section HM is performed to match the observed series of streamflow and salt generation for Lewis catchment. The catchment has historical data available from 1992 to 2009. We consider data from the years 1992 to 2004 as the calibration period, used to fit all HM models and

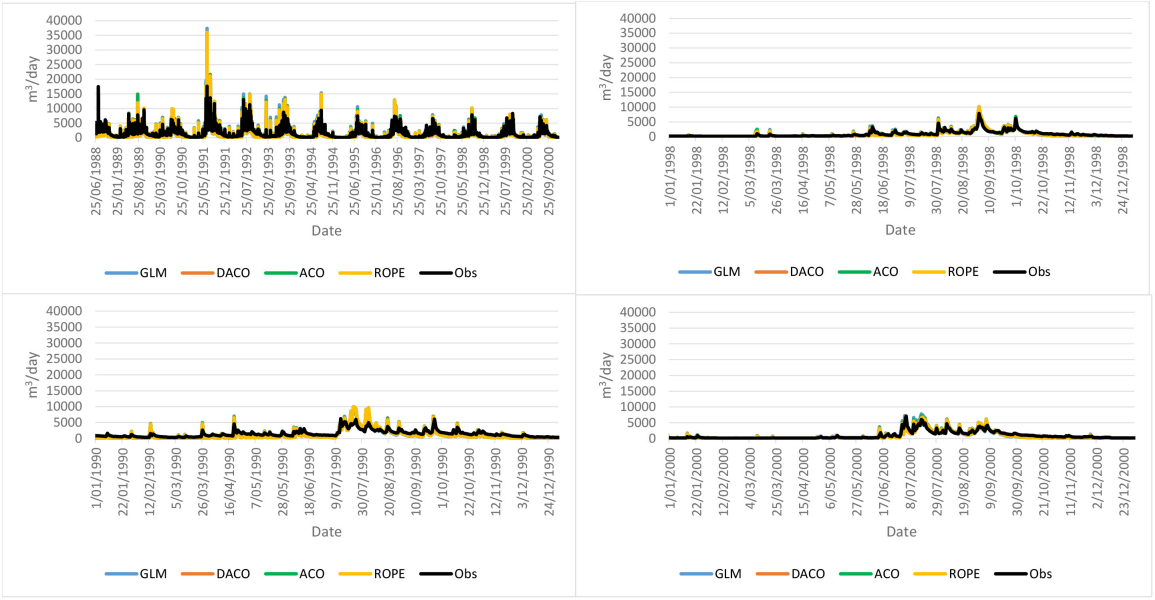

Figure 4.13: Hydrographs obtained from HM methods in the calibration period (1988-2000)

generate optimal parameter settings. These settings are then used for prediction for years 2005 to 2009, regarded as the validation period, to assess goodness of history matching for forward prediction.

Once again ACO_R, $DACO_R$, ROPE and GLM are implemented to search for six optimal performing model parameters, namely ia, K_{uv}, b, K_{ll}, c, and C_u when the salt component is present in the objective function (Equation 4.2); that is, when $\omega = 0, 0.25, 0.5,$ and 0.75. The first five parameters play important roles in both streamflow and salt generation, while C_u is an important parameter for salt generation only (Equation 3.5 in Chapter 3). Hence, when HM is only based on streamflow ($\omega = 1$), we calibrate only five input parameters (ia, K_{uv}, b, K_{ll}, c). The results are discussed in the following sections. The process of HM is repeated six times, with random initial seeds, and RMSE is recorded for each case.

4.4.5.1 Comparison of HM Method Performances on Weighted RMSE

Performance of HM methods for average RMSE, for different weightings of salt and streamflow, is presented in Table 4.14. The standard deviation of RMSE along with Levene's test is presented in Table 4.15.

The results indicate that ACO_R marginally outperforms the other methods, providing the smallest weighted RMSE values and the highest consistency within weighted RMSE for all values of ω. Nevertheless, the p-value for the ANOVA, for all weightings, is much less than 1%, indicating that average RMSE across all methods, for each setting of the weight factor, is not the same. Tukey's multiple comparisons (see 4.14) further reveals that both ant colony-based

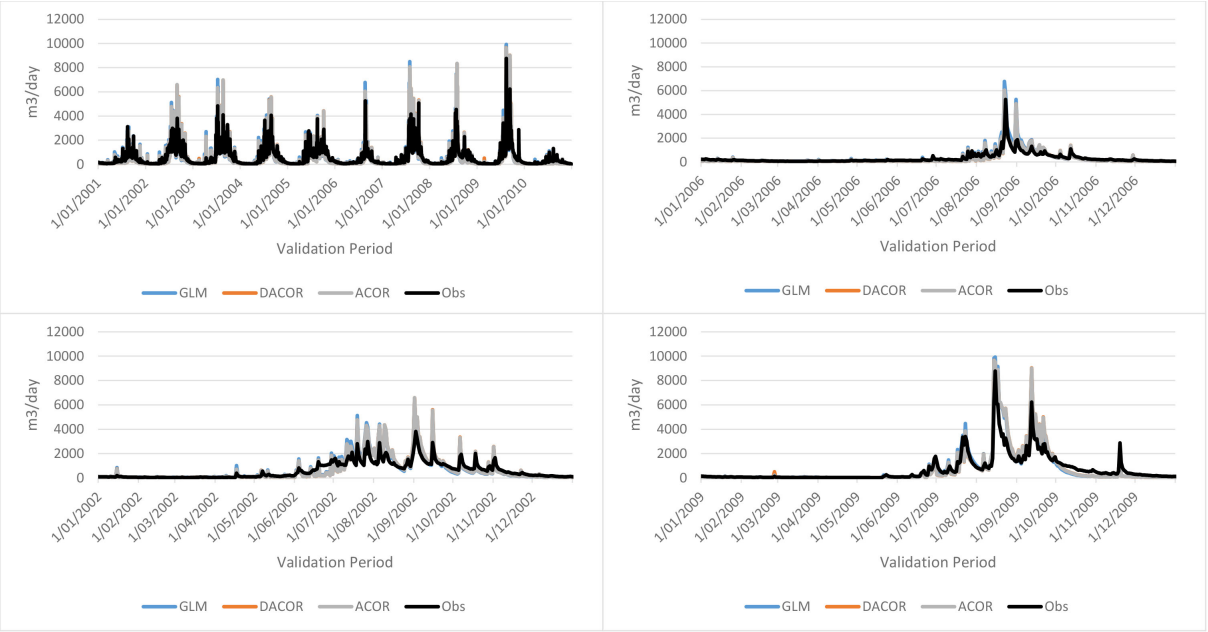

Figure 4.14: Hydrographs obtained from HM results for the validation period (2001-2010)

methods have comparable average performance across all ω values. ACO_R and ROPE have comparable average RMSE for HM based solely on streamflow or salt generation. By contrast, GLM has similar performances to ant-based methods when the objective function targets both responses.

Table 4.14: Average Weighted RMSE for History Matching methods. Methods within the same subscript represent pairs with equal means at the 5% level of significance.

ω	ACO_R	$DACO_R$	ROPE	GLM	p-value
0	1.314_a	1.317_{ab}	1.351_{ab}	1.372_b	0.003
0.25	1.406_a	1.412_{ab}	1.435_b	1.426_{ab}	0.003
0.5	1.413_a	1.414_a	1.449	1.418_a	0.000
0.75	1.239_a	1.240_a	1.283_b	1.257_{ab}	0.000
1	0.632_a	0.632_a	0.632_a	0.643	0.004

Table 4.15 shows that all algorithms provide equal variances (p-value = 0.178) when HM is performed only on salt generation. When both salt and streamflow are used, variances of RMSE for all methods are not the same (p-value is much less than 0.01).

Further, using F-test for pairwise comparisons, with Bonferroni corrections, ACO_R has the highest consistency in replicates, providing the lowest variance compared to all methods, for each weighting. When $\omega = 1$, equal variances are observed in multiple solutionbased methods of $DACO_R$ and ROPE.

Table 4.15: Standard deviation of weighted RMSE for History Matching methods. Variance is based on an F-test for pairwise comparisons with Bonferroni correction.

ω	ACO_R	$DACO_R$	ROPE	GLM	p-value
0	$1.1e-4$	$5.2e-3$	$4.6e-2$	$2.9e-2$	0.178
0.25	$3.3e-4$	$8.2e-3_a$	$1.7e-2_a$	$1.6e-2_a$	0.004
0.5	$1.4e-4$	$6.3e-4$	$1.96e-2_a$	$9.1e-3_a$	0.003
0.75	$2.0e-4$	$7.5e-4$	$2.8e-2$	$5.3e-3$	0.040
1	$2.6e-4$	$1.3e-3_a$	$7.2e-4_a$	$1.1e-2$	0.000

4.4.5.2 Comparison of Individual Predictions for Streamflow and Salt Generation Based on Parameter Calibrations for Weighted RMSE

In order to explore the performance of HM methods on individual responses of streamflow and salt generation, during calibration and validation periods, we use the results of HM from the previous section. We consider the predicted series for streamflow and salt separately for the calibration and validation periods. The results are presented in Table 4.16 for the calibration period and Table 4.17 for the validation period. These results also include the consistency of RMSE values via standard deviation of RMSE. The RMSE values for streamflow and salt generation were calculated using Equation 4.1.

Table 4.16 clearly shows that adding more weight on one response results in a better fit for the corresponding response. When the values of ω weighting on flow increases, the average RMSE of streamflow declines and by contrast, the average RMSE of salt generation gets bigger. Putting more weight on one objective function in multiple objectives gives a trade-off for which one particular output receives better matching than the other. Which is the best combination of weights of calibration depends on the purpose and use of the model so will vary by application.

It is also seen in Table 4.16 that mean RMSE salt generation for ROPE with ω= 0, 0.25, 0.5, 0.75 are slightly higher compared to ACO_R and $DACO_R$, resulting in higher weighted RMSE values in Table 4.14. These results indicate the inability of ROPE to locate the optimal regions in the input parameter space when a term for of salt appears in the objective function. GLM provides trade-offs between salt and streamflow for ω=0.25, 0.5 and 0.75 that maintain the weighted RMSE, comparable with the two ant colony-based optimisation methods (in Table 4.14). However, GLM is unsuccessful in maintaining low weighted RMSE values (Table 4.14) when HM is only used on salt (mean RMSE is 0.169) or streamflow (mean RMSE is 0.009). In terms of result consistency, ACO_R is superior compared to the other methods. On the contrary, GLM provides the highest variation within most of its results (the highest standard deviation).

The best results of HM conducted by industry are presented in last row of Table 4.16. Clearly all methods show an improvement over industry estimates. GLM can improve the accuracy by 11%, whilst multiple solutionbased methods show an improvement in accuracy of over 20%.

Table 4.16: Average RMSE for History Matching Methods on streamflow and salt generation in the calibration period.

ω/	ACO_R		$DACO_R$		ROPE		GLM	
Criteria	Flow	Salt	Flow	Salt	Flow	Salt	Flow	Salt
0								
RMSE	-	0.162	-	0.163	-	0.167	-	0.169
Stdev	-	$1.4e-5$	-	$6.5e-4$	-	$5.7e-3$	-	$3.6e-3$
0.25								
RMSE	0.021	0.166	0.021	0.166	0.022	0.170	0.019	0.175
Stdev	$3.0e-4$	$8.8e-4$	$4.2e-4$	$2.5e-3$	$9.4e-4$	$4.3e-3$	$1.1e-3$	$6.0e-3$
0.5								
RMSE	0.018	0.186	0.018	0.186	0.017	0.196	0.017	0.194
Stdev	$8.9e-5$	$8.0e-4$	$1.9e-4$	$1.7e-3$	$1.3e-3$	$1.5e-2$	$8.8e-4$	$1.0e-2$
0.75								
RMSE	0.012	0.279	0.012	0.282	0.012	0.303	0.013	0.267
Stdev	$9.0e-5$	$2.5e-3$	$1.1e-4$	$3.0e-3$	$7.5e-4$	$2.5e-2$	$1.8e-3$	$4.8e-2$
1								
RMSE	0.008	-	0.008	-	0.008	-	0.009	-
Stdev	$3.51e-6$	-	$1.7e-5$	-	$9.6e-6$	-	$1.5e-4$	-
Industry Estimates for Streamflow [42]								
RMSE	0.010							

How do the performances of HM methods-based model parameters compare in different period of time? Are model parameter results from ant colony optimisation methods still robust within this period? In order to answer these questions, we present Table 4.17, the performance of HM methods in terms of mean individual RMSE and consistency during the validation period. We also compare the HM accuracy provided by the methods with the best estimate obtained by industry.

Table 4.17: Average RMSE for History Matching Methods on Streamflow and Salt Generation in Validation Period

ω/	ACO_R		$DACO_R$		ROPE		GLM	
Criteria	Flow	Salt	Flow	Salt	Flow	Salt	Flow	Salt
0								
RMSE	-	0.106	-	0.097	-	0.116	-	0.139
Stdev	-	$1.8e-3$	-	$6.5e-3$	-	$7.7e-2$	-	$1.6e-2$
0.25								
RMSE	0.017	0.128	0.017	0.129	0.017	0.134	0.015	0.156
Stdev	$2.8e-4$	$2.8e-3$	$2.8e-4$	$7.1e-3$	$8.1e-4$	$1.2e-2$	$1.0e-3$	$1.6e-2$
0.5								
RMSE	0.013	0.183	0.014	0.183	0.014	0.196	0.013	0.195
Stdev	$8.3e-5$	$1.7e-3$	$1.8e-4$	$4.3e-3$	$1.3e-3$	$2.5e-2$	$8.3e-4$	$1.7e-2$
0.75								
RMSE	0.008	0.316	0.008	0.326	0.008	0.345	0.009	0.302
Stdev	$9.0e-5$	$3.2e-3$	$1.0e-4$	$3.9e-3$	$6.8e-4$	$3.2e-2$	$1.7e-3$	$6.2e-2$
1								
RMSE	0.005	-	0.006	-	0.006	-	0.007	-
Stdev	$8.82e-5$	-	$4.7e-4$	-	$9.9e-6$	-	$1.2e-3$	-
Industry Estimates for Streamflow [42]								
RMSE	0.009							

The individual mean RMSE values in Table 4.17 show that the model parameters obtained from ACO_R and $DACO_R$ have comparable performance (in some cases ACO_R slightly outperforms $DACO_R$ and vice-versa). Both ant colony-based optimisation methods marginally outperform the other methods for all ω values. The inability of the model parameters of ROPE to locate better optimal regions in the input parameter space, to reduce mean RMSE using salt, is also reflected in the validation period. These model parameters have higher RMSE for salt on average compared to ACO_R and $DACO_R$ ($\omega = 0, 0.25, 0.5,$ and 0.75). GLM model parameters clearly demonstrate poorer fit for ω=0 (mean RMSE for salt is 0.139) and $\omega = 1$ (mean RMSE for streamflow is 0.07) than the other methods. Moreover, ACO_R is the most consistent method providing the lowest standard deviation, which is in contrast to the GLM results. In conclusion,

model parameters of both ACO_R and $DACO_R$ are robust within the different time periods.

In spite of these different performances, all algorithms are able to increase accuracy compared to industry estimates by more than 22% (ACO_R:44%, $DACO_R$ and ROPE:33%, GLM:22%) during the validation period.

4.4.5.3 Computational Time for Convergence of HM Methods

The average computational time and number of model runs required for convergence of each HM method are listed in Table 4.18. One-way ANOVA was performed for comparing the average computational time required by each algorithm for convergence and the results are presented in Table 4.18.

Table 4.18: Computational time (in seconds) and the number of models required by HM methods for convergence.

	ACO_R		$DACO_R$		ROPE		GLM		p-value
ω	Time	Nmodel	Time	Nmodel	Time	Nmodel	Time	Nmodel	
0	4,735	400	7,432	675	62,921	6000	1,292	124	0.000
0.25	5,728	492	7,729	708	62,976	6000	3,092	293	0.000
0.5	$5{,}645_a$	483	$7{,}014_a$	617	62,295	6000	2,643	249	0.000
0.75	$6{,}113_a$	525	$6{,}748_a$	592	63,965	6000	3,130	291	0.000
1	$4{,}657_a$	375	$4{,}430_a$	367	62,723	6000	1,691	217	0.000
Time required by industry for HM					576,000				

Table 4.18 shows that at least one pair of HM methods has significantly different mean computational time for each ω value (p-value is 0.000). Among HM methods, GLM requires the least time for convergence (less than one hour) with at most about 300 model runs on average. By contrast with GLM, ROPE requires extensive computational time, requiring nearly a day to complete HM. $DACO_R$ for most of ω values (except for $\omega = 1$) converges with a larger number of model runs compared to ACO_R, since $DACO_R$ includes local and global searches to maintain diversity within its solutions. According to Tukey's comparison test, the computational time required by both ant-based optimisation methods are comparable when ω values increase. Table 4.18 also reveals that ACO_R and GLM take more models to converge when both responses (streamflow and salt generation) are present in the objective function.

4.4.5.4 Final Input Parameter Ranges

In the following figures we present the 95% Confidence Intervals (CIs) for the optimal regions of the input parameters obtained from the four HM methods for each ω value. The 95% CIs of input parameters in these figures were obtained from the six repeated experiments, the best input parameter set from each experiment for multiple solutionbased method and the six input parameter sets of GLM. Figure 4.15 to Figure 4.18 depict the region of the optimal six input

parameters for $\omega = 0, 0.25, 0.5,$ and 0.75. The last figure (Figure 4.19) excludes parameter C_u.

First of all, most of the HM methods show similar directions locating optimal regions in the input parameter space for most input parameters (Figure 4.15 to Figure 4.19). ACO_R successfully narrows down the initial uncertainty of the input parameters to very narrow 95% CIs (Figure 4.15 to Figure 4.19). Meanwhile, $DACO_R$ is able to provide significant reductions of initial input parameter ranges, especially for $\omega = 0$ and 0.5 (Figure 4.15 and Figure 4.16 respectively). Both ant colony optimisation-based methods become less consistent when HM is solely based on streamflow (Figure 4.19). In spite of these differences, both ACO_R and $DACO_R$ demonstrate high agreement in finding the location of the optimal input parameter regions.

In contrast to previous findings, ROPE and GLM are inconsistent in providing optimal input parameter regions (Figure 4.15 to Figure 4.19). ROPE demonstrates high variation, indicated by wide 95% CIs for most input parameters (Figure 4.15 to Figure 4.18). This variation, however, becomes less when the focus of HM is only on streamflow (Figure 4.19). These results reveal that ROPE becomes less capable of finding the optimum when the objective function includes salt generation. Its consistency turns worse when both streamflow and salt generation are present in the objective function. Unlike ROPE, GLM shows wider 95% CIs, mostly on parameter *ia* (Figure 4.15 to Figure 4.19). These 95% CIs are quite different when the focus of HM is on ω=0 (Figure 4.15) and 0.25 (Figure 4.16). The inability of GLM to locate a better parameter region compared to the other methods may be an effect of sensitivity of GLM to the selection of starting points for particular objective functions (when ω= 0 and 1). As in this study, the six starting points were randomly selected from a set of 50 input parameters generated by LHS. The selected points that are far from the optimal region may trap GLM in different local minimum points of the objective function and result in high variation within its results.

Further we calculated the percentage reductions of initial input parameter uncertainty for all HM methods. These results are listed in Table 4.19. As can be seen in Table 4.19, multiple solution-based methods significantly narrow the predefined input parameter ranges from 75% to 100%. For HM based only on streamflow ($\omega = 1$), GLM is less successful in reducing input parameter uncertainty than the other methods. These results reveal that the selection of the initial input parameter set is very important for GLM, especially for streamflow-only based HM. Far starting points far from the optimal ones, GLM may be trapped in local minima of the objective function and, eventually, result in high variation in its performance.

4.4.5.5 Reduction of Uncertainty in Initial Streamflow and Salt Generation (ACO_R Results)

Figure 4.20 shows the reduction in uncertainty of initial average streamflow (top picture) and salt generation (bottom picture) for different ω values in the objective function used within ACO_R for HM. The initial ranges of average responses were obtained by permuting the lowest and highest boundaries of the initial input parameters in Table 3.1. This provides 64 combinations each for streamflow and salt generation. By running the LUCICAT model based on these combinations, we obtained 64 average streamflow (m^3/year) and 64 average salt generations (kg/year). Then,

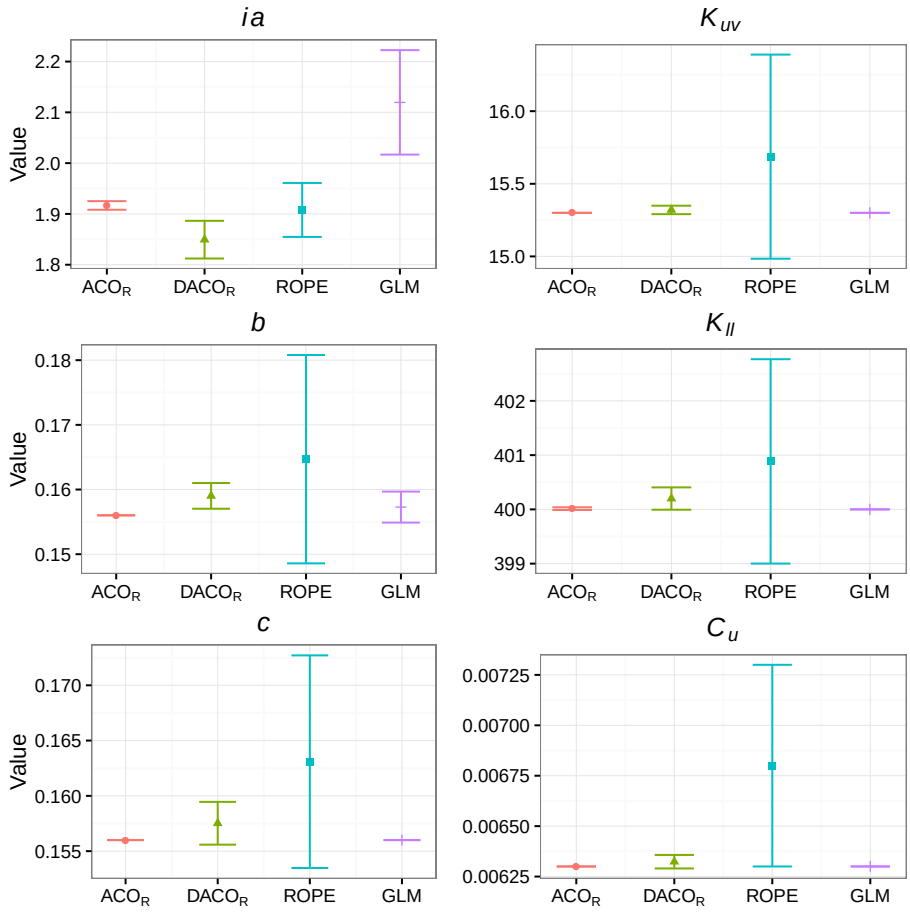

Figure 4.15: 95% Confidence intervals of HM method-based input parameters for $\omega = 0$.

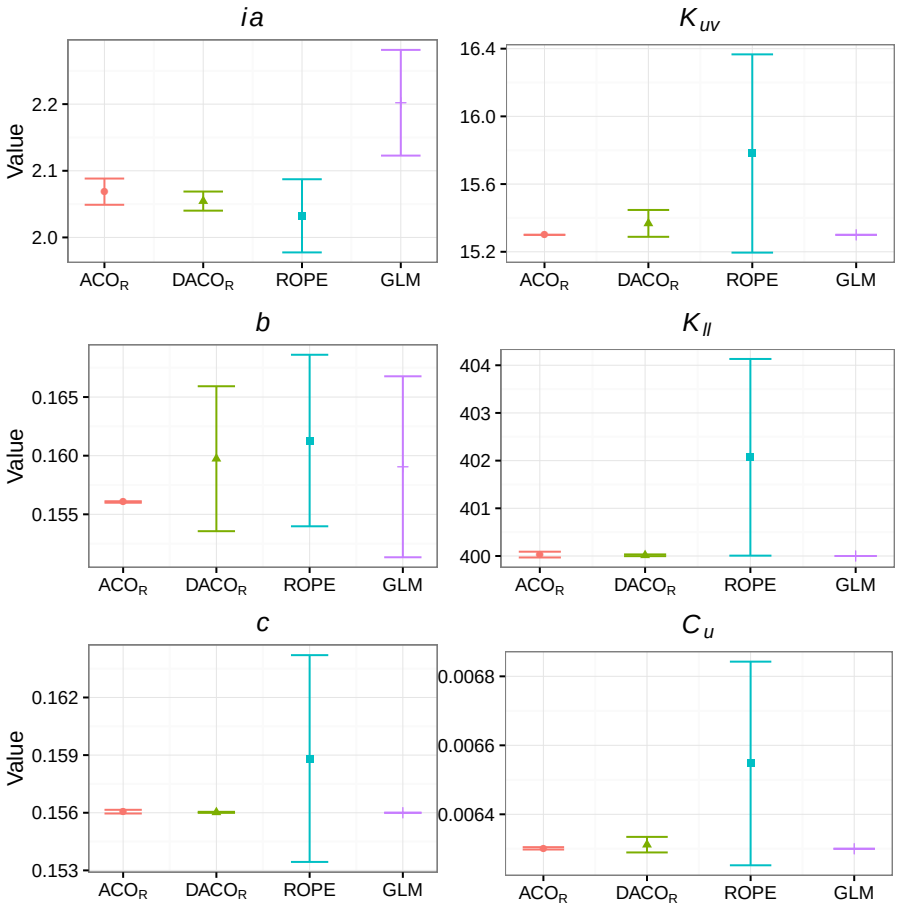

Figure 4.16: 95% Confidence intervals of HM method-based input parameters for $\omega = 0.25$.

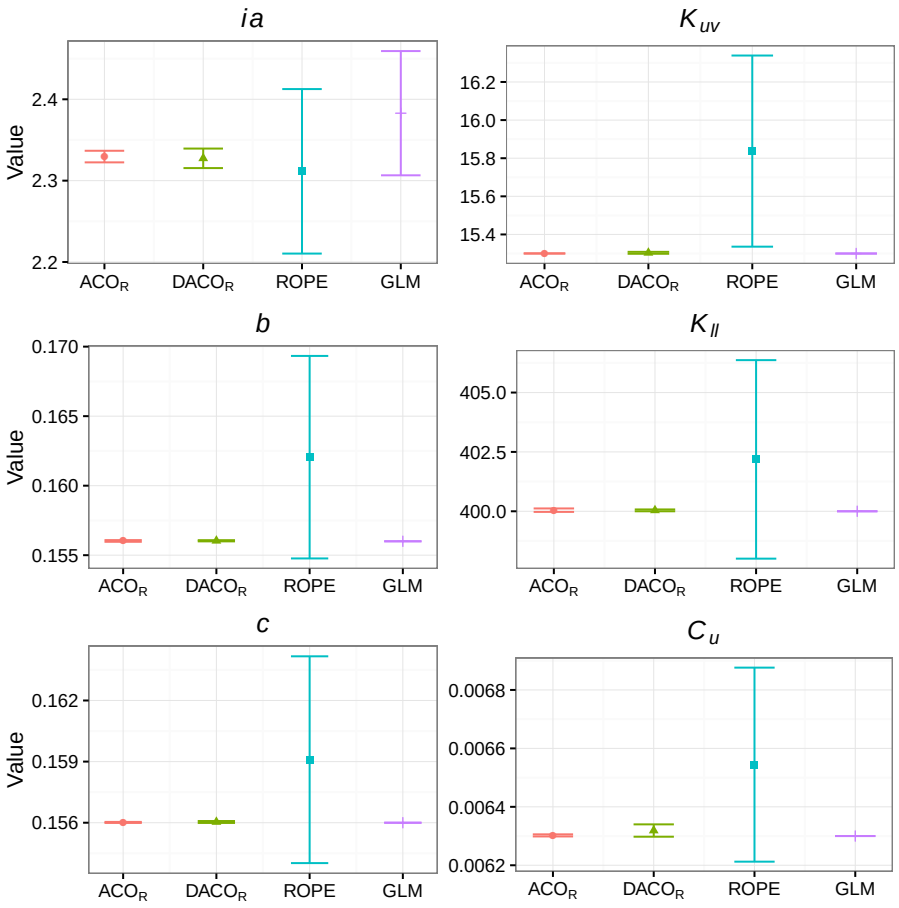

Figure 4.17: 95% Confidence intervals of HM method-based input parameters for $\omega = 0.5$.

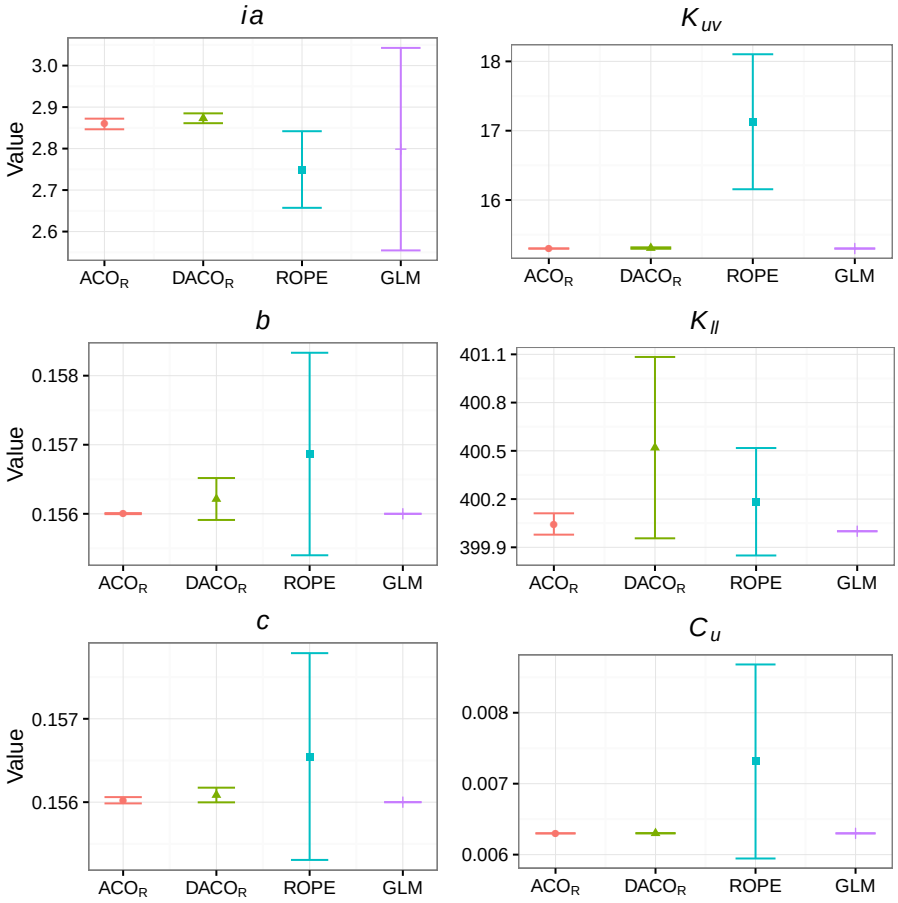

Figure 4.18: 95% Confidence intervals of HM method-based input parameters for $\omega = 0.75$.

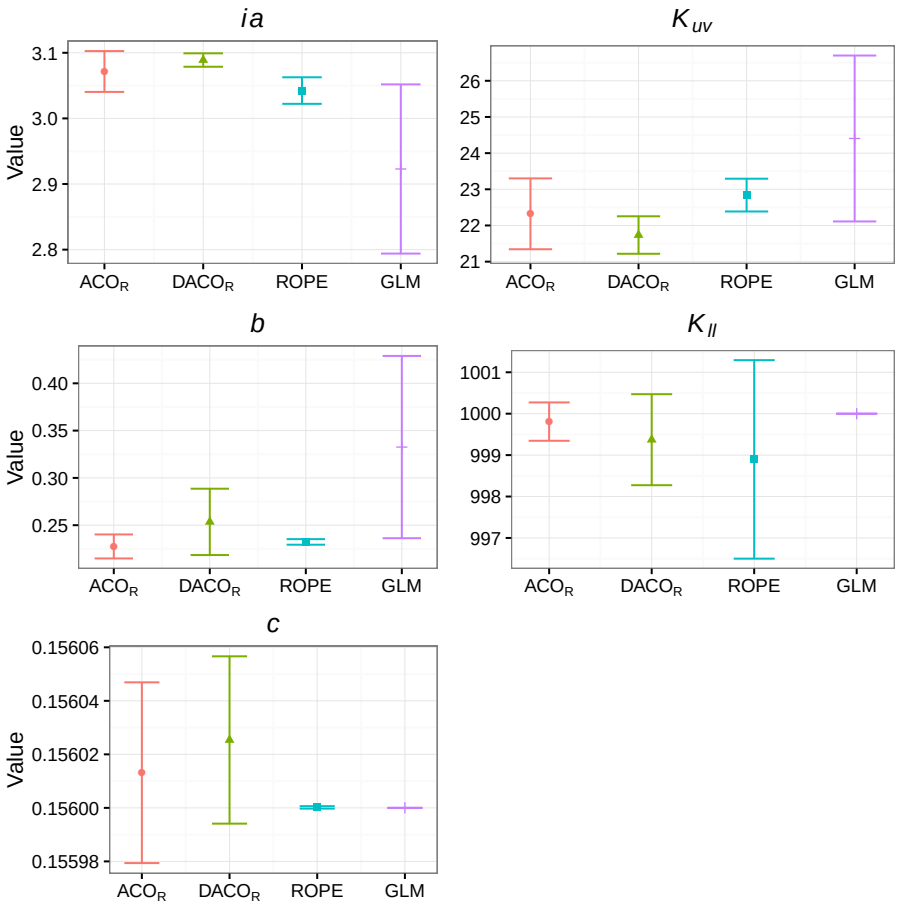

Figure 4.19: 95% Confidence intervals of HM method-based input parameters for $\omega = 1$.

Table 4.19: Percentages of reduction in initial input parameter uncertainty.

Method	ω	ia	K_{uv}	b	K_{ll}	c	C_u
ACO_R	0	98.2	100	100	100	100	100
	0.25	96.2	100	100	100	99.9	100
	0.5	98.8	100	100	100	100	100
	0.75	97.4	100	100	100	100	100
	1	93.9	78.9	92.0	99.8	100	-
$DACO_R$	0	90.9	99.3	98.6	99.9	98.6	99.5
	0.25	96.5	98.1	95.3	100	100	99.7
	0.5	97.1	99.9	100	100	100	100
	0.75	96.8	99.7	99.8	99.6	98.9	100
	1	94.7	86.2	75	99.2	100	-
ROPE	0	89.0	86.1	90.2	99.3	94.1	93.4
	0.25	88.9	88.1	95.3	99.3	96.8	96.8
	0.5	79.6	89.2	95.8	98.4	97.4	96.3
	0.75	79.5	80.8	99.2	99.9	99.3	83.6
	1	80.7	86.1	90.2	99.3	94.1	-
GLM	0	81.2	100	98.5	100	100	100
	0.25	84.7	100	95.5	100	100	100
	0.5	85.0	100	100	100	100	100
	0.75	62.7	100	100	100	100	100
	1	75.5	52.3	47.4	100	100	-

the minimum and maximum values of average streamflow and average salt were determined to obtain the ranges of initial average streamflow and salt as depicted in Figure 4.20.

From every ω value, ACO_R produced six model parameters (each model was selected from the best performing parameters obtained from each run). These model parameters were used in LUCICAT to provide ranges of average streamflow and salt generation. These ranges are depicted in Figures 4.20 as $\omega = 0, 0.25, 0.5, 0.75,$ and 1. Figures 4.20 also depict the percentile of the ranges of streamflow and salt generation with respect to initial average streamflow and salt.

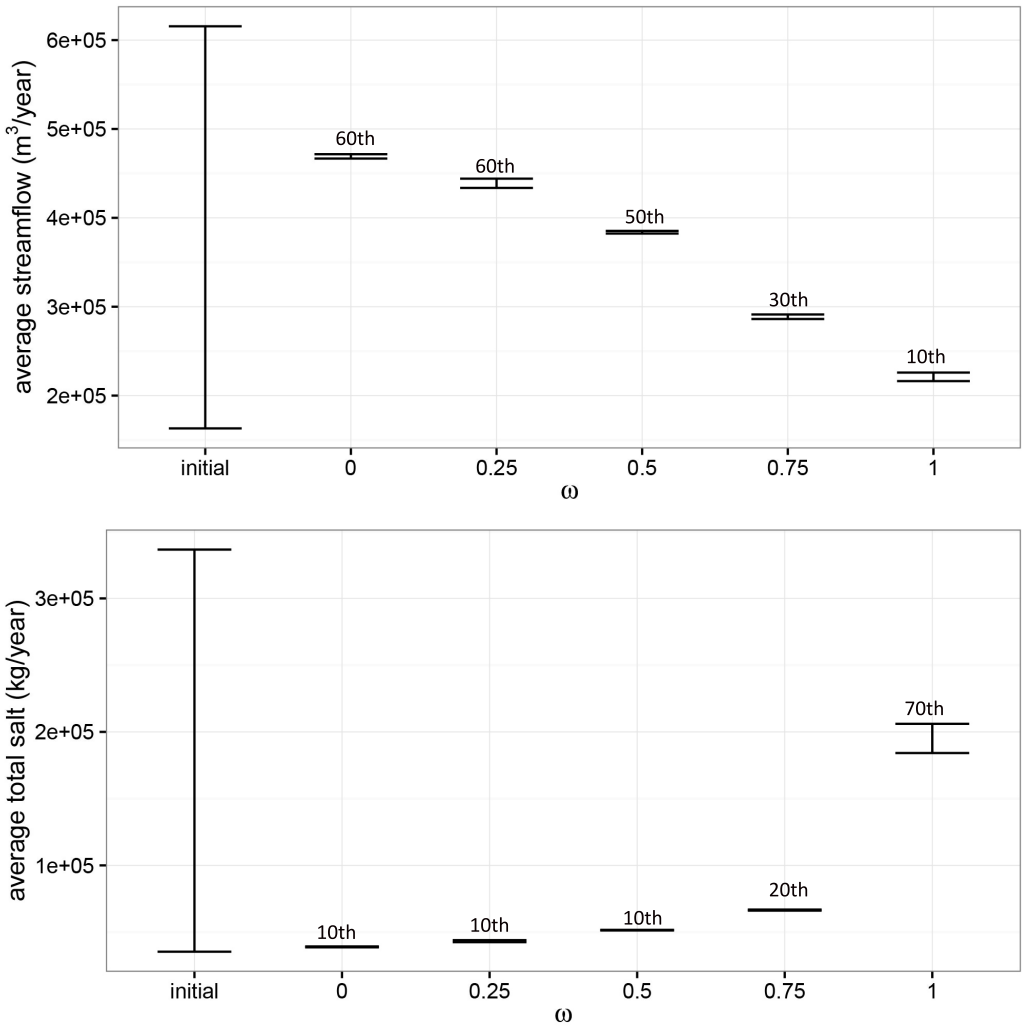

Figure 4.20: Reduction of initial streamflow and salt generation by ACO_R.

Figure 4.20 shows the obvious significant reduction in initial average streamflow and salt generation. The reduction in initial average streamflow and salt generation is more than 97% and 92% respectively. The lowest reduction is observed when HM focuses on streamflow only. Average streamflow values for $\omega = 0$ and 0.25 lie in the 60^{th} percentile and gradually reduce to the 10^{th} percentile as ω increases. On the other hand, average streamflow for the first three ω values appear in the 10^{th} percentile and move up to the 70^{th} as streamflow becomes the HM focus.

These results can be used by hydrologists for planning water management, including designing water storage, treament and distribution systems. Over- or under-prediction of streamflow and salt may lead to failure in the design of water facilities and may lead to considerable maintenance costs.

4.5 Concluding Remarks

This chapter introduced direct HM using a single-local deterministic approach, and multiple solutionbased methods. The former method was represented by GLM, while the latter were ant-based optimisation methods for continuous domains (ACO_R and $DACO_R$) and ROPE. The purpose of HM was to calibrate input parameters (up to six parameters) for one relatively big catchment (The Dandalups) and two small catchments (Bates and Lewis) in Western Australia. Input parameters were adjusted to match the available historical data, i.e., streamflow only for The Dandalups and Bates, and streamflow and salt generation for Lewis. The performance of these HM methods was compared in terms of accuracy via average RMSE, consistency via standard deviation of RMSE, computational time and reduction of initial input parameters uncertainty. The performances of HM results were also compared with the model accuracy provided by best industry estimates to date.

Among the implemented algorithms, ant-based optimisation methods performed favourably well in every catchment. ACO_R and $DACO_R$ showed similar performances (RMSE) to ROPE in The Dandalups for peak and daily streamflow series-based HM procedures. ACO_R and $DACO_R$ had comparable accuracy (RMSE) with ROPE for Bates catchment. These algorithms showed better performance compared to GLM in Bates. Moreover, ACO_R and $DACO_R$ demonstrated their superiority in terms of average weighted RMSE in Lewis catchment for HM based on both streamflow and salt generation. In addition to the greater accuracy, both ant-based optimisation methods showed high consistency in their results and are able to reduce initial input parameter uncertainty by more than 50% across catchments. Besides, for Bates and Lewis catchments, the input parameters obtained from all HM methods during the calibration period were also robust within the validation period. The application of ACO_R and $DACO_R$ also requires low to moderate computational time depending on the size of catchment.

On average, ROPE performed comparably well with ACO_R and $DACO_R$ across catchments. In Bates, ROPE was even able to find model input parameters that were also robust during the validation period, resulting in the highest accuracy (the lowest average RMSE) and consistency compared to the other algorithms. In Lewis catchment, ROPE did not perform well when salt generation was included in the objective function. The application of ROPE in all catchments was reported to have required a high computational time. On average, ROPE was able to contribute significant reduction in initial uncertainty of more than 50% across the catchments.

GLM was the most efficient method, requiring the least computational time among HM methods examined in this study. GLM generally performed poorly when the focus of HM is solely based on one response, i.e., streamflow (HM on annual daily peaks for The Dandalups,

daily time series for Bates and Lewis catchments) or salt (Lewis catchment). By contrast, GLM had comparable average performance (RMSE) with ACO_R and $DACO_R$ in Lewis catchment when streamflow and salt responses were included in the objective function. Moreover, in most scenarios GLM showed its inconsistency within its results due to its tendency to be caught near the initial points. Across catchments, GLM can reduce the initial uncertainty of input parameter ranges by more than 45%.

In spite of these differences, all HM methods were able to improve the accuracy in RMSE provided by industry estimates by 10-50%.

All computation in this chapter required execution of LUCICAT at each iteration. Methods like ROPE require 1000 runs of LUCICAT in the optimisation process, making this process computationally prohibitive, especially for large catchments. A simple solution is to replace LUCICAT by an easy to compute proxy model for use within the optimisation process. Chapter 5 will explore this concept.

CHAPTER 5

Application of Indirect History Matching Procedure in Calibration of Hydrological Model Parameters

5.1 Introduction

In chapter four we explored the performance of direct history matching (HM) using the LUCICAT hydrological model. In HM, the LUCICAT model is repeatedly executed to provide the model response at given input settings. The simulated response is then compared with the observed response. The aim of HM methods is to minimise the difference between the simulated and observed responses, by perturbing the input parameters, and re-executing LUCICAT. Typically these methods require several hundred runs of LUCICAT, where computational time required for each run may vary from 30 seconds to more than one minute depending on the size of catchments. For the catchments presented in Chapter 4, HM time for GLM varied between 0.8 and 1.5 hours. On the other hand, for multiple solution-based approaches, such as ACO_R, $DACO_R$ or ROPE the time required varied between a few hours to a few days. For larger catchments, complex hydrological models, such as a fully distributed physically-based hydrological model, MIKE-SHE [44], and in other applications such as history matching of hydrocarbon production based on 3D reservoirs reservoir model [200], the execution time of a computer simulation model may be very large. Hence, implementation of HM in the current setup for these cases could be computationally challenging. In these scenarios we can use indirect HM presented in this chapter.

Under indirect HM, first a proxy model is developed for LUCICAT. Then during the process of HM this proxy model is used within each iteration. For this approach to work the proxy model should be accurate and computationally efficient. With this requirement in mind a surrogate model is developed via multidimensional Kriging [85] and artificial neural networks (ANN) [39,55]. The background on these methods is provided in Chapter 2. For HM, we will apply ant-based optimisation methods for continuous domains (ACO_R and $DACO_R$), Robust Parameter Estimation (ROPE) and Gauss Levenberg Marquardt (GLM). Five model parameters, namely

ia, K_{uv}, b K_{ll}, and c for The Dandalups are selected to be calibrated to match annual daily peaks of historical streamflow.

The objectives of this chapter are:

(i) To investigate the appropriateness of surrogate models for HM based on model accuracy, the size of training data, the model complexity, and the computational time.

(ii) To investigate the performance of indirect HM in terms of accuracy via root mean square error (RMSE), consistency (standard deviation) and computational time.

(iii) To compare the HM results obtained from the indirect and direct approaches, in terms of the accuracy via root mean Square error (RMSE), consistency within results, computational time, and the reduction of initial input parameter uncertainty.

5.2 The Development of the Surrogate Models

To develop surrogate models, we first generated a set of input parameters $X^T = (x^1, x^2, ..., x^n)^T$, where $(x^i) = (x^i_1, x^i_2, ..., x^i_5)$ for $i = 1, 2, ..., n$. Here we set n to be 50 and 100 for all methods. The LUCICAT model is then executed on this set X to generate the model response $Y^T = (y^1, y^2, ..., y^n)^T$, where $(y^i) = (y^i_1, y^i_2, ..., y^i_{33})$ for $i = 1, ..., n$. The input parameter values, X and the responses Y become a set of data to train a proxy model ($Y_s = g(X)$). Thus the proxy model is independently trained on X and each response y_j where $j = 1, ..., 33$. Therefore, there will be 33 independent surrogate models corresponding to each annual peak. The function $g(\cdot)$ is computed by methods of multidimensional Kriging or artificial neural network (ANN).

The performance of a surrogate model is evaluated by using testing data sets containing a set of input parameters $(x^k_t) = (x^k_{t1}, x^k_{t2}, ..., x^k_{t5})$ and a corresponding set of responses $(y^k) = (y^k_1, y^k_2, ..., y^k_{33})$ for $k = 1, ..., 100$. The performance of each surrogate model (at each j) was evaluated by:

$$RMSE_j = \sqrt{\frac{\sum_{k=1}^{100}(Y^s_k - Y^{LUC}_k)^2}{100}} \tag{5.1}$$

where Y^s_k and Y^{LUC}_k are the k^{th} predicted and the LUCICAT response for j^{th} annual peak.

The overall performance of a surrogate model was calculated using :

$$\bar{RMSE} = \frac{\sum_{j=1}^{33} RMSE_j}{33} \tag{5.2}$$

The surrogate model was then used to replace the LUCICAT model within the process of HM. The HM methods aim to minimise the differences between the surrogate model response, $(Y^s) = (Y_1, Y_2, ..., Y_{33})$ and observed values of $y = y^h = (y^h_1, y^h_2, ..., y^h_{33})$ via RMSE defined as follows:

$$RMSE = \sqrt{\frac{\sum_{j=1}^{q}(Y^h_{flow_j} - Y^s_{flow_j})^2}{q}} \tag{5.3}$$

where $Y^h_{flow_j}$ and $Y^s_{flow_j}$ are the j^{th} observed and model streamflow respectively.

5.3 Sensitivity Analysis Test to Find a Good Proxy Hydrological Model

To develop a good proxy model for LUCICAT we considered multidimensional Kriging and ANN methodology. These methods are suited for modelling data from computer simulated experiments. For each of these methods we explore the sensitivity to size of the data set required, model complexity and accuracy in mimicking the hydrological model.

5.3.1 Kriging Models

In computer simulated experiments, multidimensional Kriging is a popular method that can recognize complex model patterns effectively. Multidimensional Kriging uses the adjusted error of nearby points to improve the predictions of the model. The method has a common mathematical formulation expressed as follows:

$$y = \mu + Z(x) \tag{5.4}$$

where $Z(x)$ is realization of a stochastic process with mean zero and covariance $V(r,s) = \sigma^2 R(r,s)$ between $Z(r)$ and $Z(s)$, σ^2 is the process variance and $R(r,s)$ is the correlation. A common form of correlation function is the Gaussian family:

$$R(X_i, X_j) = \Pi_{k=1}^{p} exp(-\theta_k |X_{ik} - X_{jk}|^2) \tag{5.5}$$

where $\theta_k > 0 \ \forall k$.

The unknown parameters in the correlation function are estimated by maximising the likelihood function:

$$-\frac{1}{2}(nln(\hat{\sigma})^2 + ln|R|) \tag{5.6}$$

where

$$\hat{\sigma^2} = \frac{1}{n}(\mathbf{y} - \mathbf{f}\hat{\boldsymbol{\beta}})^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{f}\hat{\boldsymbol{\beta}}) \tag{5.7}$$

and

$$\hat{\beta} = (\mathbf{f}^T \mathbf{R}^{-1} \mathbf{f})^{-1} \mathbf{f}^T \mathbf{R}^{-1} \mathbf{y}, \tag{5.8}$$

Welch et al. [184] and Na-Udom [11] proposed an algorithm to obtain the MLE for $(\theta_k : 1, ..., p)$.

The Settings for Developing Multidimensional Kriging Models

Outlined below are the major steps in fitting a multidimensional Kriging model:

- Training data sets contain a matrix of input parameter, $\mathbf{X}$, of size of $n \times p$ and a matrix of responses, $\mathbf{Y}$, of size $n \times q$, where n was selected to be 50 and 100, p was five input parameters and q was the 33 annual daily peaks of the LUCICAT modelled streamflow. In the multidimensional Kriging model, the input parameters were normalized with respect to the original boundaries of the input parameters. The training data size was restricted

to 100, as a larger size is computationally extensive and will not work on the economy of scales.

- The input parameter settings, $\mathbf{X}$, were generated using LHD based on S-optimality (maximising the mean distance from each design point to all the other points in the design). We used the lhs.design function from the DoW.wrapper R package [178].
- The response setting for 33 annual daily peaks of streamflow , $\mathbf{Y}$, was obtained by executing LUCICAT for each row of $\mathbf{X}$, representing the settings of input parameters.
- We used the algorithm proposed by Na-Udom [11] to find the MLE of θ in Equation (5.5).
- The multidimensional Kriging model was trained separately for each annual daily peak. This resulted in 33 independent multidimensional Kriging models, for which five θ parameters $(\theta_1, ...\theta_5)$ were estimated for each annual daily multidimensional Kriging model in relation with the corresponding input parameters. The performances of 33 independent models were evaluated by using a set of testing data with size of 100 (the development of testing data underwent the same process as training data). The RMSE of each model of 33 models were calculated using Equation (5.1). The overall performance of a multidimensional Kriging model was computed by averaging the RMSE of 33 models (Equation (5.2)).
- The experiment was conducted three times using different sets of training data for each size of training data. The average RMSE, standard deviation, and computational time were recorded.

Multidimensional Kriging Model Results

We present the results of the sensitivity testing for multidimensional Kriging models in Table 5.1.

Table 5.1: Comparison of multidimensional Kriging models developed training data of size 50 and 100

	training data size	
	50	100
RMSE	0.185	0.129
stdev	0.022	0.004
computational time (s)	1,931	12,358

Table 5.1 shows that on average the multidimensional Kriging model developed from training data size of 100 provides better fit (p-value = 0.011) compared to the model obtained from a training data size of 50. It is also noticed that higher consistency within RMSE is provided by the sample of size 100. This is expected from the sample size increase. However, the computational time required to develop a multidimensional Kriging model dramatically increases. For

doubling the sample size from 50 to 100, computational time increases 6-fold.

The distribution of the parameters for the best performing multidimensional Kriging model $(\theta_k : 1, ..., p)$ for each training data of size 50 and 100 are depicted in Figure 5.1; θ_1, θ_2, θ_3, θ_4, and θ_5 represent the input parameters ia, K_{uv}, b, K_{ll}, and c respectively.

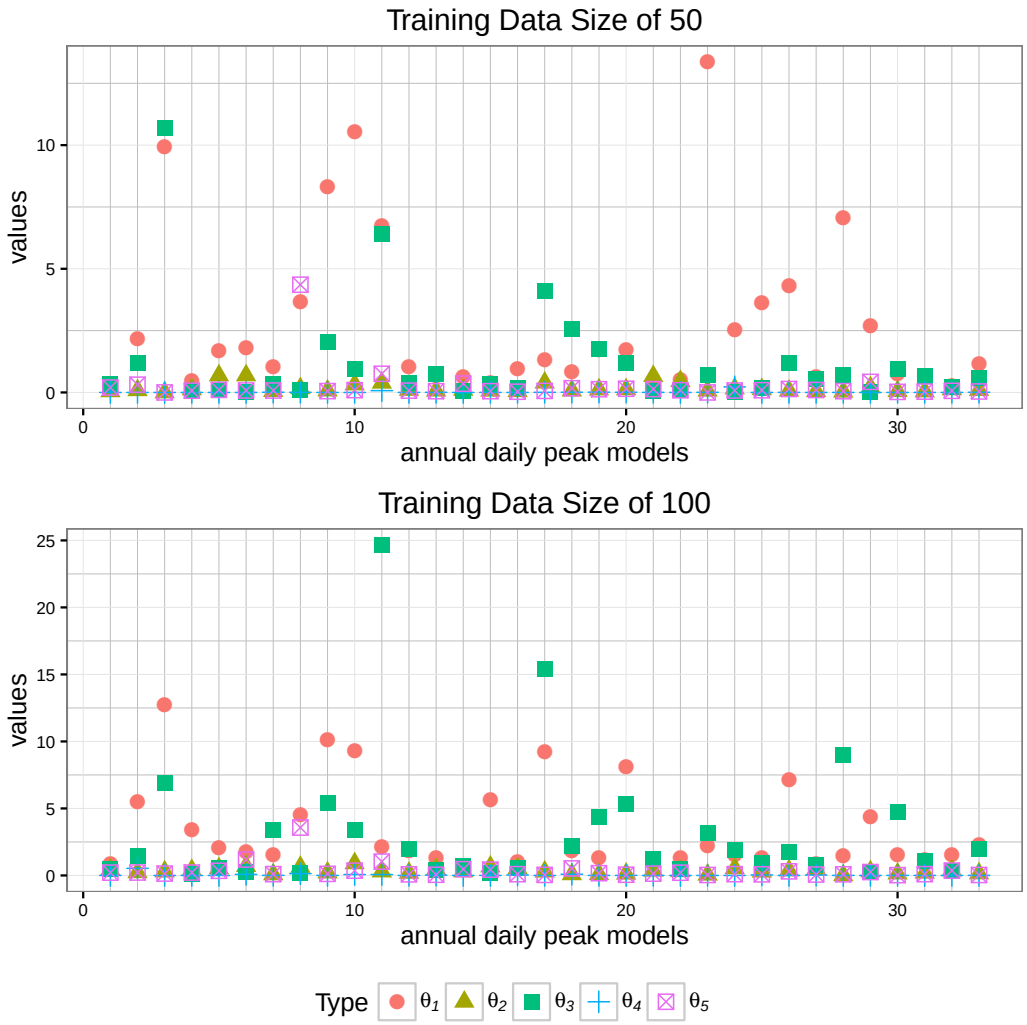

Figure 5.1: The distribution of theta for 33 multidimensional Kriging models developed from 50 and 100 training data size

.

Observing Figure 5.1, θ_2 , θ_4 and θ_5 are concentrated on the area of the lowest boundary of zero, while θ_1 and θ_3 of the LUCICAT model are more spread out and take higher values. This may indicate these input parameters have greater influence on the modelled response.

5.3.2 Artificial Neural Network (ANN)

ANN is one of the most popular and powerful tools in machine learning. The ability to identify a relationship between large input parameters and response makes ANN appropriate for modelling complex phenomenon [3,36,172]. Bandyopadhyay and Bhadra [3] claim that ANN has provided better solution in complex systems that are difficult to understand, in problems involving noise, pattern recognition or predictive modelling, and when there are incomplete and ambiguous inputs.

An ANN is developed through a learning process. The training data is fedforward through an ANN and the resulted output is compared with the real output. The differences between ANN output and real output model are iteratively evaluated by learning algorithms to update the sypnatic weights, also known as ANN parameters. Two learning algorithms, namely back-propagation (Backprop) and resilent backpropagation (Rprop) are explored in this study. The brief theory of Backprop and Rprop have been described in Chapter 2.

The Settings for Developing ANN Models

The development of ANN models followed the procedure below:

- We used the nnet R package [185] and the neuralnet R package [163] to develop ANN models. The nnet R package is based on the Backprop learning algorithm whilst Neuralnet uses the Rprop algorithm..

- Training data sets contained a matrix of input parameter, $\mathbf{X}$, of size of $n \times p$ and a matrix of responses, $\mathbf{Y}$, of size of $n \times q$, where n was selected to be 50, 100 and 200, with $p = 5$ and $q = 33$. As in the multidimensional Kriging model, the input parameters for ANN were normalized with respect to the original boundaries of the input parameters.

- The input parameter settings were generated using LHD based on S-optimality (maximising mean distance from each design point to all the other points in the design). The lhs.design function in the DoW.wrapper R package [178] was used for this purpose. The response setting was collected from the 33 annual daily peaks of streamflow, obtained from executing the LUCICAT model on a corresponding set of input parameters, $\mathbf{X}$.

- There is no specfic rule for selecting the number of layers for ANN [3]. Therefore, it is required to perform experiments to determine the optimal number of layers. Hornik *et al.* [95] suggest that one hidden layer is sufficient to develop a model of any piecewise continuous function. However, in this study we considered up to three hidden layers for ANN with the Rprop learning algorithm, following a study conducted by Bandyopadhyay and Bhadra [3]. For ANN with Backprop, only one hidden layer for ANN with Backprop was applied (due to a limitation of the nnet package). We denote the first, second, and third hidden layer as h1, h2, and h3 respectively. The scenarios of the number of neurons in each hidden layer for Backprop and Rprop ANN are listed in Table 5.2.

- The general model of ANN for this study is depicted in Figure 5.2. Five input parameters are passed to one, two or three hidden layers to generate one annual daily peak, for a specified year. We independently repeated the process for 33 each year. This led to the development of 33 independent models for each year. The performances of 33 independent models were then evaluated using a testing data set of size 100 (the development of testing data underwent the same process as the development of training data). The RMSE of each model in 33 models were calculated using Equation (5.1). Then the RMSE values of 33 ANN models were averaged to provide the overall performance of a ANN model (Equation (5.2)).

- The experiment for each scenario was conducted three times using different sets of training data for each size of training data. The average RMSE, standard deviation, and computational were evaluated.

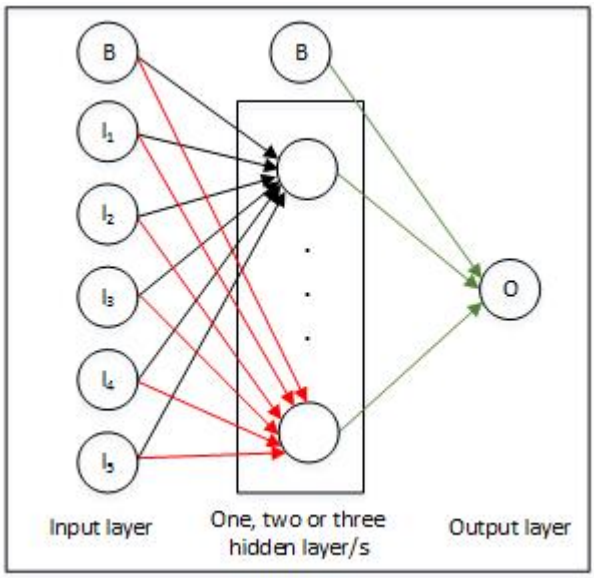

Figure 5.2: General model of ANN models

Backprop ANN Results

Table 5.3 presents average RMSE and variation in RMSE for 10 different ANN models corresponding to different numbers of nodes for training data of size 50 and 100 respectively. P-values of ANOVA shown in the last row of Table 5.3 indicate that average RMSE across the number of nodes is not the same for each specified training data size (N=50:p-value=0.003, N=100:p-value=0.000).

According to Tukey's multiple comparison test, for ANNs based on training data of size 50, the number of nodes 30 and 40 perform differently on average from the other models. For ANN based on training data of size 100, the larger numbers of nodes, such as 80, 90, 100 have different

Table 5.2: Number of neurons in each hidden layer for Backprop and Rprop ANN

Number of hidden layers	number of neurons h1	h2	h3	number of ANN models	learning algorithm
1	10			10	bprop and Rprop
1	20				
1	...				
1	100				
2	10	10		100	Rprop
2	10	20			
2	...	...			
2	20	10			
2	20	20			
2	...	...			
2	100	10			
2	100	20			
2	...	...			
2	100	100			
3	10	10	10	1000	Rprop
3	10	10	20		
3	...	...	...		
3	20	10	10		
3	...	...	...		
3	20	20	20		
3	...	...	...		
3	100	10	10		
3	100	10	20		
3	...	...	...		
3	100	100	100		

average performances from the remaining models. In this case smaller numbers of nodes give favourable results. It is also observed that variation in RMSE is not equal across the number of nodes for ANN models developed by both training data sizes (P-values 0.000). With the number of nodes at 30 and 40 and training data size of 50 there is higher variation compared to the remaining models for the same data size. Conversely, with the number of nodes at 30, 40, and 50 there is higher consistency for training data of size 100.

Rprop ANN Results

Next we conduct a sensitivity analysis of the modelling accuracy of LUCICAT for all for scenarios listed in Table 5.2 for ANN methodology via the Rprop learning algorithms. The results are presented in Table 5.4, Table 5.5, and Table 5.6 for one, two and three layers respectively.

Table 5.3: Average RMSE and standard Deviation of ANN Models-Backprop with one hidden layer
Methods within the same subscript represents pairs with equal means RMSE at 5% level of significance based on Tukey's multiple comparison.
Variance is based on an F-test for pairwise comparison, with Bonferroni correction.

Number of nodes in	training data size 50		100	
h1	mean	stdev	mean	stdev
10	0.235_a	0.013_a	0.457_b	0.212_b
20	0.224_a	0.008_a	$\mathbf{0.170}_b$	0.012_b
30	0.870_b	0.285_b	$\mathbf{0.127}_b$	0.001_a
40	0.646_b	0.557_b	0.118_a	0.003_a
50	0.211_a	0.016_a	$\mathbf{0.117}_b$	0.000_a
60	$\mathbf{0.196}_a$	0.009_a	0.508_b	0.326_b
70	0.201_a	0.005_a	0.717_a	0.294_b
80	$\mathbf{0.197}_a$	0.007_a	1.761_a	0.230_b
90	0.202_a	0.011_a	1.797_b	0.703_b
100	0.204_a	0.014_a	1.434_b	0.726_b
P-value	0.003	0.000	0.000	0.000

Table 5.4 for one hidden layer shows that within each training data size, 30 or more nodes give comparable average RMSE. For training data of size 50, the standard deviation of RMSE is comparable, implying consistency in the performance of the algorithm. For training data of size 100, there is no obvious pattern. Further comparing training data size across each node, comparable performances are seen for 10 and 20 nodes. However, when the number of nodes increases, the performance of training data size 100 outperforms training data size 50.

Table 5.5 for two hidden layer shows comparable results within each training data size (average RMSE and standard deviation in RMSE). Further, comparing training data size across each node, training data size of 100 has smaller average RMSE and should be preferred. Similar results are reported in Table 5.6 when three hidden layers are used within the ANN.

Which one is the best performing ANN-Rprop model? ANN model with one, two or three hidden layers? We selected the best performing ANN model (the lowest mean RMSE) in each layer for each data size, and compare these scenarios for average RMSE, standard deviation of RMSE and computational time. Results are presented in Table 5.7.

Table 5.7 shows all ANN models have similar average ANN-Rprop performances across number of hidden layers for 50 training data of size (p-value = 0.311), whereas this is not the case for size 100. In this case, it is observed that there are significant differences among mean RMSE for different numbers of hidden layers. Tukey's multiple comparison shows that ANN (from training data size of 100) with two and three hidden layers have comparable performances and outperform ANN with one layer. In spite of this ANN models with different numbers of hidden

Table 5.4: Average RMSE and standard deviation of ANN models-Rprop with one hidden layer
Methods within the same subscript represents pairs with equal means RMSE at 5% level of significance based on Tukey's multiple comparison.
Variance is based on F-test for pairwise comparison with Bonferroni correction.

Number of nodes in	training data size 50		100	
h1	mean	stdev	mean	stdev
10	0.258	0.019	0.220	0.036_b
20	0.221_a	0.017	0.192	0.025_b
30	0.211_a	0.006	$\mathbf{0.160}_a$	0.006_a
40	0.219_a	0.006	$\mathbf{0.146}_a$	0.001_a
50	0.210_a	0.006	$\mathbf{0.144}_a$	0.000_a
60	0.207_a	0.014	$\mathbf{0.152}_a$	0.006_a
70	0.207_a	0.012	0.149_a	0.028_b
80	0.212_a	0.011	$\mathbf{0.145}_a$	0.004_a
90	0.207_a	0.019	$\mathbf{0.155}_a$	0.002_a
100	0.210_a	0.001	$\mathbf{0.151}_a$	0.002_a
p-value	0.011	0.053	0.000	0.000

layers and training data sizes show consistency in standard deviation of RMSE (p-value = 0.531 for data size 50 and p-value = 0.064 for data size of 100). In terms of computational time, it is obviously seen that the larger training data size and the more complex model (increasing number of hidden layers), the longer the average computational time required to develop an ANN model.

5.3.3 Summary of the Performances of Surrogate Models

We present the summary of the performances of surrogate models discussed in the previous section in Figure 5.3. This figure depicts the error bar representing the 95% confidence interval of obtained RMSE values. The point in the middle of each error bar represents the average RMSE value, while the labels on the top of the error bar show the size of training data for the corresponding surrogate models. This figure also presents the computational time in seconds required for developing each surrogate model (x-axis).

In Figure 5.3 the models developed from training data of size 100 generally have lower average RMSE compared to models based on training data of size 50.

ANN-Rprop models with one hidden layer have higher average RMSE compared to ANN-Backprop, ANN-Rprop two and three hidden layers, and multidimensional Kriging models

In addition to average RMSE values, ANN-Backprop models, for models developed from 100 training data size, demonstrate higher consistency compared to ANN-Rprop with two and three

Table 5.5: Average RMSE and standard deviation of ANN models-Rprop with two hidden layers for data training size of 50 and 100

training data size					
50			100		
h1,h2	mean	stdev	h1,h2	mean	stdev
10,80	0.199	0.015	10,50	**0.117**	0.007
10,90	0.200	0.009	10,60	**0.118**	0.004
20,100	0.200	0.009	10,100	**0.119**	0.011
10,50	0.201	0.013	10,90	**0.119**	0.001
30,70	0.201	0.008	10,30	**0.120**	0.013
20,80	0.203	0.011	10,80	**0.120**	0.006
40,40	0.204	0.005	20,40	**0.120**	0.008
20,30	0.204	0.005	10,20	**0.121**	0.005
10,100	0.204	0.012	20,90	**0.121**	0.008
20,50	0.204	0.017	10,70	**0.123**	0.004
p-value	1	0.1643		0.993	0.2336

hidden layers, and the multidimensional Kriging model.

We also notice from Figure 5.3 that the larger training data size, the longer the average computational time required for developing a surrogate model. The time difference in the development of the ANN with Bprop from data training size of 50 and 100 is less than two seconds on average. Furthermore, for multidimensional Kriging, as the training data size doubles, the computation time increases 6-fold. In general Kriging methods require more computational time for no gain in accuracy compared to the other methods. For instance, the ANN-Backprop model developed from 100 data points only requires about 5 seconds on average, while multidimensional Kriging from the same training data size requires nearly 3.5 hours on average. Even with larger computational time, average RMSE for Kriging is higher than that of Bprop. According to a p-value of t-test, average RMSE of multidimensional Kriging and ANN with Bprop from training data size of 100 is 0.038 for which average RMSE of Bprop is lower than the value provided by multidimensional Kriging.

To explore time efficiency in modelling we also fitted polynomial models of degree 2, 3, 4, and 5, based on training data of size 50 and 100. The results showed that polynomial model performance was worse with average RMSE ranging from 0.248 to 0.310 for 50 training data and from 0.216 to 0.232 for 100 training data compared to multidimensional Kriging and ANN models, although running these models was very time efficient.

Table 5.6: Average RMSE and standard deviation of ANN Models-Rprop with three hidden layers for data training size of 50 and 100

training data size					
50			100		
h1,h2,h3	mean	stdev	h1,h2,h3	mean	stdev
10,50,70	0.197	0.010	10,50,70	**0.116**	0.006
10,60,100	0.203	0.011	10,100,80	**0.117**	0.006
10,100,90	0.203	0.004	10,40,40	**0.120**	0.011
10,80,90	0.204	0.018	10,10,40	**0.121**	0.008
10,60,70	0.204	0.013	10,70,70	**0.121**	0.009
10,40,100	0.205	0.011	10,100,70	**0.121**	0.013
10,40,90	0.206	0.021	10,20,80	**0.121**	0.008
10,30,70	0.207	0.006	10,20,70	**0.122**	0.005
20,90,90	0.207	0.009	10,30,30	**0.122**	0.006
20,80,70	0.207	0.009	10,60,100	**0.122**	0.008
P-value	0.994	0.036		0.996	0.831

5.4 Application of Surrogate Models in History Matching

In this section, we present the use of proxy models in HM. The proxy model replaces the LUCICAT model within the iterations of the search algorithms. For example, during the run of ACO_R, for a specified setting of input parameters, observed streamflow is estimated from the proxy model instead of recalling and running the LUCICAT model. From the range of proxy modelling methods described in section 5.3.3, the best performing model (lowest RMSE) is selected for each category and training data size to perform HM. These modelling settings are presented in Table 5.8 below.

5.4.1 Indirect History Matching Settings

ACO_R, $DACO_R$, ROPE, and GLM are implemented to find the five sensitive input parameters, namely ia, K_{uv}, b, K_{ll}, and c for The Dandalups by matching the estimated response of annual daily peak streamflow from proxy model with observed streamflow. This study was discussed in Chapter 4 section 4.4.1, where direct HM methodology was used to estimate model parameters. Indirect HM procedures settings are described as follows:

- We used the same procedure for generating initial settings of input parameters for all HM methods as discussed in Chapter 4 section 4.4.1. The initial settings for ACO_R and $DACO_R$ were generated by using LHD based on the S-optimality criterion. We

Table 5.7: Average RMSE, standard deviation and computational time of the best performances of ANN Models-Rprop with one, two and three hidden layers for different data training size

Training Data Size	Number of Hidden Layers			P-value
	one	two	three	
50				
mean RMSE	0.207	0.199	0.197	0.311
stdev	0.019	0.015	0.015	0.531
computational time (s)	24.0	50.7	89.0	0.000
100				
mean RMSE	0.144	0.117_b	0.116_b	0.001
stdev	0.000	0.007	0.006	0.064
computational time (s)	59.5	116.8	148.7	0.000

Table 5.8: The surrogate models applied for HM.

	Surrogate Models			
	Kriging	ANN		
		one layer Backprop	two layers Rprop	three layers Rprop
training size of 50	50	(5,60,1)	(5,10,80,1)	(5,10,50,70,1)
training size of 100	100	(5,50,1)	(5,10,50,1)	(5,10,50,70,1)

used the maximin criterion (maximizing the minimum distance between design points) for ROPE. Meanwhile, for GLM, the initial settings were randomly selected from a set of 50 combinations of input parameters generated by LHD based on S-optimality. All criteria are provided by the lhs.design function in the DoW.wrapper R package [178]. Tuning variables of all HM methods were adopted from Chapter 4.3.

- For each model in Table 5.8, each HM method was repeated six times using six different initial input parameter settings.
- The objective function presented in Equation (5.3) is used for HM . The aim of HM is to calibrate five input parameters (ia, K_{uv}, b, K_{ll}, and c) to match 33 annual daily peaks between 1960 and 1992.
- For ACO_R and DACO_R, the stopping criterion was an improvement of less than 0.005 within four consecutive iterations. For ROPE, it was the maximum of five iterations or the

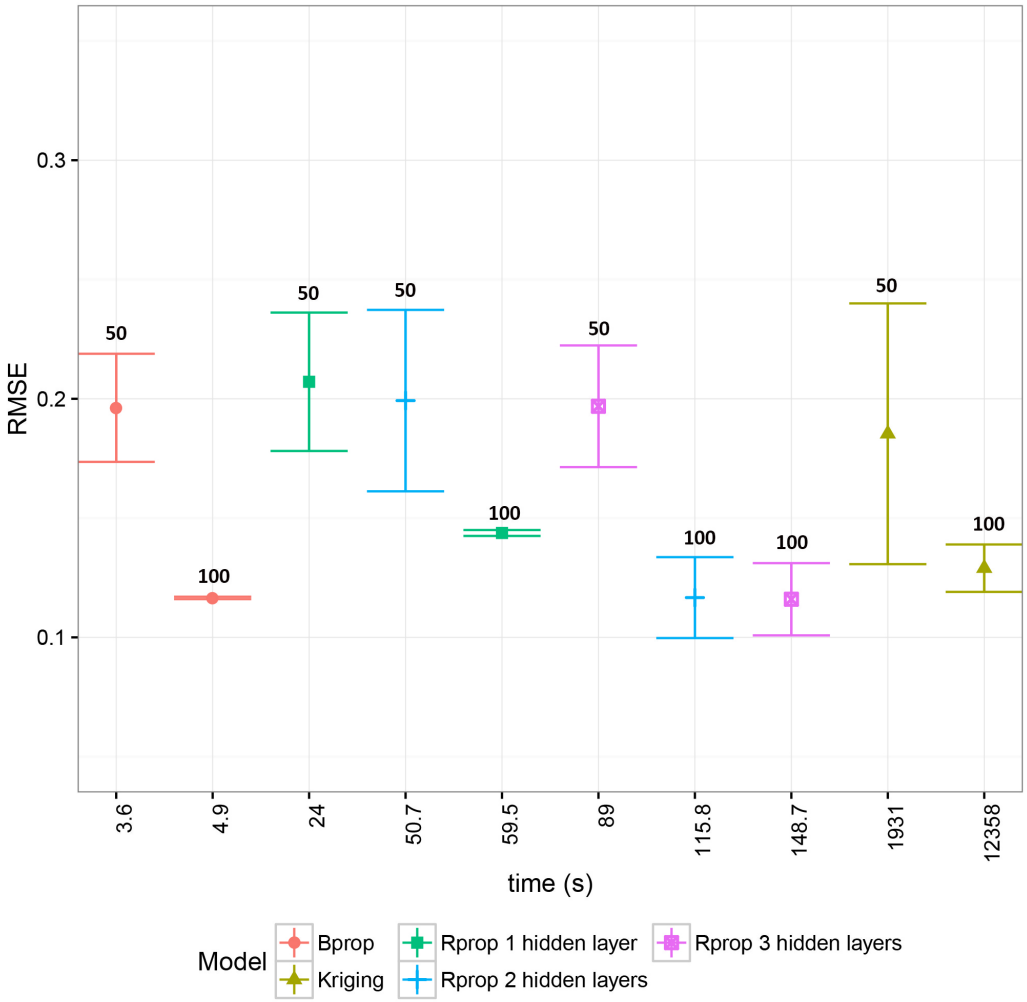

Figure 5.3: 95% Confidence interval of RMSE of surrogate models and computational time

improvement of less than 0.005. Meanwhile, GLM was stopped when the RMSE obtained from four consecutive iterations had improvement of less than 0.005.

- The performances of all algorithms were compared with direct HM (Chapter 4 section 4.4.1) in terms of average RMSE, variance in RMSE, computational time and number of models, and uncertainty of input parameter reduction, using streamflow.

5.4.2 Performance of Indirect History Matching Procedure: Accuracy and Consistency

The results for HM based on the selected proxy models are presented in Table 5.9 to Table 5.12. For each scenario we present the mean, standard deviation, and 95% confidence interval for RMSE resulting from six experiments. These results are then compared to those provided by

HM methods based on the LUCICAT model implemented in Chapter 4, via t-test and F-test for comparison of average and variance respectively.

Table 5.9 to Table 5.12 show that the average RMSE for indirect HM are comparable with the results provided in Chapter 4 section 4.4.1 (indicated with 95% CI and p-values), except for those obtained from multiple solution-based methods coupled with multidimensional Kriging (p-value = 0.001) from a training data size of 100. For multidimensional Kriging (developed from 100 data points) average RMSE is higher. This is an unusual observation and no computational errors were located in this case. This observation may be attributed to the interpolative nature of Kriging methods, and the error surface for Kriging interfering with the current tuning parameters of the HM algorithms to locate the minimum region in input parameter space. This may be explored further if Kriging is the choice of the method for developing a proxy model.

In terms of consistency within the results of indirect HM, all proxy modelling methods show less or similar consistency for ACO_R (Table 5.9) , ROPE (Table 5.11), and GLM (Table 5.12). This is in contrast to the results provided by $DACO_R$ where direct HM showed lower variability than those obtained from surrogate models in Table 5.10 (p-value=0). This was largely due to the almost zero standard deviation for $DACO_R$.

The results indicate that the ANN-based proxy model, coupled with algorithms like ACO_R, $DACO_R$, ROPE and GLM, can safely be used for HM in this instance. The results of chapter 4 and Figure 5.3 can be used as a guide for choice of the best HM method, and/or choice of methodology for developing a proxy model. Computational time in implementing HM should also be considered as a trade-off between accuracy and implementation time.

5.4.3 The Computational Time For Indirect History Matching Procedure

We now compare the computational time required for performing MH via indirect and direct approaches. We consider proxy models developed for training data of size 50 as this size shows comparable performances with those obtained in Chapter 4 section 4.4.1. The summary of average computational time required is listed in Table 5.13.

As shown in Table 5.13, indirect HM achieves a reduction in computational time required for HM. The gain in computational time is more significant for multiple solution-based methods. ACO_R with surrogate models can reduce computational time by at least 85% over the corresponding HM counterpart. For $DACO_R$, indirect HM requires only 20% of the computational time required by $DACO_R$ with the real LUCICAT model. The implementation of surrogate models provides the greatest advantage when the HM methods used are computational expensive, such as the ROPE algorithm. When using the direct HM approach, ROPE took a few days to complete a HM process, while by using the indirect HM approach, the algorithm only required 1.5 hours at most (with multidimensional Kriging), significantly reducing the computation time (at least 98% reduction compared to direct HM). As is seen in Table 5.13, the indirect HM can reduce computational time required by GLM by over 16% for ANN based proxy models,

Table 5.9: Comparison of average RMSE, standard deviation and 95% CI of surrogate models-based ACO_R with LUCICAT model-based ACO_R implemented in Chapter 4, via t-test and F-test for comparison of average and variance respectively.

Training data size of 50				
ACO_R with	Kriging	ANN-Backprop	ANN-Rprop 2 hidden layers	ANN-Rprop 3 hidden layers
mean RMSE	3.663	3.658	3.670	3.658
stdev	$2.282e-3$	$7.584e-4$	$8.242e-3$	$1.870e-3$
95% CI lower	3.660	3.657	3.661	3.656
95% CI upper	3.665	3.658	3.678	3.660
p-value t-test	0.725	0.223	0.426	0.273
p-value F-test	0.001	0.000	0.371	0.001
Training data size of 100				
ACO_R with	Kriging	ANN-Backprop	ANN-Rprop 2 hidden layers	ANN-Rprop 3 hidden layers
mean RMSE	3.702	3.657	3.669	3.657
stdev	$1.481e-2$	$7.282e-4$	$8.380e-3$	$7.425e-4$
95% CI lower	3.687	3.656	3.660	3.656
95% CI upper	3.718	3.658	3.678	3.658
p-value t-test	0.001	0.181	0.475	0.189
p-value F-Test	0.715	0.000	0.391	0.000
ACO_R with	LUCICAT			
mean RMSE	3.665			
stdev	$1.246e-2$			
95% CI lower	3.655			
95% CI upper	3.675			

Table 5.10: Comparison of average RMSE, standard deviation and 95%CI of surrogate models-based DACO$_R$ with LUCICAT model-based DACO$_R$ implemented in Chapter 4, via t-test and F-test for comparison of average and variance respectively

Training data size of 50				
DACO$_R$ with	Kriging 3	ANN-Backprop	ANN-Rprop 2 hidden layers	ANN-Rprop 3 hidden layers
mean RMSE	3.665	3.663	3.667	3.663
stdev	$1.026e-2$	$1.129e-2$	$1.107e-2$	$1.111e-2$
95% CI lower	3.654	3.651	3.655	3.651
95% CI upper	3.676	3.675	3.678	3.674
p-value t-test	0.098	0.212	0.115	0.174
p-value F-Test	0.000	0.000	0.000	0.000
Training data size of 100				
DACO$_R$ with	Kriging	ANN-Backprop	ANN-Rprop 2 hidden layers	ANN-Rprop 3 hidden layers
mean RMSE	3.678	3.663	3.660	3.662
stdev	$1.257e-2$	$8.860e-3$	$3.656e-3$	$1.036e-2$
95% CI lower	3.665	3.653	3.656	3.652
95% CI upper	3.690	3.672	3.664	3.673
P-value t-test	0	0.127	0.093	0.206
P-value F-Test	0.000	0.000	0.000	0.000
DACO$_R$ with	LUCICAT			
mean RMSE	3.656			
stdev	$2.983e-4$			
95% CI lower	3.656			
95% CI upper	3.656			

Table 5.11: Comparison of average RMSE, standard deviation and 95%CI of surrogate models-based ROPE with LUCICAT model-based ROPE implemented in Chapter 4, via t-test and F-test for comparison of average and variance respectively

Training data size of 50				
ROPE with	Kriging 3	ANN-Backprop	ANN-Rprop 2 hidden layers	ANN-Rprop 3 hidden layers
mean RMSE	3.671	3.679	3.672	3.671
stdev	$9.728e-3$	$9.922e-3$	$2.414e-3$	$2.068e-3$
95% CI lower	3.661	3.669	3.669	3.668
95% CI upper	3.681	3.689	3.674	3.673
p-value t-test	0.523	0.563	0.529	0.389
P-value F-Test	0.654	0.654	0.002	0.002
Training data size of 100				
ROPE with	Kriging	ANN-Backprop	ANN-Rprop 2 hidden layers	ANN-Rprop 3 hidden layers
mean RMSE	3.723	3.676	3.674	3.670
stdev	$1.607e-3$	$1.117e-2$	$3.139e-3$	$2.883e-3$
95% CI lower	3.721	3.664	3.670	3.667
95% CI upper	3.724	3.688	3.677	3.673
p-value t-test	0.000	0.927	0.745	0.342
p-value F-Test	0.000	0.8721	0.011	0.005
ROPE with	LUCICAT			
mean RMSE	3.675			
stdev	$1.208e-2$			
95% CI lower	3.665			
95% CI upper	3.685			

Table 5.12: Comparison of average RMSE, standard deviation and 95%CI of surrogate models-based GLM with LUCICAT model-based GLM implemented in Chapter 4, via t-test and F-test for comparison of average and variance respectively.

Training data size of 50				
GLM with	Kriging	ANN-Backprop	ANN-Rprop 2 hidden layers	ANN-Rprop 3 hidden layers
mean RMSE	3.696	3.665	3.712	3.680
stdev	$7.305e-2$	$1.179e-2$	$3.254e-3$	$3.635e-3$
95% CI lower	3.638	3.656	3.709	3.677
95% CI upper	3.755	3.675	3.715	3.683
P-value t-test	0.177	0.109	0.213	0.134
P-value F-Test	0.001	0.917	0.010	0.020
Training data size of 100				
GLM with	Kriging	ANN-Backprop	ANN-Rprop 2 hidden layers	ANN-Rprop 3 hidden layers
mean RMSE	3.697	3.658	3.672	3.659
stdev	$3.172e-3$	$6.304e-6$	$3.270e-4$	$1.443e-3$
95% CI lower	3.694	3.658	3.672	3.658
95% CI upper	3.670	3.658	3.673	3.661
P-value t-test	0.172	0.098	0.120	0.100
P-value F-Test	0.012	0.000	0.000	0.000
GLM with	LUCICAT			
mean RMSE	3.840			
stdev	$2.201e-1$			
95% CI lower	3.664			
95% CI upper	4.016			

Table 5.13: Average computational time (Seconds) in indirect and direct HM.

	Kriging	Backprop 1 hidden	Rprop 2 hidden	Rprop 3 hidden	LUCICAT
	50	50	50	50	
		ACO_R			
data acquisition	2,484	2,484	2,484	2,484	
model development	1,931	4.9	115.8	148.7	
HM process	60	0.5	0.6	1.2	33,001
total	4,475	2,489.4	2,600.4	2,633.9	33,001
% reduction	86.4	92.5	92.1	92.0	
		$DACO_R$			
data acquisition	2,484	2,484	2,484	2,484	
model development	1,931	4.9	115.8	148.7	
HM process	26.2	0.3	0.3	0.6	23,062
total	4,441.2	2,489.2	2,600.1	2,633.3	23,062
% reduction	80.7	89.2	88.7	88.6	
		ROPE			
data acquisition	2,484	2,484	2,484	2,484	
model development	1,931	4.9	115.8	148.7	
HM process	311	94.2	146.6	162.2	295,074
total	4,726	5,1031.1	5,266.4	5,314.9	295,074
% reduction	98.4	99.1	99.1	99.1	
		GLM			
data acquisition	2,484	2,484	2,484	2,484	
model development	1,931	4.9	115.8	148.7	
HM process	345	2.5	4.8	5.9	3,140
total	4,760	2,491.4	2,604.6	2,638.6	3,140
% reduction	-	20.7	17.1	16.0	

compared to the direct HM approach.

In many applications the HM algorithms will be run multiple times to arrive at the best solution. The multiple runs of HM, with the same proxy model, will result in more economy of time. For instance, for a single run of HM with multidimensional Kriging no reduction in computational time is achieved. However, if we repeat HM six times, as done in Chapter 4 section 4.4.1, the multidimensional Kriging proxy model with GLM also shows an increase in time efficiency. To support this finding, we provide the following calculations: the total computational time (s) = 2,484 (data preparation) + 1,931 (model development) + 345 $\times$ 6 = 6,485. The computational time used in direct HM was $3,140 \times 6 = 18,840$s. Therefore, there will still be a time reduction of more than 65%.

To explore the efficiency of Indirect HM, we compared the above results with the expert-guided best estimate. HM for The Dandalups catchment for matching of 33 annual peak streamflows resulted in an RMSE of 1.616, with over 3 moths 1,728,000 (s) of computing time. Whereas the indirect HM approach could be completed in 5,000 seconds, or 1.5 hours at most. There was also an improvement of at least 10% in accuracy.

5.4.4 Comparison of Reduction of Initial Input Parameter Uncertainty Between Direct and Indirect History Matching Procedures.

We present 95% CIs of the reduction of initial input parameter uncertainty in Figure 5.4 to Figure 5.7. These figures correspond to the input parameters obtained from the four HM algorithms via direct and indirect HM procedures. For indirect HM, a proxy model, based on 50 data points using multidimensional Kriging, ANN-Backprop, and ANN-Rprop with two and three hidden layers are considered. For direct HM we consider final input parameters obtained from HM methods and the LUCICAT model (in Chapter 4 section 4.4.1).

For the ACO$_R$ results in Figure 5.4, the obtained input parameters using indirect HM procedure are contained within the 95% CI for direct HM results, with an exception of once case (ANN-Rprop, two hidden layers for parameter k_{ll}). However, the range of the parameter in this case, is still a significant uncertainty reduction in the original range of parameters, and in the same direction as indirect HM. Moreover, indirect HM results generally have much smaller ranges than those found in the direct HM results. This may be due to less variation imputed by the proxy model.

The reduction of input parameter uncertainty for DACO$_R$ provided by indirect HM (Figure 5.5) is within the 95% CIs of direct HM results, with the exception of few cases. These cases being ANN-Rprop with 2 hidden layers (ia and c), ANN-Rprop with 3 hidden layers (K_{uv}), and multidimensional Kriging (K_{ll}). This may be due to the confounding effect of a number of factors, like the nature of these models, choice of error function, tuning parameters of DACO$_R$ or stopping criterion for DACO$_R$. In spite of this, all results demonstrate that the obtained input parameters have similar consistent direction of uncertainty reduction as provided by DACO$_R$

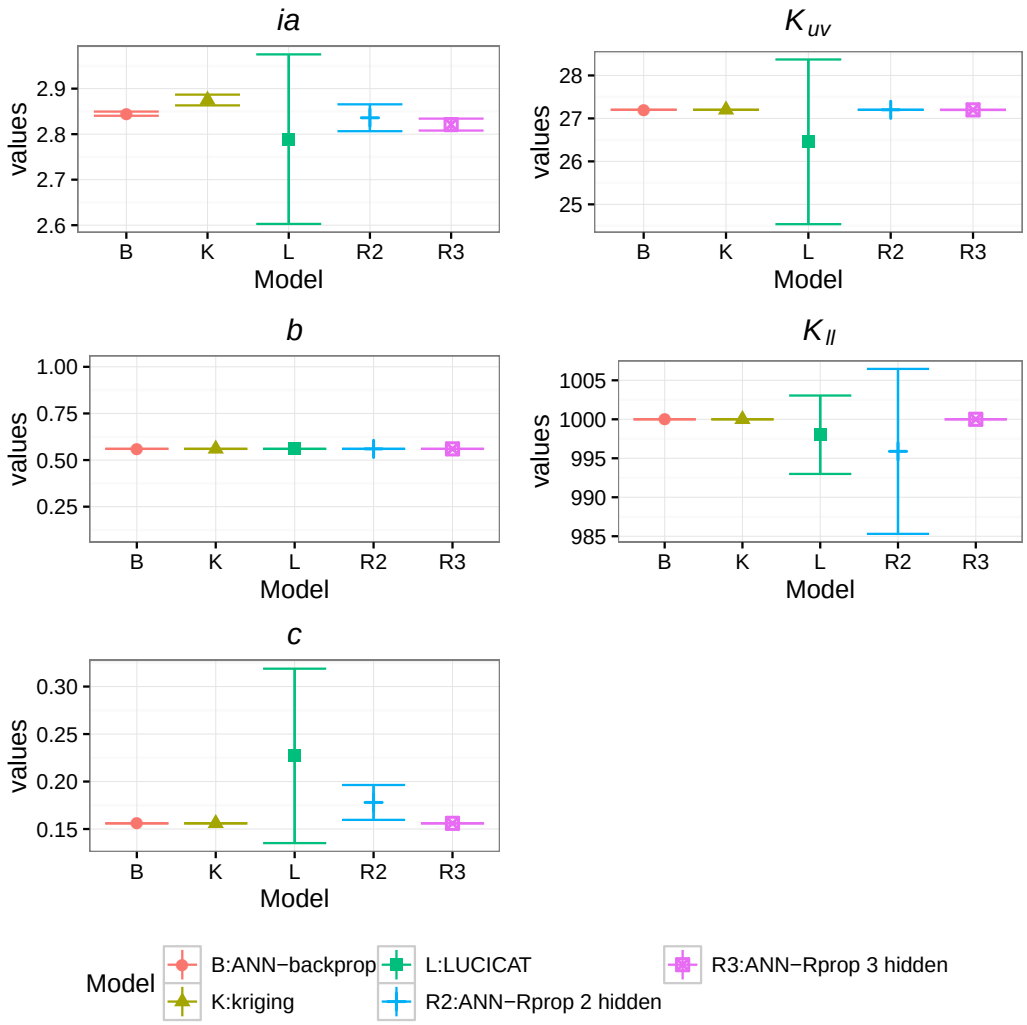

Figure 5.4: The reduction of initial input parameter uncertainty using direct and indirect HM procedure by ACO_R

and LUCICAT. The estimated parameters would still lead to comparable estimates of water resource.

ROPE results for indirect and direct HM are within comparable ranges as depicted in Figure 5.6. We observe that the 95% CIs of ANN-Backprop mostly show higher variation compared to other models across the input parameters. Moreover, most surrogate models used for HM result in locating smaller regions compared to those provided by direct HM.

For GLM (Figure 5.7), except for input parameter K_{ll}, the ranges of obtained input parameters of indirect HM have greater agreement with the ranges of the direct HM procedure. Meanwhile, input parameter K_{ll} for multidimensional Kriging and ANN-Backprop model have larger ranges than those provided by the other models. Despite this variation, K_{ll} ranges provided by multidimensional Kriging and ANN-Backprop models include the range obtained from GLM and the LUCICAT model.

All of the above indirect HM methods have resulted in comparable input parameter ranges

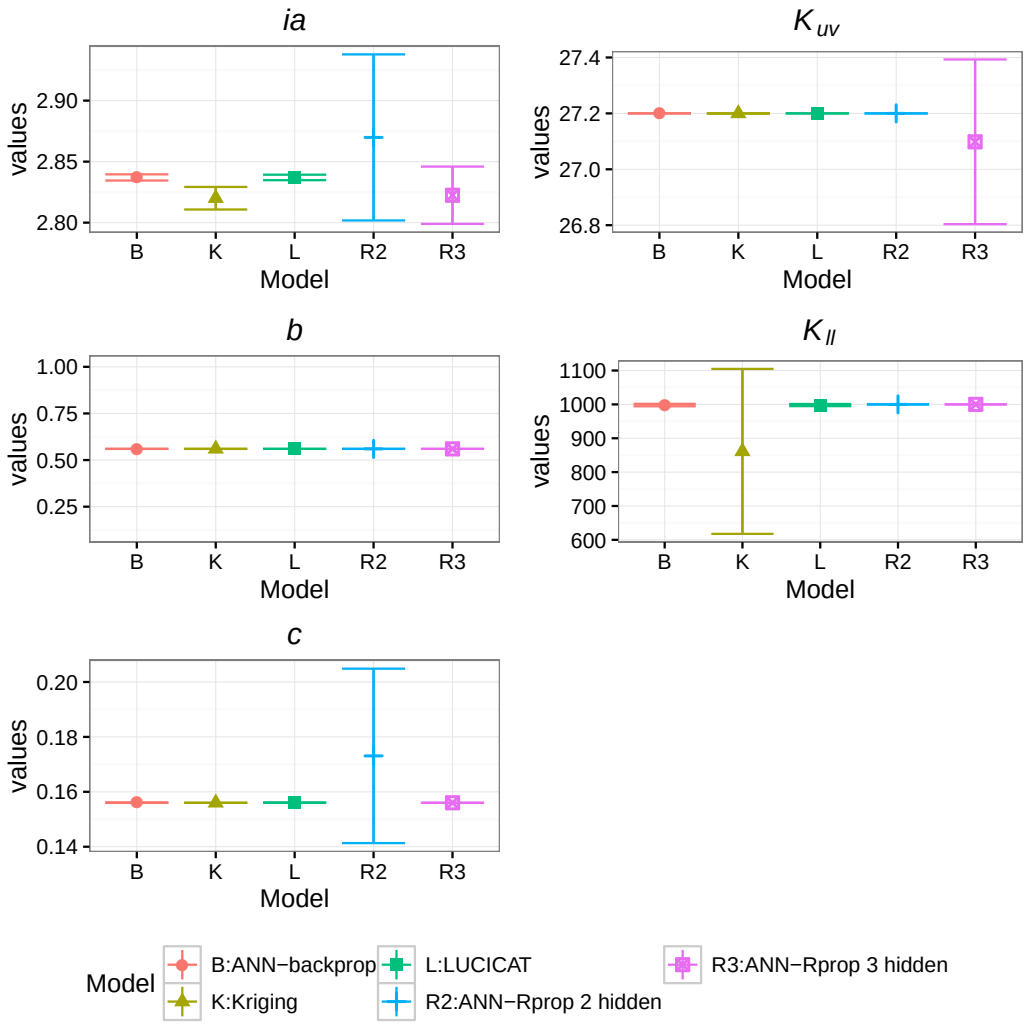

Figure 5.5: The reduction of initial input parameter uncertainty using direct and indirect HM procedure by DACO$_R$

as direct HM methods, that would lead to similar estimates of water resources in this case study.

5.5 Concluding Remarks

In this chapter, we applied the indirect HM approach by developing a proxy model for LUCICAT to be used during the process of HM. Four HM methods, namely ACO$_R$, DACO$_R$, ROPE and GLM, were applied to estimate the five sensitive input parameters of The Dandalups by matching the historical 33 annual daily peaks of streamflow. In this indirect HM approach, the HM methods aimed to reduce the differences between the observed 33 annual daily peaks of streamflow and the model response provided by proxy models. We selected two different classes of techniques for developing a proxy model, namely multidimensional Kriging and ANN, suited for computer simulated experiments. Polynomial models were also experimented with, however,

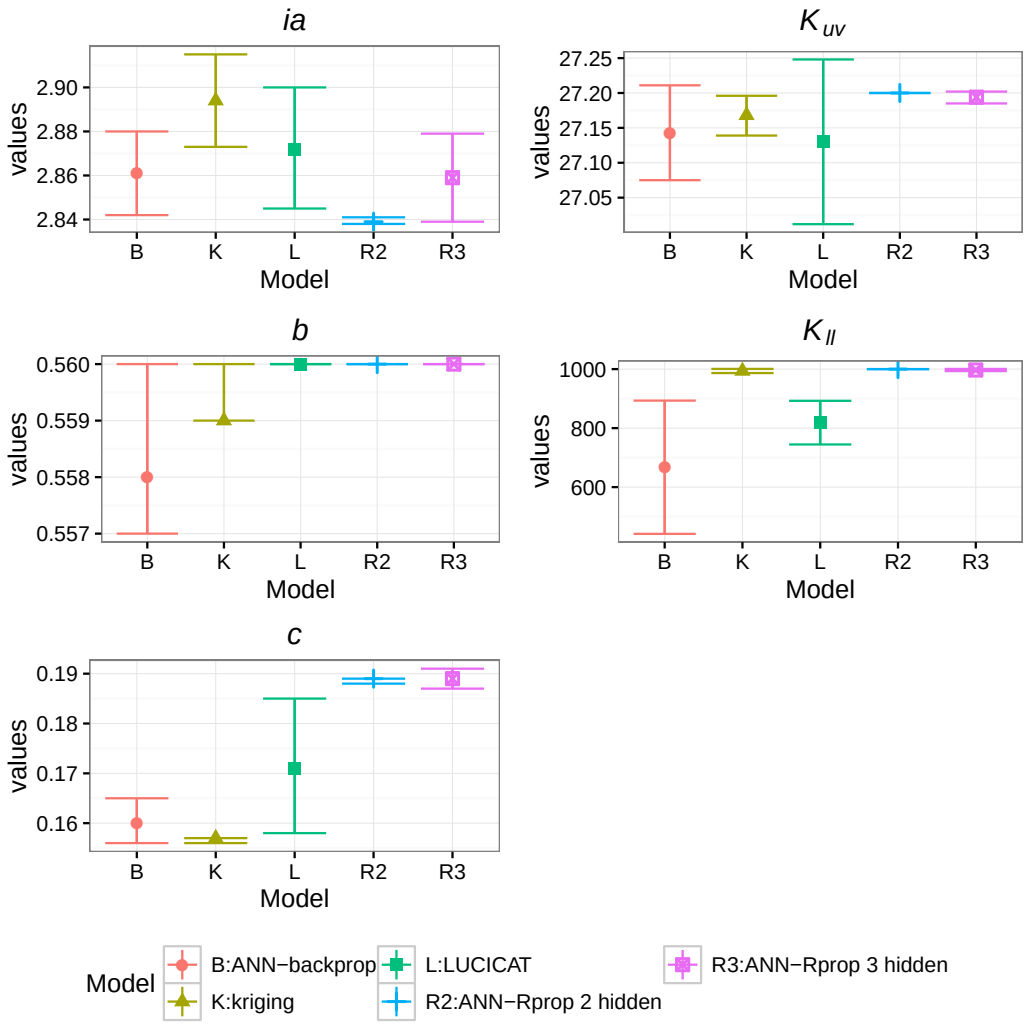

Figure 5.6: The reduction of initial input parameter uncertainty using direct and indirect HM procedure by ROPE

these models show incomparable performance with multidimensional Kriging and ANN and are not presented here.

Before HM, we conducted sensitivity analysis tests to select the best method for developing a proxy model for LUCICAT. Methods of mutlidimensional Kriging and ANN were investigated as these methods were more suited for computer simulation experiments. The model development via ANNs were based on two different learning algorithms, namely Backprop and Rprop. ANN with Backprop was considered to have one hidden layer. Meanwhile, we considered ANN-Rprop with one, two, and three hidden layers. We also included variation of the number nodes in each hidden layer for ANN. For multidimensional Kriging, distinct weighting was used for each parameter. Models were developed for training data of sizes 50 and 100. Different modelling scenarios were compared for accuracy (RMSE) and computational time. The sensitivity analysis results revealed that the larger the training data size used to develop proxy models, the better the average performance of surrogate models in terms of RMSE and consistency (standard

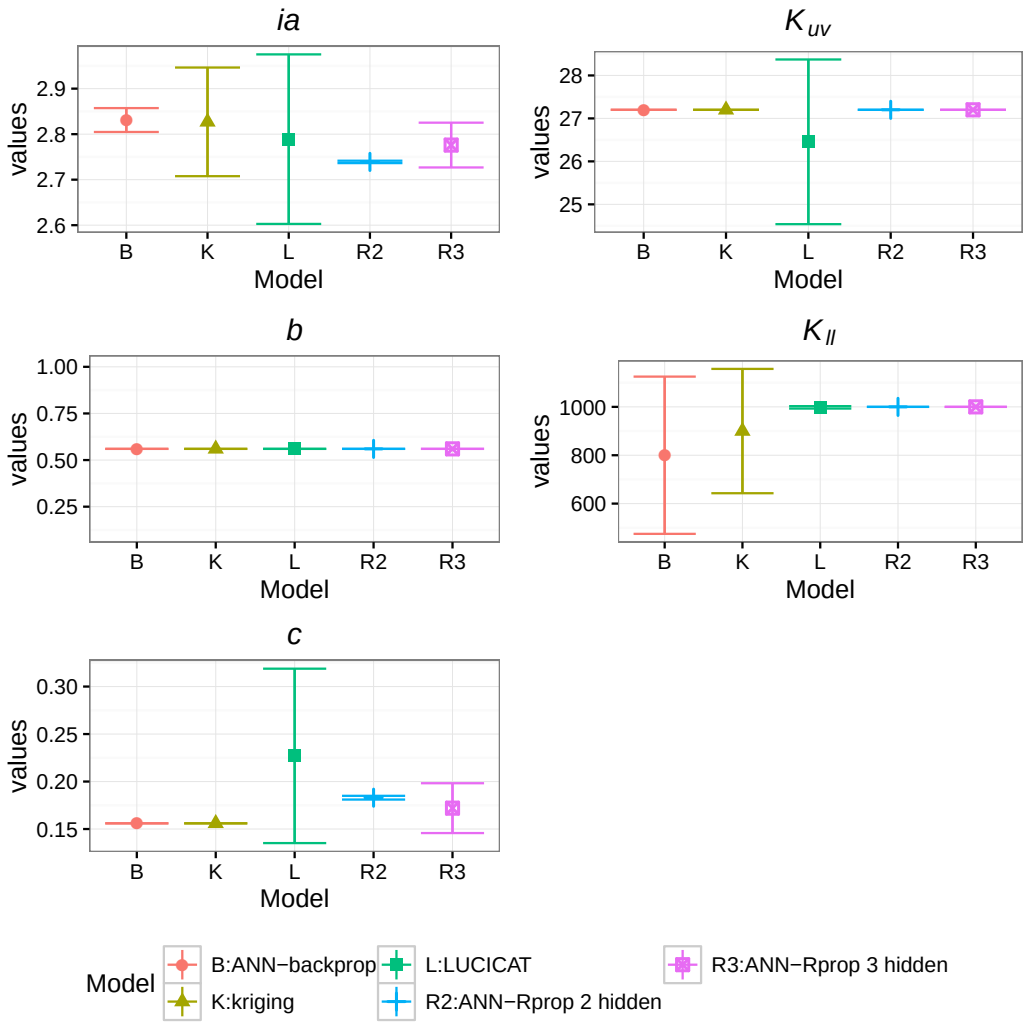

Figure 5.7: The reduction of initial input parameter uncertainty using direct and indirect HM procedure by GLM

deviation). ANN-Rprop with one hidden layer also showed lower performance compared to ANN models with two hidden layers. In terms of computational time, modelling via multidimensional Kriging is time consuming compared to the other methods.

Based on the sensitivity tests, we selected the best performing proxy model within each set of techniques to use for HM. The results showed that the indirect HM approach showed comparable average performances with direct HM, except for multidimensional Kriging developed from 100 data points. The results also generally had higher consistency than the HM methods when a proxy model for LUCICAT was developed by using ACO_R, ROPE, and GLM methods.

Indirect HM provided a significant advantage in reduction of computational time, especially for HM using multiple solution-based methods. Compared to direct HM, ACO_R was able to reduce computational time by more than 85%, while $DACO_R$ and ROPE achieved reduction of over 80% and 98% respectively. GLM coupled with ANN models also showed a reduction of computational time by at least 16%.

In terms of initial input parameter uncertainty, all HM methods in the indirect HM approach showed a consistent direction of reduction of input parameters as those performed by direct HM approach. Furthermore, most of the indirect HM cases had much smaller optimal parameter ranges than those obtained from direct HM. Nevertheless, in comparison to the initial input parameter range, all results of direct and indirect HM span a very narrow range of each parameter uncertainty. In comparison to the results obtained by expert-guided HM performed by industry, the indirect HM approach provided similar accuracy with significant reduction of computational time.

This chapter presents a proof of concept of successful HM via a proxy model for LUCICAT. This approach can be tested on other catchments and scenarios, and possibly be used as a method of choice in the future application of HM.

CHAPTER 6

Summary and Future Research Directions

6.1 Summary and Main Contributions of the Thesis

The Land Use Change Incorporated CATchment (LUCICAT) hydrological model is widely used in Western Australia to predict streamflow and total salt/salinity output of catchments based on climate parameters and geology. This thesis demonstrated application of direct and indirect history matching (HM) approaches for estimation of the input parameters of LUCICAT by matching historical data (streamflow and/or salt). The focus is to make the process of HM efficient, by reducing computational time and improving the model's accuracy.

Four popular HM methods explored here are ant-based optimisation methods for continuous domains, namely ACO_R and diversity ACO_R, also known as $DACO_R$, robust parameter estimation (ROPE) methods, and a gradient-based approach, i.e., Gauss Levenberg Marquardt (GLM). The first three methods are multiple solution-based methods, while the latter technique is a single solution approach. Ant colony optimisation methods are inspired by the natural foraging behaviours of ants. These approaches have been widely applied for solving engineering optimisation problems. However, their application in water resources has been very limited. GLM is a very popular calibration technique which has been applied in hydrological studies. Meanwhile, ROPE is a new method based on the data depth concept. ROPE has been applied to solve rainfall-runoff calibration problems. Both direct and indirect HM approaches were used here. In direct HM the LUCICAT model was used repeatedly in the HM process, while in the indirect approach proxy models were developed to replace LUCICAT in HM. The direct HM approach was discussed in Chapter 4, and the indirect HM approach was described in Chapter 5.

In chapter 4 direct HM methods were implemented to calibrate input parameters (between five and six parameters) for The Dandalups, Bates, and Lewis catchments in Western Australia. Bates and Lewis are small catchments within The Dandalups (700 km^2), each with the area of 2 km^2. The Dandalups are protected water supply catchments as the catchment contains water resources used for maintaining Perth city's water supply. The input parameters of The Dandalups and Bates were calibrated to match streamflow. However, for Lewis the input parameters were calibrated to match observed streamflow and salt generation. The performances of all methods were evaluated according to the accuracy (RMSE), consistency (standard deviation of RMSE),

computational time and reduction of initial input parameter uncertainty. The performance of HM methods were also compared with the best available industry estimates. The findings of this chapter:

- ACO_R and $DACO_R$ performed favourably well across catchments. Both ant-based optimisation methods demonstrated comparable accuracy (RMSE) with ROPE in The Dandalups for HM peak and daily streamflow series and Bates. Moreover, ACO_R and $DACO_R$ had good performance in average weighted RMSE of streamflow and salt generation for Lewis catchment. Ant-inspired optimisation methods also showed high consistency within their results and reduced initial uncertainty of input parameters by more than 50% for all catchments.The parameters provided by ACO_R and $DACO_R$ were robust in the validation period for Bates and Lewis catchment. Ant colony optimisation methods require low to moderate computational time for HM.

- The performance of ROPE was comparable with ACO_R and $DACO_R$ for all catchments. ROPE performed favourably well in Bates, providing model parameter estimates that had the highest accuracy (the lowest average RMSE) and these parameters were also reported robust in the validation period. The performance of ROPE was good when only one response was considered in the objective function. It was also observed that ROPE was computationally expensive. In spite of these disadvantages, ROPE provided a significant reduction of initial uncertainty of input parameters, by more than 50%.

- GLM is the most efficient method in terms of computational time for HM. Its performance was comparable with the other HM methods in The Dandalups for matching daily time series. In other cases with the objective function focused on one response, such as for annual daily peaks of The Dandalups (streamflow), Bates (streamflow) and Lewis (streamflow or salt), GLM showed poor performance. By contrast, GLM had comparable performance with ACO_R and $DACO_R$ when the target of HM was to match streamflow and salt. Compared to other HM methods, GLM was inconsistent in multiple repetitions. For all catchments, GLM provided a reduction of initial uncertainty of input parameters of more than 45%.

- All HM methods were able to improve the accuracy (RMSE) provided by industry estimates by 10-50%.

Detailed performance of each HM method in each scenario/catchment as described in Chapter 4 is summarized in Table 6.1,Table 6.2 and Table 6.3.

In chapter 5, a proxy model was developed to mimic performance of LUCICAT for Dandalup catchment peak annual streamflow. The proxy model was developed using multidimensional Kriging and artificial neural network (ANN). Then HM methods from chapter 4 were implemented, using the proxy model instead of LUCICAT. The two modelling approaches were then

Table 6.1: Comparison of HM method performances for The Dandalups and Bates catchment

	HM method			
	ACO_R	$DACO_R$	ROPE	GLM
The Dandalups: HM based on annual daily peak of streamflow (33 points) Data year: 1960-1992				
accuracy (RMSE)	high (3.665)	high (3.656)	high (3.675)	low (3.840)
consistency	high (1.2e-2)	high (3e-4)	high (1.2e-2)	low (2.2e-1)
computational time (hour)	moderate (9.2)	moderate (6.4)	high (8.2)	low (0.9)
reduction of initial parameter uncertainties	moderate (> 63%)	high (> 99%)	low (> 54%)	low (> 50%)
The Dandalups: HM based on daily time series of streamflow (12,006 points) Data year: 1960-1992				
accuracy (RMSE)	similar (1.372)	similar (1.372)	similar (1.376)	similar (1.410)
consistency	similar (7.9e-4)	similar (4.3e-4)	similar (2.7e-3)	similar (5.2e-2)
computational time (hour)	moderate (7)	moderate (7.6)	high (82.7)	low (1.5)
reduction of initial parameter uncertainties	high (> 95%)	high (> 95%)	high (> 89%)	moderate (> 69%)
Bates: HM based on daily time series of streamflow (4,573 points) Data year: 1988-2000				
accuracy (RMSE)	high (1.158e-2)	high (1.158e-2)	high (1.158e-2)	low (1.181e-2)
consistency	similar (2.6e-5)	similar (4.3e-5)	similar (2.3e-5)	similar (2.2e-5)
computational time	moderate (1.3)	moderate (2.1)	high (17.5)	low (0.4)
reduction of initial parameter uncertainties	moderate (> 75%)	moderate (> 65%)	high (> 85%)	high (> 85%)

Table 6.2: Comparison of HM method performances in Lewis catchment

	HM Method			
	ACO_R	$DACO_R$	ROPE	GLM
Lewis: HM based ω=0 (4,524 points); Data year: 1992-2004				
accuracy (RMSE)	high (1.314)	moderate (1.317)	moderate (1.351)	low (1.372)
consistency	similar (1.1e-4)	similar (5.2e-3)	similar (4.6e-2)	similar (2.9e-2)
computational time (hour)	moderate (1.3)	moderate (2.1)	high (17.5)	low (0.4)
reduction of initial parameter uncertainties	high (> 98%)	high (> 90%)	high (> 86%)	high (> 81%)
Lewis: HM based ω=0.25 (4,524 points); Data year: 1992-2004				
accuracy (RMSE)	high (1.406)	moderate (1.421)	low (1.435)	moderate (1.426)
consistency	high (3.3e-4)	low (8.2e-3)	low (1.7e-2)	low (1.6e-2)
computational time (hour)	moderate (1.6)	moderate (2.1)	high (17.5)	low (0.9)
reduction of initial parameter uncertainties	high (> 96%)	high (> 95%)	high (> 88%)	high (> 84%)
Lewis: HM based on ω=0.5 (4,524 points); Data year: 1992 and 2004				
accuracy (RMSE)	high (1.413)	high (1.414)	low (1.449)	high (1.418)
consistency	high (1.4e-4)	moderate (6.3e-4)	low (2e-2)	low (9.1e-3)
computational time	moderate (1.6)	moderate (1.9)	high (17.3)	low (0.7)
reduction of initial parameter uncertainties	high (> 98%)	high (> 97%)	moderate (> 79%)	high (> 85%)

Table 6.3: Comparison of HM method performances in Lewis catchment

	HM Method			
	ACO_R	$DACO_R$	ROPE	GLM
Lewis: HM based on ω=0.75 (4,524 points); Data year: 1992 and 2004				
accuracy (RMSE)	high (1.239)	high (1.240)	low (1.283)	moderate (1.257)
consistency	high (2.0e-4)	low (7.5e-4)	low (2.8e-2)	moderate (5.3e-3)
computational time	moderate (1.7)	moderate (1.9)	high (17.8)	low (0.9)
reduction of initial parameter uncertainties	high (> 97%)	high (> 96%)	moderate (> 79%)	moderate (> 62%)
Lewis: HM based on ω=1 (4,524 points); Data year: 1992 and 2004				
accuracy (RMSE)	high (0.632)	high (0.632)	high (0.632)	low (0.643)
consistency	high (2.6e-4)	moderate (1.3e-3)	moderate (7.2e-4)	low (1.1e-2)
computational time	moderate (1.3)	moderate (1.2)	high (17.4)	low (0.5)
reduction of initial parameter uncertainties	moderate (> 78%)	moderate (> 75%)	high (> 80%)	low (> 47%)

compared along with the sensitivity of number of model parameters and training data size for developing proxy models. Training data of sizes 50 and 100 were considered. Two different learning algorithms, namely backprop and Rprop were used for ANN. In addition, for ANN-backprop, one hidden layer and for ANN-Rprop one, two and three hidden layers were considered. Besides the number of layers we also tested the number of nodes. The results demonstrated that the larger the training data size used for developing surrogate models, the better average performances of proxy models (RMSE) became, and also higher consistency within results. Among ANN models, ANN-Rprop with one hidden layer did not perform well. In terms of computational time used for the development of proxy models, a larger training data size obviously requires a longer computational time. Compared to other surrogate models, the multidimensional Kriging model is extensive in terms of computational time requiring 30 minutes (model using 50 data points) to a few hours (model using 100 data points) for model development. The results were then compared with those obtained in Chapter 4. The findings of Chapter 5 are summarised as follows:

- Most of the proxy models showed comparable performance with the results provided for direct HM in Chapter 4. Multidimensional Kriging did not perform well.
- Compared to the direct HM approach, indirect HM provided a significant reduction of computational time. Indirect HM was perceived to have more benefit for HM using multiple solution-based methods. In indirect HM, ACO_R was able to reduce the computational time of direct HM by more than 85%; $DACO_R$ needed only 20% of the time required in direct HM; ROPE reduced the time consumed by direct HM by more than 98% using the selected surrogate models. Meanwhile, for a single deterministic approach, GLM, indirect HM also showed a reduction of computational time by at least 16%.
- In terms of initial input parameter uncertainty reduction, all methods in the indirect HM approach showed a consistent reduction of uncertainty of the input parameters, as those found by the direct HM approach. The results for the indirect HM approach had much smaller optimal parameter ranges than those obtained from direct HM.
- The indirect HM approach had similar accuracy to the results provided by industry estimates, but using much less computational time.

To conclude ACO_R and $DACO_R$ can be seen as promising alternative HM methods in hydrology. The implementation of proxy models in HM has provided a significant reduction in computational time with comparable accuracy. Therefore, it will be advantageous to use them for future HM applications.

6.2 Future Research Directions

The work in this thesis has opened several interesting new areas for future research. We discuss these below.

Considering there might be correlated parameters occur in the hydrological model, it is desirable to apply different generation initial sample approach that have capability to include correlation between parameters rather than using LHD approach.

In hydrological models, the high and the low streamflows commonly occur in a period of time of wet and dry seasons. The HM which aims to match the whole series has difficulties capturing both the peak and the low streamflow simultaneously. It is suggested to use different methods that consider both the extreme responses, for example by using log-transformed streamflow, or a discrete wavelet transformation.

RMSE as an error measure is not sufficient to compare the performances of the history matching methods. Since RMSE is only based on magnitude, neglecting the direction of model outputs. Therefore, a single value RMSE can produce numerous different shapes of model outputs. It is desirable to also apply the measure that also considers the direction of model outputs, such as wavelet transform.

It is useful to conduct sensitivity analysis on the length of the period of historical data before performing calibration. It is possible that a certain period within a series of data has a significant impact on the quality of HM results. By only including this period in the HM process, the computational burden for HM will be far less.

Future study would be better to consider various weights on history matching responses since some responses might not have equal importance on the model performance.

Since Bates and Lewis are in The Dandalups, it will be more effective and efficient in comparing HM method performances if the HM is conducted for three areas simultaneously in the future study.

Bibliography

[1] A. A. E. Hassan, H. O. Sharif, T. Jackson and S. Chintalpudi, "Performance of a conceptual and physically based model in simulating the response of a semi-urbanized watershed in san antonio, texas," *Hydrological Processes*, vol. 27, pp. 3394–3408, 2013.

[2] A. B. Dariane and A. M. Moradi, "Reservoir operating by ant colony optimization for continuous domains (acor) case study: Dez reservoir," *International Journal of Mathematical, Pysical and Engineering Sciences*, vol. 3, no. 2, pp. 125–129, 2008.

[3] A. Bandyopadhyay and A. Bhadra, "Development of an ann model for runoff prediction," in *Mathematics Research Developments: Focus on Artificial Neural Network*, J. A. Flores, Ed. New York: Nova Science Publishers, Inc, 2011, pp. 355–374.

[4] A. Bardossy and S. K. Singh, "Robust estimation of hyrological model parameters," *Hydrology and Earth System Sciences*, vol. 12, pp. 1273–1283, 2008.

[5] A. Colorni, M. Dorigo, and V. Maniezzo, "An investigation of some properties of an ant algorithm," in *Proceedings of the parallel problem solving from nature conference (PPSN 92)*, 1992, pp. 509–520, R. Manner and B. Manderic (Eds.).

[6] A. Colorni, M. Dorigo, V. Maniezzo, "The ants system: optimization by a colony of cooperating agents," *IEEE Trans Syst Man Cybern - Par B*, vol. 34, pp. 39–53, 1996.

[7] A. Emerick, E. Silva, B. Messer, F. Luciana, D. Szwarcman, M. Aurelio, and Pacheco, "Well placement optimization using a genetic algorithm with nonlinear constraints," in *Reservoir Simulation Symposium, Woodland, Texas, USA, 2–4 February*, 2009, SPE118808.

[8] A. Jain and A. M. Kumar, "Hybrid neural network models for hydrologic time series forecasting," *Applied Soft Computing*, vol. 7, no. 2, pp. 585–592, 2007.

[9] A. Kitsios, M. A. Bari, and S. P. Charles, "Projected streamflow reduction in the serpentine catchment, western australia, downscaling from multiple general circulation models (gcms)," Department of Water, Perth, Tech. Rep., 2006.

[10] A. Lv, Z. Lu, and P. Wang, "A new learning function for kriging and its applications to solve reliability problems in engineering," *Computers and Mathematics with Applications*, vol. 70, pp. 1182–1197, 2015.

[11] A. Na-Udom, "Experimental design methodology for modelling response from computer simulated experiments," Ph.D. dissertation, Mathematics and Statistics Department, Curtin University, Australia, 2007.

[12] A. Ouenes, "Application of simulated annealing to reservoir characterization and petrophysics inverse problems," Ph.D. dissertation, New Mexico Technology, Socoro, NM, 1992.

[13] A. R Awad and I. Von Poser, "Calibrating conceptual rainfall-runoff models using a real genetic algorithm combined with a local search method," in *Latest Trends on Computer*, vol. 1, 2010, pp. 174–181.

[14] A. Tjia, R. Gupta, and M. Alam, "Ants do history matching," in *Ozwater 2015, 12-14 May, Adelaide, Australia*, 2015.

[15] A. Watson and W. Lee, "A new algorithm for automatic history matching production data," in *Unconventional Gas Technology Symposium, Louisville, USA, 18-21 May*, 1986, SPE15228.

[16] A.G.R. Vaz, B. Elsinga, W.G.J.H.M. van Sark, and M.C. Brito, "An artificial neural network to assess the impact of neighbouring photovoltaic systems in power forecasting in utrect, the netherlands," *Renewable Energy*, vol. 85, pp. 631–641, 2016.

[17] A.L. Cuevas, B.C. Toledo, L.M. Ceja, and C.V. Mejia, "State and parameter estimation of a neural mass model from electrophysiological signals during the status epilepticus," *NeuroImage*, vol. 113, pp. 374–386, 2015.

[18] Australia Bureau of Statistics, "Salinity and land management on western australian farms," http://www.abs.gov.au/ausstats/abs@.nsf/0/f59529c371a21f55ca256db800783a4f/-$FILE/ATT34B9V, 2003, retrieve on 25-10-2015.

[19] ——, "Future article: water in australia," http://www.abs.gov.au/AUSSTATS/abs.nsf/-9e1dd9680bdd9821ca257090002029cc/330bc8fdfd50bee4ca2573c6001049f9OpenDocument, 2007, retrieve on 25-10-2015.

[20] ——, "Population projections, australia, 2012 (base) to 2101," http://www.abs.gov.au/ausstats/abs.nsf/lookup/3222.0Media Release12012 (base) to 2101, 2013, retrieve on 2-11-2015.

[21] ——, "4610.0 - water account, australia, 2012-13," http://www.abs.gov.au/AUSSTATS/abs@.nsf/DetailsPage/4610.02012-13?OpenDocument, 2014, retrieve on 17-02-2015.

[22] ——, "Population clock," http://www.abs.gov.au/ausstats/abs.nsf/0/1647509ef7e25faaca-2568a900154b63
?OpenDocument, 2015, retrieve on 2-11-2015.

[23] B. E. Skahill and J. Doherty, "Efficient accommodation of local minima in watershed model calibration," *Journal of Hydrology*, vol. 329, pp. 122–139, 2006.

[24] B.H. Lunelli, D.I.P. Atala, M.R. Wolf Maciel, and R, Maciel Filho, "Estimation of kinetic parameters of a lactic acid production process from sugar cane molasses using genetic algorithm," *Journal of Biotechnology*, vol. 150, p. 569, 2010.

[25] Bureau of Meteorology, "State of climate 2014," http://www.bom.gov.au/state-of-the-climate, 2014, retrieve on 25-10-2015.

[26] Business Insider Australia, "Here's how many days a person can survive without water," http://www.businessinsider.com.au/how-many-days-can-you-survive-without-water-2014-5, 2014, Retrieve on 2-15-2015.

[27] C. Chartres and J. Williams, "Can australia overcome its water scarcity problem?" *Journal of Developments in Sustainable Agriculture*, vol. 1, pp. 17–24, 2006.

[28] C. F. Wen, "Breakdown properties of location estimates based on halfspace depth and projected outlyingness," *The Annals of Statistics*, vol. 20, pp. 1803–1827, 1992.

[29] C. Fralye, A. Raftery, and L. Scrucca, "Package mclust version 4.2," http://cran.r-project.org/web/packages/mclust/index.html, 2014.

[30] C. Lung, "Ant colony optimization in green manufacturing," in *Ant Colony Optimization-Methods and Applications*, A. Ostfeld, Ed. InTech, 2011, pp. 113–127, ISBN 978-953-307-157-2.

[31] C. Mclennan, "Subdued by drought, salinity is back threatening irrigation areas," http://www.weeklytimesnow.com.au/agribusiness, 2014, The Weekly Times, retrieve on 25-10-2015.

[32] C. Zheng and P. Wang, "Parameter structure identification using tabu search and simulated annealing," *Advances in Water Resources*, vol. 19, no. 4, pp. 215–224, 1996.

[33] C.E. Imrie, S. Durucan , and A. Korre , "River flow prediction using artificial neural networks: generalisation beyond the calibration range," *Journal of Hydrology*, vol. 233, no. 1–4, pp. 138–153, 2000.

[34] W. Corporation, "Wungong catchment environment and water management project," Tech. Rep., 2005. [Online]. Available: http://www.watercorporation.com.au/wungong/

[35] ——, "Drinking water quality: Annual report 2012/2013," Tech. Rep., 2012, Accessed: 14/09/2015. [Online]. Available: http://www.watercorporation.com.au/ /media/files/about-us/our-performance/drinking-water-quality/annual-report-2013.pdf

[36] C.W. Dawson, R.J. Abrahart, A.Y. Shamseldin and R.L. Wilbyd, "Flood estimation at ungauged sites using artificial neural networks," *Journal of Hydrology*, vol. 319, no. 1–4, pp. 391–409, 2006.

[37] C.Y. Kong, P. M. McMahon, and G. S. Gazelle, "Calibration of disease simulation model using an engineering approach," *International for Pharmacoeconomics and Outcomes Research (ISPOR)*, vol. 12, no. 4, pp. 521–529, 2009.

[38] D. Blueschke, V. Blueschke-Nikolaeva, and I. Savin, "New insights into optimal control of nonlinear dynamic econometric models: Application of a heuristic approach," *Journal of Economic Dynamic and Control*, vol. 37, pp. 821–837, 2013.

[39] D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[40] D. N. Khoi and V. T. Thom, "Parameter uncertainty analysis for simulating streamflow in a river catchment of vietnam," *Global Ecology and Conservation*, vol. 4, pp. 538–548, 2015.

[41] D. R. Dawdy and T. O'Donnel, "Mathematical models of catchment behaviour," *Journal of Hydraulic Division ASCE*, vol. 91, no. HY4, pp. 123–137, 1965.

[42] D. Tjia, R. Gupta, M.S. Alam and T. Stokes, "Application of automatic history matching methods in hydrology." Presented in Bauxite Hydrology Committee, Perth, 2015.

[43] D. Tumac, "Artificial neural network application to predict the sawability performance of large diameter circular saws," *Measurement*, vol. 80, pp. 12–20, 2016.

[44] D. Yang, T. Oki, S. Herath and K. Musiake, "Mathematical models of small watershed hydrology and applications," in *Mathematical Models of Small Watershed Hydrology and Application*, V. Singh and D. Frevert, Eds. Colorado: Water Resources Publication, LCC, 2002, pp. 259–300.

[45] Department of the Environment, "Climate change impacts in australia," https://www.environment.gov.au/climate-change/climate-science/impacts, 2016, retrieve on 17-02-2016.

[46] Department of Water, "Understanding salinity," http://www.water.wa.gov.au/water-topics/water-quality/managing-water-quality/understanding-salinity, 2003, retrieve on 25-10-2015.

[47] ——, *LUCICAT user manual*, 2012.

[48] D.P. Boyle, H. V. Gupta, and S. Sorooshian, "Towards improved calibration of hydrologic models: combining the strengths of manual and automatic methods," *Water Resources Research*, vol. 36, no. 12, pp. 3663–3674, 2000.

[49] D.S. Oliver and Y. Chen, "Recent progress on reservoir history matching: a review," *Computational Geosciences*, vol. 15, no. 1, pp. 185–221, 2011.

[50] D.W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *Journal of the Society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

[51] E. Bonabeau, G. Theraulaz, J.L. Deneubourg, S. Aron, and S. Camazine, "Self-organization in social insects," *TREE*, vol. 12, no. 5, pp. 188–193, 1997.

[52] E. Roux and P. O. Bouchard, "Kriging metamodel global optimization of clinching joining processes accounting for ductile damage," *Journal of Materials Processing Technology*, vol. 213, pp. 1038–1047, 2013.

[53] E. Todini, "Hydrological catchment modelling: past, present and future," *Hydrology and Earth System Sciences*, vol. 11, no. 1, pp. 468–482, 2007.

[54] F. Glover, "Future paths for integer programming and links to artificial intelligence," *Computers and Operation Research*, vol. 13, pp. 533–549, 1986.

[55] F. Gunther and S. Fritsch, "Neuralnet: Training of neural networks," *The R Journal*, vol. 2, no. 1, pp. 30–38, 2010.

[56] F. J. Hingston and V. Gailitis, "The geographic variation of salt precipitated over western australia," *Australian Journal of Soil Research*, vol. 14, pp. 319–335, 1976.

[57] F. Simsek and F. Mergen, "Optimization of induction motors using continuous ant colony system," *International Journal of Recent Trends in Engineering and Technology*, vol. 4, no. 3, pp. 62–65, 2010.

[58] F.M. Bass, "A new product growth model for consumer durables," *Management Science*, vol. 15, pp. 215–227, 1969.

[59] G. Bilchev and C. Parmee, "The ant colony metaphor for searching continuous design spaces," in *In Proceedings of AISB Workshop on Evolutionary Computing Lecture Notes in Computer Science*, vol. 993 of LNCS. Berlin: Springer-Verlag, 1995.

[60] G. Chen and O. H. Campanella, "An optimization algorithm for estimation of microbial survival parameters during thermal processing," *International Journal of Food Microbiology*, vol. 154, pp. 52–58, 2012.

[61] G. Chicco, "Ant colony system-based applications to electrical distribution system optimization," in *Ant Colony Optimization-Methods and Applications*, A. Ostfeld, Ed. InTech, 2011, pp. 237–262, ISBN 978-953-307-157-2.

[62] G. Dellino, P. Lino, C. Meloni, and a. Rizzo, "Kriging metamodel management in the design optimization of a cng injection system," *Mathematics and Computers in Simulation*, vol. 79, pp. 2345–2360, 2009.

[63] G. Leguizamon, and C. A. Coello Coello, *An alternative ACO_R algorithm for continuous optimization problems*, Ants's 10 Proceedings of the 7th International Conference on Swarm Intelligence, 2010.

[64] G. Slater and E. Durrer, "Adjustment of reservoir simulation models to match field performance," in *45th Annual Fall Meeting, Houston, Texas, USA 4–7 October*, 1970, SPE2983.

[65] G.C. Chow, *Analysis and Control of Dynamic Economic System.* New York: John Wiley & Sons, 1975.

[66] G.R. Gavalas, P.C. Shah and J.H. Seinfeld, "Reservoir history matching by bayesian estimation," *SPE Journal*, vol. 16, no. 6, pp. 337–350, 1976.

[67] H. Gavin, "The levenberg–marquardt method for nonlinear least squares curve–fitting problems," Department of Civil and Engineering, Duke University, Tech. Rep., 2011.

[68] H. Kunstmann, A. Heckl, and A.Rimmer, "Physically based distributed hydrological modeling of the upper jordan catchment and investigation of effective model equations," *Advances Geosciences*, vol. 9, pp. 123–130, 2006.

[69] H. Madsen, "Automatic calibration of a conceptual rainfall-runoff model using multiple objectives," *Journal of Hydrology*, vol. 235, pp. 276–288, 2000.

[70] H. Madsen, G. Wilson, and H.C. Ammentorp, "Comparison of different automatic strategies for calibration of rainfall-runoff models," *Journal of Hydrology*, vol. 261, pp. 48–59, 2002.

[71] H. Oja, "Descriptive statistics for multivariate distributions," *Statistics and Probability Letters*, vol. 1, pp. 327–332, 1983.

[72] H. R. Maier, A. R. Simpson, A. C. Zecchin, W. K. Foong, K. Y. Phang, H. Y. Seah, and C. Tan, "Ant colony optimization for design of water distribution systems," *Journal of Water Resources Planning and Management Division, ASCE*, vol. 129, no. 3, pp. 200–209, 2003.

[73] H. Wei and S. Deyun, *Extracting solar cell model parameters based on chaos Particle Swarm Algorithm*, International Conference on Electric Information and Control Engineering (ICEICE), 2011:398402, 2011.

[74] H.B. Nielsen, "Damping parameter in marquardt's method," Department of Mathematical of Modelling, Tech. Rep., 1999, Accessed: 11/05/2015.

[75] I. Arasaratnam and S. Haykin, "Cubature kalman filters," *IEEE Transaction on Automatic Control*, vol. 54, pp. 1254–1269, 2009.

[76] I. Couckyuyt, A. Forrester, D. Gorissen, F. De Truck, and T. Dhaene, "Blind kriging: Implementation and performance analysis," *Advances in Engineering Software*, vol. 49, pp. 1–13, 2012.

[77] I. Ruts and P. J. Rousseeuw, "Computing depth contours of bivariate point clouds," *Computational Statistics and Data Analysis*, vol. 23, pp. 153–168, 1996.

[78] J. Cullmann, T. Krausse, P. Saile, "Parameterising hydrological models : Comparing optimisation and robust parameter estimation," *Journal of Hydrology*, vol. 404, pp. 323–331, 2011.

[79] J. Doherty, *PEST: model independent parameter estimation, User Manual*, 5th ed. New Jersey: Prentice-Hall Inc, 2005.

[80] J. Dreo and P. Siarry, "Continuous interacting ant colony algorithm based on dense hierarchy," *Future Generation Computer Systems*, vol. 20, no. 5, pp. 841–856, 2004.

[81] J. Fernandez, D. Echeverria, and T. Mukerji, "Application of particle swarm optimization to reservoir modeling and inversion," in *IAMG09 Conference, Stanford University, 23-28 August*, 2009.

[82] J. K. Ruprecht and Stoneman, G. L., "Water yield issues in the jarrah forest of south-western australia," *Journal of Hydrology*, vol. 150, pp. 369–391, 1993.

[83] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, 1995. Proceedings., IEEE International Conference on (Volume:4)*, November 1995, pp. 1942–1948.

[84] J. L. Deneubourg, S. Aron, S. Goss, and J. M. Pasteels, "The self-organizing exploratory pattern of the argentine ant," *Journal of Insect Behavior*, vol. 3, pp. 159–168, 1990.

[85] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, "Design and analysis of computer experiments," *Statistical Science*, vol. 4, no. 4, pp. 409–435, 1989.

[86] J. Simunek, M. T. van Genuchten, M. M. Gribb, J. W. Hopmans, "Parameter estimation of unsaturated soil hydraulic properties from transient flow processes," *Soil Tillage Research*, vol. 47, pp. 27–36, 1998.

[87] J. Wu and X. Zhu, "Using the shuffled complex evolution global optimization method to solbe groundwater management models," in *Frontiers of WWW research and development - APWeb 2006*, H. T. S. M. K. X. Zhou, J. Li and Y. Zhang, Eds. Berlin, Germany: Springer-Verlag, 2006, vol. 3841 of LNCS, pp. 986–995.

[88] J.C. Meza, "Steepest descent," in *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, no. 6. John Wiley and Sons, Inc, 2010, pp. 719–722.

[89] J.W. Tukey, "Mathematics and the picturing of data," in *Proceedings of the International Congress of Mathematicians, Vancouver, B. C., 1974*, vol. 2. Montreal, Quebec: Canada Mathematics Congress, 1975, pp. 523–531.

[90] ——, *Exploratory Data Analysis*. Addison-Wesley, Reading, MA, 1977.

[91] K. Ananda Babu and, R. K. Shrivastav, "Developing objective functions by ant colony optimization for continuous domains (acor) : Sondur reservoir," *International Journal of Latest Trends in Engineering and Technology (IJLTET)*, vol. 2, pp. 58–64, 2013.

[92] K. C. Abbaspour, R. Schulin, and M. van Genuchten, "Estimating unsaturated soil hydraulic parameters using ant colony optimization," *Advances in Water Resources*, vol. 24, pp. 827–841, 2001.

[93] K. Coats, J. Dempsey, and J. Henderson, "A new technique for determining reservoir description from field performance data," in *43th Annual Fall Meeting, Houston, Texas, 2 September–2 October*, 1968, SPE2344.

[94] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE. Trans. Evolution Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[95] K. Hornik, M. Stichcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359–366, 1989.

[96] K. J. Beven and A. M. Binley, "The future of distributed models: model calibration and uncertainty prediction," *Hydrological Processes*, vol. 6, pp. 279–298, 1992.

[97] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *The Quaterly of Applied Mathematics*, vol. 2, pp. 164–168, 1944.

[98] K. Socha, "Ant colony optimization for continuous and mixed-variable domains," Ph.D. dissertation, IRIDIA-CoDE, University of Brussels, 2008.

[99] K. Socha and M. Dorigo, "Ant colony optimization for continuous domains," *European Journal of Operational Research*, vol. 185, no. 3, pp. 1155–1173, 2008.

[100] ——, "Optimization and parameters estimation in petroleum engineering problems using ant colony algorithm," *Journal of Petroleum Science and Engineering*, vol. 74, pp. 147–153, 2010.

[101] K. Thomas, L. Hellums, and Reheis, "A nonlinear automatic history matching technique for reservoir simulation models," in *46th Annual Fall Meeting, New Orleans, USA, 3–6 October*, 1971, SPE3475.

[102] L. Falat and L. Pancikova, "Quantitative modeling in economics with advanced artificial neural networks," *Procedia Economics and Finance*, vol. 34, pp. 194–201, 2015.

[103] L. Liu, H. Ding, and H.B. Huang, "Improved simultaneous estimation estimation of tracer kinetic models with artifical immune network based optimization method," *Applied Radiation and Isotopes*, vol. 107, pp. 71–76, 2016.

[104] L. Mohamed, M. Christie, and V. Demyanov, "Comparison of stochastic sampling algorithms for uncertainty quantification," in *Reservoir Simulation Symposium, The Woodlands, Texas, USA, 2–4 February*, 2009, SPE119139.

[105] L. Peters, R. Arts, G. Brouwer, C. Geel, S. Cullick, R.J. Lorentzen, and P. Sarma, "Results of the brugge benchmark study for flooding optimization and history matching," *SPE Reservoir Evaluation Engineering*, vol. 13, no. 03, pp. 391–405, 2010.

[106] L. R. Joyce, "The hydrological impacts of climate change and variability in the murray hotham catchment, western australia," Ph.D. dissertation, School of Environmental Systems Engineering, The University of Western Australia, 2007.

[107] M. A. Bari and K. R. J. Smettem, "Modelling monthly runoff generation processes following land use changes: groundwater-surface runoff interactions," *Hydrology and Earth System Sciences*, vol. 8, no. 5, pp. 903–922, 2004.

[108] ——, "A daily salt balance model for stream salinity generation processes following partial clearing from forest to pasture," *Hydrology and Earth System Sciences*, vol. 10, pp. 519–534, 2006.

[109] M. A. Bari and K. Senathirajah, "Modelling yields for different rainfal scenarios at wungong water supply catchment, western australia," in *29th Hydrology and Water Resoures Symposium, 21–23 February 2005, Canberra*, 2005.

[110] M. A. Bari and M. L. Berti, "Predicting stream salinity management options in the kent river catchment using the lucicat model," in *29th Hydrology and Water Resoures Symposium, 21–23 February 2005, The Institute of Engineers, Canberra, Australia*, 2005.

[111] M. A. Bari and S. Roger, "Generating a 100-year daily runoff series for the ord river catchment using the lucicat model, 2006," in *30th Hydrology and Water Resoures Symposium, Hobart, Australia*, 2006.

[112] M. A Bari, D. M. Shakya, and M. Owen, "Lucicat live: A modeling framework for predicting catchment management options," in *18th World IMACS / MODSIM Congress, Cairns, Australia 13–17 July 2009*, 2009. [Online]. Available: http://mssanz.org.au/modsim09

[113] M. Dorigo, "Optimization, learning and natural algorithms," Ph.D. dissertation, Dipartimento di Elettronica, Politecnico di Milano, Italy, 1992.

[114] M. Dorigo and L. Gambardella, "Ant colony system: A cooperative learning approach to the travelling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 53–66, 1997.

[115] M. Dorigo, V. Maniezzo, and A. Colorni, "The ant system: optimization by a colony of cooperating agents," *IEEE Transaction on Systems, Man and Cybertics-Part B*, vol. 26, no. 2, pp. 29–41, 1996.

[116] M. Franchini and G. Galeati, "Comparing several genetic algorithm schemes for the calibration of conceptual rainfall–runoff models," *Hydrology Sciences Journal*, vol. 43, no. 3, pp. 357–379, 1997.

[117] M. I. A. Lourakis, "A brief description of the levenberg-marquardt algorithm implemented by levmar," Institute of Computer Science, Foundation for Reserach and Technology, Hellas, Tech. Rep., 2005.

[118] M. K. Gill, Y. H. Kaheil, A. Khalil, M. McKee, and L. Bastidas, "Multiobjective particle swarm optimization for parameter estimation in hydrology," *Water Resources Research*, vol. 42, no. W07417, pp. 1–14, 2006.

[119] M. K. Transtrum and J. P. Sethna, "Improvements to the levenberg-marquardt algorithms for nonlinear least-squares minimization," *Preprint submitted to Journal of Computational Physics*, 2012.

[120] M. L. Berti, M. A. Bari, S. P. Charles, and E. J. Hauck, "Climate change, catchment runoff and risk to water supply in the south-west of western australia," Department of Environment, Tech. Rep., 2004.

[121] M. Miller, "Rprop-description and implementation details," Institut fur Logic, Komplexitat und Deduktionssyteme, University of Karlsruhe, Tech. Rep., 1994.

[122] M. Owens, *The Dandalups Catchment*, Department of Water, 2014, 1:250,000, released on 27-02-2014.

[123] M. Riedmiller, "Advanced supervised learning in multi-layer perceptrons-from backpropagation to adaptive learning algorithms," *International Journal of Computer Standards and Interfaces*, vol. 16, pp. 265–278, 1994.

[124] M. Shafii and F. De Smedt, "Multi-objective calibration of a distributed hydrological model (wetspa) using a genetic algorithm," *Hydrology and Earth System Sciences*, vol. 13, p. 2137?2149, 2009.

[125] M. Ushada, T. Okayama, A. Suyantohadi, N. Khuriyati, and H. Murase, "Daily worker evaluation model for sme-scale food production system using kansei engineering and artificial neural network," *Agriculture and Agricultural Science Procedia*, vol. 13, pp. 84–88, 2015.

[126] M. Zemzami and L. Benaabidate, "Improvement of artificial neural networks to predict daily streamflow in a semi arid area," *Hydrological Sciences Journal*, 2015.

[127] M.P. Rajurkar, U.C. Kothyari, and U.C. Chaube, "Modeling of the daily rainfall-runoff relationship with artificial neural network," *Journal of Hydrology*, vol. 285, pp. 96–113, 2004.

[128] M.S. Alam, *Lewis Bates Catchments*, Department of Water, 2015, 1:40,000, released on 19-09-2015.

[129] N. J. Schofield, G. L. Stoneham, and J. C. Loh, *Hydrology of the jarrah forest*, b. del, j. j. havel, and n malajczuk ed. Dordrecht, The Netherlands: Kluwer Academic Publishers, 1989.

[130] N. Katsoulas, K. Peponakis, K. P. Ferentinos, and C. Kittas, "Calibration of a growth model for tomato seedlings (tomseed) based on heuristic optimisation," *Biosystems Engineering*, vol. 140, pp. 34–47, 2015.

[131] N. Monmarche, G. Venturini, and M. Slimane, "On how pachycondyla apicalis ants suggest a new search algorithm," *Future Generation Computer Systems*, vol. 16, pp. 937–946, 2000.

[132] N. R. Sumner, P. M. Fleming, and B. C. Bates, "Calibration of a modified sfb model for twenty-five australian catchments using simulated annealing," *Journal of Hydrology*, vol. 197, pp. 166–188, 1997.

[133] National and Water Resources Audit, "Australian dryland salinity assesment 2000," National Land and Water Resources Audit: Canberra, ACT, Tech. Rep., 2001.

[134] O. Baskan and S. Haldenbilen, "Ant colony optimization approach for optimizing traffic signal timings," in *Ant Colony Optimization-Methods and Applications*, A. Ostfeld, Ed. InTech, 2011, pp. 205–220, ISBN 978-953-307-157-2.

[135] O. Korb, T. Stützle, and T. E. Exner, "Plants: application of ant colony optimization to structure-based drug design," in *Proceedings of ANTS 2006, Brussels, Belgium 4-7 September*, vol. 4150 of LNCS. Berlin, Germany: Springer-Verlag, 2006, pp. 247–258.

[136] O. Kovarik, "Ant colony optimization for continuous problems," Ph.D. dissertation, Faculty of Electrical Engineering, Czech Technical University in Prague, 2006.

[137] P. C. Mahalanobis, "On the generalized distance in statistics," *Proceedings of the National Academy of Science of India*, vol. 12, pp. 49–55, 1936.

[138] P. K. Kitanidis and R. W. Lane, "Maximum likelihood parameter estimation of hydrologic spatial processes by the gauss-newton method," *Journal of Hydrology*, vol. 79, pp. 53–71, 1984.

[139] P. Nogueira and D. Schiozer, "An efficient methodology of production strategy optimization based on genetic algorithms," in *Latin American and Caribbean Petroleum Engineering Conference, Cartagena, Colombia, 31 May–3 June*, 2009, SPE122031.

[140] P. P. Grasse, "La reconstruction du nid et les coordinations interindividuelles chez bellicositermes natalensis et cubitermes sp. la theorie de la stigmergie: essai d'interpretation du comportement des termites constructeurs," *Insectes Sociaux*, vol. 6, pp. 41–81, 1959.

[141] Q. Duan, S. Sorooshian, and V. Gupta, "Effective and efficient global optimization for conceptual rainfall-runoff models," *Water Resources Research*, vol. 28, no. 4, pp. 1015–1031, 1992.

[142] Q. J. Wang, "The genetic algorithm and its application to calibrating conceptual rainfall–runoff models," *Water Resources Research*, vol. 27, no. 9, pp. 2467–2471, 1991.

[143] Qpzm, "North dandalup demographics (wa) local stats," http://north-dandalup.localstats.com.au/demographics/wa/south-western/south-west/north-dandalup, 2016, Retrieve on 20-11-2016.

[144] R. A. Freeze and J. A. Cherry, *Groundwater*. New Jersey: Prentice-Hall Inc, 1979.

[145] R. C. Ward and M. Robinson, *Principles of hydrology*, 3rd ed. London: Mc-Graw Hill Book Company, 1990.

[146] R. E. White, *Principles and practice of soil science: the soil as a natural resource*, 3rd ed. Melbourne: Blackwell Science, 1997.

[147] R. Imanirad and J.S. Yeomans, "Swarm intelligence in optimization," in *Recent Advances in Swarm Intelligence and Evolutionary Computation*, X. Yang, Ed. Springer, 2015, vol. 585, pp. 49–69.

[148] R. Kumar, R.K. Agarwal, J.D. Sharma, "Comparison of regression and artificial neural network models for estimation of global solar radiations," *Renewable and Sustainable Energy Reviews*, vol. 52, pp. 1294–1299, 2015.

[149] R. Rojas, *Neural networks*. Berlin: Springer-Verlag, 1996.

[150] R. Rwechungura, M. Dadashpour, and J. Kleppe, "Advanced history matching techniques reviewed," in *Society Petroleum Engineer, London, United Kingdom*, 2011, SPE 142497.

[151] R. S. Blasone, J. A. Vrugt, H. Madsen, D. Rosbjerg, and B. A. Robinson, and G. A. Zyvoloski, "Generalized likelihood uncertainty estimation (glue) using adaptive markov chain monte carlo sampling," *Advances in Water Resources*, vol. 31, pp. 630–648, 2008.

[152] R. Schulze-Riegert, J. Axman, O. Haase, D. Rian, and Y. You, "Optimization methods for history matching of complex reservoirs," in *Reservoir Simulation Symposium, Houston, Texas, USA, 11–14 February*, 2001, SPE66393.

[153] R. Sedaqatvand, M.N. Esfahany, T. Behzad, and M. Mohseni, "Parameter estimation and characterization of a single-chamber microbial fuel cell for dairy wastewater treatment," *Bioresource technology*, vol. 146, pp. 247–253, 2013.

[154] R. Serfling, "Depth functions in nonparametric multivariate inference," in *DIMACS: Series in discrete mathematics and theoritical computer science*, D. L. S. R. Y. Liu, R. Serfling, Ed. American Mathematical Society, 2006, pp. 1–16.

[155] R. Storn and K. Price, "Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces," Technical Report for International Computer Science Institute, Berkeley, TR-95-012, Tech. Rep., 1995.

[156] R. Venkatesan and V. Kumar, "A genetic algorithms approach to growth phase forecasting of wireless subscribers," *International Journal of Forecasting*, vol. 18, pp. 625–646, 2002.

[157] R. Y. Liu, "On a notion of data depth based on random simplices," *The Annals of Statistics*, vol. 18, pp. 405–414, 1990.

[158] R. Y. Liu, J. M. Parelius, and K. Singh, "Multivariate analysis by data depth: Descriptive statistics, graphics and inference (with discussion)," *Annals of Statistics*, vol. 27, pp. 783–858, 1999.

[159] R. Y. Liu, R. Serfling, D. L. Souvaine, "Data depth: Robust multivariate analysis, computational geometry and applications," in *DIMACS: Series in discrete mathematics and theoritical computer science*, D. L. S. R. Y. Liu, R. Serfling, Ed. American Mathematical Society, 2006, pp. 317–324.

[160] R.S. Blasone, H. Madsen and Dan Rosbjerg, "Parameter estimation in distributed hydrological modelling: comparison of global and local optimisation techniques," *Nordic Hydrology*, vol. 38, no. 4–5, pp. 451–476, 2007.

[161] S. Buyuksaatci, "Bat algorithm application for the single row facility layout problem," in *Recent Advances in Swarm Intelligence and Evolutionary Computation*, X. Yang, Ed. Springer, 2015, vol. 585, pp. 101–120.

[162] S. E. Christodouou and G. Ellinas, "Pipe routing through ant colony optimization," *Journal of Infrastructure System*, vol. 16, no. 2, pp. 149–159, 2010.

[163] S. Fritsch and F. Guenther, *neuralnet: Training of neural networks*, 2012, r package version 1.32. [Online]. Available: http://CRAN.R-project.org/package=neuralnet

[164] S. Goss, J. L. Aron, J.L. Deneubourg, and J. M. Pasteels, "Self-organized shortcuts in the argentine ant," *Naturwissenschaften*, vol. 76, pp. 579–581, 1989.

[165] S. Harun, N. I. A. Nor, and A. H. M. Kassim, "Artificial neural network model for rainfall-runoff relationship," *Jurnal of Teknologi*, vol. 37 (B) Dis., pp. 1–12, 2002.

[166] S. J. Wu, H. C. Lien and C. H. Chang, "Calibration of a conceptual rainfall–runoff model using a genetic algorithm integrated with runoff estimation sensitivity to parameters," *Journal of Hydroinformatics*, vol. 14, no. 2, pp. 497–511, 2011.

[167] S. K. Singh, "Robust parameter estimation in gauged and ungauged basins," Ph.D. dissertation, Civil and Environmental Engineering, University of Stuttgart, 2010.

[168] S. Li, Y. Liu, and H. Yu, "Parameter estimation approach in groundwater hydrology using hybrid ant colony system," in *Computational Intelligence and Bioinformatics*, vol. 4115 of LNCS. Berlin, Germany: Springer-Verlag, 2006, pp. 182–191.

[169] S. Pedersen, T. Randen, L. Sonneland, and O. Steen, "Automatic 3d fault interpretation by artificial ants," in *EAGE 64th Conference and Exhibition, Florence, Florence, Italy, 27–30 May*, 2002.

[170] S. Riad and J. Mania, "Rainfall-runoff model using an artificial neural network approach," *Mathematical and Computer Modelling*, vol. 40, pp. 839–846, 2004.

[171] S. Selberg, R. Schulze-Riegert, and K. Stekolschikov, "Event–targeting model calibration used for history matching large simulation cases," in *Reservoir Simulation Symposium, Houston, Texas, USA, 26–28 February*, 2007, SPE106044.

[172] S. Srinivasulu and A. Jain, "A comparative analysis of training methods for artificial neural network rainfall-runoff models," *Applied Soft Computing*, vol. 6, no. 3, pp. 295–306, 2006.

[173] T. Krausse and J. Cullmann, "Toward a more representative parametrisation of hydrologic models via synthesizing the strength of particle swarm optimisation and robust parameter estimation," *Hydrology and Earth System Sciences*, vol. 16, pp. 603–629, 2012.

[174] T. M. Carpenter and K. P. Georgakakos, "Inter comparison of lumped versus distributed hydrologic model ensemble simulations on operational forecast scales," *Journal of Hydrology*, vol. 329, pp. 174–185, 2006.

[175] T. W. Simpson, J. D. Peplinski, P. N. Koch, and J. K. Allen, "Metamoels for computer-based engineering design: Survey and recomendations," *Engineering with Computers*, vol. 17, no. 2, pp. 129–150, 2001.

[176] T.J. Mulvaney, "On the use of self-registering rain and flood gauges in making observations of rainfall and flood discharges in a given catchment," *Transactions of the Institution of Civil Engineers of Ireland*, vol. IV, no. II, pp. 19–33, 1851.

[177] Trading Economics, "Australia recession 2016," http://www.tradingeconomics.com/australia/population, 2016, retrieve on 17-02-2016.

[178] U. Groemping, *DoE.wrapper: Wrapper package for design of experiments functionality*, 2014, r package version 0.8-10. [Online]. Available: http://CRAN.R-project.org/package=DoE.wrapper

[179] Udayraj, K. Mulani, P. Talukdar, A. Das, and R. Alagirusamy, "Performance analysis and feasibility study of ant colony optimization, particle swarm optimization and cuckoo search algorithms for inverve heat transfer problems," *International Journal of Heat and Mass Transfer*, vol. 89, pp. 359–378, 2015.

[180] V. Klemes, "Conceptualisation and scale in hydrology," *Journal of Hydrology*, vol. 65, pp. 1–23, 1983.

[181] V. Maniezzo, A. Colorni, and M. Dorigo, "The ant system applied to the quadratic assignment problem, technical report iridia/94-28," Universite Libre de Bruxelles, Belgium, Tech. Rep., 1994.

[182] W. Chamley, "Australia's water crisis," http://www.fabians.org.au, 2014, retrieve on 26-11-2016.

[183] W. D. Kruger, "Determining areal permeability distribution by calculations," in *35th Annual Fall Meeting, Denver, USA, 2–5 October*, 1960, SPE1580.

[184] W. J. Welch, R. J. Buck, and J. Sacks, "Screening, predicting, and computer experiments," *Technometrics*, vol. 34, pp. 15–25, 1992.

[185] W. N. Venables and B. D. Ripley, *Modern applied statistics with S*, 4th ed. New York: Springer, 2002, ISBN 0-387-95457-0. [Online]. Available: http://www.stats.ox.ac.uk/pub/MASS4

[186] Water Corporation, "How many litres of water do people use in perth each year?" http://www.smh.com.au/national/water-consumption-in-the-act-continues-to-rise-20141127-11vbml.html, 2016, Retrieve on 20-11-2016.

[187] Water Resources Division, "North dandalup pipehead dam catchment area drinking water source protection plan: Integrated water supply system, water resource protection report series no. wrp 55," Department of Environment, Tech. Rep., 2005, Accessed: 14/09/2015.

[188] ——, "South dandalup dam catchment area and south dandalup pipehead dam catchment area drinking water source protection plan: Integrated water supply system, water resource protection report series no. wrp 55," Department of Environment, Tech. Rep., 2005, Accessed: 14/09/2015.

[189] W.Semmler and G. Gong, "Estimating parameters of real business cycle models," *Journal of Economic Behavior and Organization*, vol. 30, pp. 301–325, 1996.

[190] X. Sun, W. Sun, J. Wang, Y. Zhang, and Y. Gao, "Using a grey-markov model optimized by cuckoo search algorithm to forecast the annual foreign tourist arrivals to china," *Tourist Management*, vol. 52, pp. 369–379, 2016.

[191] X. Xie and D. Zhang, "Data assimilation for distributed hydrological catchment modelling via ensemble kalman filter," *Advances in Water Resources*, vol. 33, no. 6, pp. 678–690, 2010.

[192] X.S. Yang, "Firefly algorithm," *Nature-Inspired Metaheuristic Algorithms 20*, pp. 79–90, 2008.

[193] ——, "A new metheuristic bat-inspired algorithm, in: Nature inspired cooperative strategies for optimization (nisco 2010," in *Recent Advances in Swarm Intelligence and Evolutionary Computation*, D. A. P. C. Cruz, J.R Gonzales and G. Terrazzas, Eds. Springer, 2010, vol. 284, pp. 65–74.

[194] ——, "Multiobjective firefly algorithm for continuous optimization," *Engineering with Computers*, vol. 29, no. 2, pp. 175–184, 2013.

[195] X.S. Yang and S. Deb, "Cuckoo search via levy flights," in *Procedeeings of world congress on nature and biologically inspired computing (NaBIC 2009), India.* IEEE, 2009, pp. 210–214.

[196] ——, "Multiobjective cuckoo search for design optimization," *Computers and Operation Research*, vol. 40, pp. 1616–1624, 2013.

[197] X.S. Yang, M. Karamanoglu, and S. Fong, "Bat algorithm for topology optimization in microelectronic applications," in *Future Generation Communication Technology (FGCT), 2012 International Conference on 12-14 December 2012, London*, 2012, pp. 150–155, IEEE.

[198] Y. Ding, C. Wang, M. Chaos, R. Chen and S. Lu, "Estimation of beech pyrolysis kinetic parameters," *Bioresource technology*, vol. 200, pp. 658–665, 2016.

[199] Y. Gu and D.S. Oliver, "History matching of the punq-s3 reservoir model using the ensemble kalman filter," *SPE Journal*, vol. 10, no. 02, pp. 217–224, 2005.

[200] Y. Hajizadeh, M. Christie, and V. Demyanov, "Ant colony optimization for history matching and uncertainty quantification of reservoir models," *Journal of Petroleum Science and Engineering*, vol. 77, pp. 78–92, 2011.

[201] Y. Li, A. B. Chan Hilton, and L. Tong, "Development of ant colony optimization for long-term groundwater monitoring," in *Proceedings of the 2004 World Water and Environmental Resources Congress, Salt Lake City, UT*, 2004, pp. 1–10.

[202] Y. Li and A. B. Chan Hilton, "Analysis of the primal and dual problem for long-term groundwater monitoring spatial optimization," in *Proceedings of the 2005 World Water and Environmental Resources Congress, May 15–19, 2005, Anchorage, Alaska*, 2005.

[203] ——, "An algorithm for groundwater long-term monitoring spatial optimization by analog to ant colony optimization for tsp," in *Proceedings of the 2006 World Environmental and Water Resources Congress, May 21–25, 2006, Omaha, Nebraska*, 2006.

[204] Y. Wang, J. Li, J. Gu, Z. Zhou, and Z. Wang, "Artificial neural networks for infectious diarrhea prediction using meteorological factors in shanghai (china)," *Applied Soft Computing*, vol. 35, pp. 280–290, 2015.

[205] Y. Zuo and R. Serfling, "General notions of statistical depth function," *Annals of Statistics*, vol. 28, pp. 461–482, 2000.

[206] Y.B. Liu, O. Batelaan, F. De Smedt, J. Poórová, and L. Velcická, "Automated calibration applied to a gis-based flood simulation model using pest," in *Floods, from Defence to Management*, v. B. van Alphen and Taal, Eds. London: Taylor Francis Group, 2005, pp. 317–324.

Appendix
History Matching and Statistical Methods Codes

Ant Colony Optimization (ACO_R)

```
#Ant Colony Optimization (ACOR) for history matching

ACO_output<-ACO(input_par,obs_resp, file_to_run_model,file_input_parameter,
                file_output) {
  #The function requires:
      #*a set of input parameter values (with size of nxp); n:number solutions,
                                                  #p:number of input parameters
      #*a set of observed response
      #*a bat file to run the hydrological model
      #*a file containing input parameter for the hydrological model
      #*a file of output
  #the output of function:
      #*a history of the best optimized input parameters with corresponding RMSE
        #values from each iteration
      #*computational time required for history matching process

  #Rpackage required DoE.wrapper

  ###Subfunctions:

  ###A. Compute model output
  ####  The function requires:
  #*a vector of input parameter (1xp), p:number of input parameters
  #*a set of observed response
  #*a bat file to run the hydrological model
  #*a file containing input parameter for the hydrological model
  #*a file of output

  #### The output is the model response
```

```
mod_output<-function(input_par,file_to_run_model,file_input_parameter,file_output){
  #Extract input parameters
  input_par[1]<-x[1] #It can be added to number of parameters concerned to be
  input_par[2]<-x[2] #calibrated
  input_par[3]<-x[3]
  input_par[4]<-x[4]
  input_par[5]<-x[5]
  input_par[6]<-x[6]

  #Write input parameters in Lucicat.par
  write.table(file_input_parameter)

  #Run the LUCICAT model
  system(file_to_run_model)

  #Extract the output
  mod_response<-read.csv(file_output)

  #Convert to matrix form
  mod_response<-matrix(mod_response, nrow=1)

  return(mod_response)

}

###B. Compute RMSE
####  The function requires:
#*a vector of input parameter (1xp) (input_par), p:number of input parameters
#*a set of observed response (obs_resp)
#*a bat file to run the hydrological model (file to run_model)
#*a file containing input parameter for the hydrological model (file_input_parameter)
#*a file of output (file output)

#### The output is the model response
obj_func<-function(input_par,obs_resp,file_to_run_model,file_input_parameter,
                   file_output) {
  # Assign variable
  n_ants<-dim(input_par)[1]  #number of ants are created
  n_points<-length(obs_resp) #number of points to be matched
  y<-matrix(0, nrow=n_ants, ncol=n_points)
```

```
  #Find the output by running the LUCICAT for every set of input parameter
  for (i in 1:n_ants) {
    #print(x_new[i,])
    single_x<-input_par[i,]
    y[i,]<-mod_output(single_x, obs_resp, file_to_run_model,file_input_parameter,
                      file_output)

  }

  #Calculate RMSE
  RMSE<-matrix(0,nrow=n_ants, ncol=1)

  #replicate the observed response for calculating RMSE
  y_dat<-matrix(rep(obs_resp,each=dim(y)[1]),nrow=dim(y)[1])
  RMSE<-sqrt((rowSums((y-y_dat)^2))/n_points)

  return(RMSE)
}

###C. Select solutions to put into the bins. This will be used for
   ###sampling new solutions
###The input are index, number of ants (m_new) and number of model
   ###solutions (k_sol)
###The outputs is a matrix of bins (nxk)

pop_bins<-function(index,m_new,k_sol) {
  #Assign the variables
  len<-length(index)
  idx<-matrix(0,ncol=len)
  const<-0

  #Construct the bins
  for (i in (m_new+1):len) {
    if (((i-1)%%m_new)==0) {
      const=const+1
      val=const*k_sol

    }
    idx[i]=val
```

```
  }
  return(idx)
}

###D. Generating new solutions
###convert normal random values generated
###from mu=0,sd=1 to the values generated from mean=mu
###standard deviation (sigma)
###The conversion is based on the index of the good
###performing input parameters described in
### population bins matrix.

###The inputs are :
###*the index (bins),
###*the new generations of standard normal distribution (normal_rands)
###*the standard deviations(sigmas),
###*the input parameter (input_par)
###*the the minimum boundary of input parameter (par_min)
###*the the maximum boundary of input parameter (par_max)
###*the number of ants (n_new_ants)

###The output is new solutions (x_new)

xnew<-function(index, normal_rands,sigmas,input_par,par_min,par_max,n_new_ants) {
  #Initialize variables
  x1<-matrix(0,nrow=dim(normal_rands)[1],ncol=dim(normal_rands)[2])
  x_new<-matrix(0,nrow=dim(normal_rands)[1],ncol=dim(normal_rands)[2])
  new_sigmas<-matrix(0,nrow=dim(normal_rands)[1],ncol=dim(normal_rands)[2])

  #
  for (i in 1:length(new_sigmas)) {
    #select sigma for new solution at ith row
    new_sigmas[i]<-sigmas[index[i]]

    #select mu for new solution at ith row
    x1[i]<-input_par[index[i]]

    #convert normal random value at row ith to
    #normal value (mean=mu and sd=sigma) at row ith
    #bounded the new solutions with predefined ranges of values
```

```
    x_new[i]<-normal_rands[i]*new_sigmas[i]+x1[i]

    n=1

    if (i<=n_new_ants){
    } else {
      if ((i-1)%%n_new_ants==0) {
        n=n+1
      }
    }
    x_new[i]<-min(max(par_min[n],x_new[i]),par_max_max[n])
  }

  return(x_new)
}

#########################################################################
####Main Program

###Call the package to generate initial input parameter values using LHD
require(DoE.wrapper)

###Retrieve the observed streamflow files and retrieve input parameter
y_data<-obs_resp   #streamflow in m3/day
y_data<-y_data/86400    #convert observed streamflow into m3/sec
dt<-file_input_parameter   #assign input parameter

###Generate initial input parameter values using LHD
x0<-input_par

###Set the predefined boundary for each input parameter
x_min<-matrix(c(1.8, 15.3,0.156,350,0.156),nrow=1)
x_max<-matrix(c(3.1, 27.2,0.56,1000,0.56),nrow=1)

###Assign parameters of ACOR
eps<-0.5  #eps determines how quickly ACOR forgets the bad solutions
          #large eps results in slow convergence as the ACOR revisits the
#previous regions
q<-0.05   #when q is small, the generation of ants are selected from the rank of
```

```
          #solutions

#Assign variables
iter_count<-0     #initiate iteration count
max_iter_count<-1 #maximum iteration
min_val<--Inf;    #used for creating bins
k_sol<-dim(x0)[1] #number of model solutions in the archive solutions
m_new<-k_sol       #number of ants generated for each iteration

###Set the initial time
ptm <- proc.time()

###Convert data frame x0 into matrix domain and denoted as x
x<-as.matrix(x0)

###Assign number of input parameters
num_dim<-dim(x)[2]

###Compute the objective function values of x and combined with
###their objective values
vals_old_x<-obj_func(x,y_data, file_to_run_model, dt, file_output)
com_old_x_obj<-cbind(x,vals_old_x)

###Sort the arrays based on objective function
com_old_x_obj<-com_old_x_obj[order(com_old_x_obj[,num_dim+1]),]

###Reassign x based on the order objective function
x<-com_old_x_obj[1:k_sol,1:num_dim]

###Compute weight w. Weights are actually constants
###w1>w2>w3>....It means that the best objective function (minimum value)
###with highest w has a chance to be selected
weights<-exp(-((1:k_sol)-1)^2/(2*q^2*k_sol^2))/(q*k_sol*sqrt(2*pi))

###Calculate the probability and cumulative probability that will
#be used later for sampling
probs<-weights/sum(weights)
cum_probs<-cumsum(probs)

###Create empty matrix to store convergence history of the best solution
```

```
###obtained of each iteration and solution history of x
convg_history<-matrix(rep(1,max_iter_count*(num_dim+1)),nrow=max_iter_count)
x_history<-matrix(0,nrow=k_sol*max_iter_count,ncol=num_dim+1)

###Start iteration process to find the optimal input parameter values (x)
while (abs(min_val)>0.0001 && iter_count<max_iter_count) {

  #distance between xi and any other x values
  distances<-x[rep(1:k_sol,k_sol),]-matrix(rep(x,each=k_sol),nrow=k_sol*k_sol)
  distances<-matrix(distances,nrow=k_sol)

  #sigma is the average distance
  sigmas<-eps*colSums(abs(distances))/(k_sol-1)
  sigmas<-matrix(sigmas,nrow=k_sol)

  #compute bins
  bins<-qnorm(cum_probs, 0,1)

  #Define max value of bins
  for(i in 1:length(bins)) {
    if (bins[i]==Inf){
      max_val<-bins[i-1]
      break
    }

  }

  #Assign infinite values generated by qnorm (normal inverse) with
  #maximum numeric values of bins
  for( i in 1:k_sol) {
    if(is.infinite(bins)[i])
      bins[i]<-max_val
  }

  #Generate normal random with dimension m_new x num_dim as defined above
  normal_rands<-matrix(rnorm(m_new*num_dim,0,1),m_new,num_dim)

  #Compute the difference between each random values and each value of bins.
  diff<-matrix(rep(normal_rands,each=k_sol),nrow=k_sol*m_new)-
        matrix(rep(bins,m_new*num_dim),nrow=k_sol*m_new)
```

```
diff<-matrix(diff,nrow=k_sol,ncol=m_new*num_dim)

#Find the bin with the smallest difference
#index<-min_ind(abs(diff))
index<-matrix(0,ncol=m_new*num_dim)
for (i in 1:(m_new*num_dim)) {
  index[i]<-which.min(abs(diff[,i]))

}
index<-pop_bins(index,m_new,k_sol)+index
index<-matrix(index,nrow=m_new)

#Convert N(0,1) into N(mu, sigma) and combine with initial x array
x_new_ants<-xnew(index, normal_rands,sigmas,x,x_min,x_max,m_new)

#Evaluate the objective function
vals_new_ants<-obj_func(x_new_ants, y_data, file_to_run_model, dt, file_output)

#Combine new ants with their objective values
com_new_x_obj<-cbind(x_new_ants,vals_new_ants)

#Combine old ants with new ants
com_x_obj<-rbind(com_old_x_obj,com_new_x_obj)

#Sort the arrays based on objective function
com_x_obj<-com_x_obj[order(com_x_obj[,num_dim+1]),]

#Extract the k_sol good solutions and put in com_old_x_obj
#Reassign x
com_old_x_obj<-com_x_obj[1:k_sol,]
x<-com_x_obj[1:k_sol,1:num_dim]

#Extract the input parameter set with lowest value of RMSE for
#each iteration and store it in convg_history and
convg_history[iter_count+1,]=com_old_x_obj[1,]

#Add up the iteration count
iter_count<-iter_count+1

#Termination criteria if total sum of differences between 4
```

```
    #consecutive iterations of objective function
    #is less than a threshold, we stop the process
    if (iter_count>=8){
      if (abs(convg_history[iter_count,num_dim+1]-
              convg_history[iter_count-7,num_dim+1])<0.005){
        break
      }
    }

  }

  ###Compute computational time
  comp_time<-proc.time() - ptm

  ###Create the ouput storage
  output<-list()

  output[[1]]<-convg_history
  output[[2]]<-comp_time

  return(output)

}
```

Diversity Ant Colony Optimization (DACO_R)

```
###Diversity Ant Colony Optimization
###The algorithm was developed by Leguizamon and Coello Coello (2010)

DACO_output<-DACO(input_par,obs_resp, file_to_run_model,file_input_parameter,
                  file_output) {
  #The function requires:
  #*a set of input parameter values (with size of nxp); n:number solutions,
  #p:number of input parameters
  #*a set of observed response
  #*a bat file to run the hydrological model
  #*a file containing input parameter for the hydrological model
  #*a file of output
  #the output of function:
  #*a history of the best optimized input parameters with corresponding RMSE
   #values fromeach iteration
  #*computational time required for history matching process

  #Rpackage required is DoE.wrapper

  ###Subfunctions:

  ###A. Compute model output
  ####  The function requires:
  #*a vector of input parameter (1xp), p:number of input parameters
  #*a set of observed response
  #*a bat file to run the hydrological model
  #*a file containing input parameter for the hydrological model
  #*a file of output
```

```
#### The output is the model response
mod_output<-function(input_par,file_to_run_model,file_input_parameter,file_output){
  #Extract input parameters
  input_par[1]<-x[1] #It can be added to number of parameters concerned to be
  input_par[2]<-x[2] #calibrated
  input_par[3]<-x[3]
  input_par[4]<-x[4]
  input_par[5]<-x[5]

  #Write input parameters in Lucicat.par
  write.table(file_input_parameter)

  #Run the LUCICAT model
  system(file_to_run_model)

  #Extract the output
  mod_response<-read.csv(file_output)

  #Convert to matrix form
  mod_response<-matrix(mod_response, nrow=1)

  return(mod_response)

}

###B. Compute RMSE
####  The function requires:
#*a vector of input parameter (1xp) (input_par),
 #p:number of input parameters
#*a set of observed response (obs_resp)
#*a bat file to run the hydrological model (file to run_model)
#*a file containing input parameter for the hydrological model
 #(file_input_parameter)
#*a file of output (file output)

#### The output is the model response
obj_func<-function(input_par,obs_resp,file_to_run_model,file_input_parameter,
                   file_output) {
  # Assign variable
```

```
  n_ants<-dim(input_par)[1]  #number of ants are created
  n_points<-length(obs_resp) #number of points to be matched
  y<-matrix(0, nrow=n_ants, ncol=n_points)

  #Find the output by running the LUCICAT for every set of input parameter
  for (i in 1:n_ants) {
    #print(x_new[i,])
    single_x<-input_par[i,]
    y[i,]<-mod_output(single_x, obs_resp, file_to_run_model,file_input_parameter,
                      file_output)

  }

  #Calculate RMSE
  RMSE<-matrix(0,nrow=n_ants, ncol=1)

  #replicate the observed response
  y_dat<-matrix(rep(obs_resp,each=dim(y)[1]),nrow=dim(y)[1])

  #Calculate RMSE
  RMSE<-sqrt((rowSums((y-y_dat)^2))/n_points)

  return(RMSE)
}

###C. Function to check the boundaries of input parameter values
###The inputs are :
###*a set of input parameter matrix size of nxk (input_par)
###*a vector of minimum boundary of input par (par_min)
###*a vector of maximum boundary of input par (par_max)

###The output is a matrix of input parameter matrix within the ranges of
###the boundary

boundary<-function(input_par,par_min,par_max){
  #Assign variables
  m_ant<-dim(input_par)[1]    #the number of rows/solutions
  num_par<-dim(input_par)[2]  #the number of input parameters
  input_par_new=matrix(0,nrow=m_ant,ncol=num_par) #initialize the new x matrix
```

```
  for (i in 1:m_ant){
    for (j in 1:num_par){
      input_par_new[i,j]<-min(max(par_min[j],x[i,j]),par_max[j])
    }

  }
  return(input_par_new)
}

#############################################################################
####Main Program

###Call the package to generate initial input parameter values using LHD
require(DoE.wrapper)

###Retrieve the observed streamflow files and retrieve input parameter
y_data<-obs_resp   #streamflow in m3/day
y_data<-y_data/86400    #convert observed streamflow into m3/sec
dt<-file_input_parameter   #assign input parameter

###Generate initial input parameter values using LHD
x0<-input_par

###Set the predefined boundary for each input parameter
x_min<-matrix(c(1.8, 15.3,0.156,350,0.156),nrow=1)
x_max<-matrix(c(3.1, 27.2,0.56,1000,0.56),nrow=1)

###Assign parameters of DACOR
###These parameters need to calibrate before performing history matching (HM)
eps<-0.6  #eps determines how quickly DACOR forgets the bad solutions
          #large eps results in slow convergence as the DACOR revisits the
          #previous regions
u<-0.1    #parameter that controls the switch between global or local search

#Assign variables
iter_count<-0       #initialise iteration count
max_iter_count<-1   #maximum iteration
min_val<--Inf;      #initialise minimum value found so far
k_sol<-dim(x0)[1]   #number of model solutions in the archive solutions
m_new<-k_sol        #m_new has to be the same as k_sol
```

```
###Set the initial time
ptm <- proc.time()

###Create a matrix form of x0
x<-as.matrix(x0)

###Assign number of input parameters
num_dim<-dim(x)[2]

#Evaluate the objective function
vals<-obj_func(x,y_data,file_to_run_model,file_input_parameter,file_output)
com_x_vals<-cbind(x,vals)

###Create empty matrix to store convergence history of the best solution
###obtained of each iteration
convg_history<-matrix(1,nrow=max_iters,ncol=num_dim+1)

###Start iteration process to find the optimal input parameter values (x)
for (iter_count in 1:max_iters) {

  #distance between xi and any other x values
  distances<-x[rep(1:k_sol,k_sol),]-matrix(rep(x,each=k_sol),nrow=k_sol*k_sol)
  distances<-matrix(distances,nrow=k_sol)

  #sigma is the average distance
  sigmas<-eps*colSums(abs(distances))/(k_sol-1)
  sigmas<-matrix(sigmas,nrow=k_sol)

  #Generate normal random with dimension m_new x num_dim as defined above
  normal_rands<-matrix(rnorm(k_sol*num_dim,0,1),k_sol,num_dim)

  #Find the minimum of objectvive function and its minimum index
  min_val<-min(vals)
  min_idx<-which(vals==min_val)

  #Sometimes the minimum values of objective function can be more than 2,
  #this code below is meant to limit number of minimum values to 1
  if (length (min_idx)>1) {
    min_idx<-min_idx[1]
```

```
}

#Create new solution (the algoritm search all possible regions)
if (u > runif(1)) {
  #Convert values from normal_rands to values generated from normal of
  #the corresponding mu and sigmas
  x_new<-normal_rands*sigmas+x
  x_new<-boundary(x_new,x_min,x_max)
} else {

  #Assign the minimum index
  min_index<-min_idx

  #Compute matrix of sigmas based on minimum index
  min_sigmas<-matrix(rep(sigmas[min_index,],each=k_sol),nrow=k_sol)
  min_x<-matrix(rep(x[min_index,],each=k_sol),nrow=k_sol)

  #Convert values from normal_rands to values generated from normal of
  #the best input parameters
  x_new<-normal_rands*min_sigmas+min_x

  #Cek the boundary of new input parameters
  x_new<-boundary(x_new,x_min,x_max)
}

#Calculate objective function for new solutions
vals_new<-obj_func(x,dt,y_obs_index,y_dat)

#Find the minimum objective value and index of new solutions
min_val_new<-min(vals_new)
min_idx_new<-which(vals_new==min_val_new)

#Check if the new x values improves by comparing its corresponding
#values of objective function. If its objective function value is smaller
#than the old one, it will be replaced with the new one.
#Otherwise, the code will hold the old x values and carry it
#to the next iteration
update_idx<-vals_new <vals
vals[update_idx]<-vals_new[update_idx]
x[update_idx,]<-x_new[update_idx,]
```

```
    com_x_vals<-cbind(x,vals)

    #Store the best model solution x in convg_history matrix for each iteration
    convg_history[iter_count,]<-com_x_vals[which.min(vals),]

    #Termination criteria if total sum of differences between 4 consecutive
    #iterations of objective function is less than a threshold,
    #we stop the process

    if (iter_count>=8){
      if (abs(convg_history[iter_count,num_dim+1]-
              convg_history[iter_count-4,num_dim+1])<0.005){
        break
      }
    }


  }

  ###Compute computational time
  comp_time<-proc.time() - ptm

  ###Create the ouput storage
  output<-list()

  output[[1]]<-convg_history
  output[[2]]<-comp_time

  return(output)
}
```

Gauss Levenberg Marquardt (GLM)

```
#Gauss Levenberg Marquardt (GLM) for history matching
#The algorithm was developed by Levenberg (1944) and Marquardt (1963)
#The following codes were adopted from Gavin (2011)

GLM_output<-GLM(input_par,obs_resp, file_to_run_model,file_input_parameter,
                file_output) {

    #The function requires:
    #*a vector of input parameter (1xp), p:number of input parameters
    #*a set of observed response
    #*a bat file to run the hydrological model
    #*a file containing input parameter for the hydrological model
    #*a file of output
    #the output of function:
    #*a history of the best optimized input parameters with corresponding RMSE
     #values from each iteration
    #*computational time required for history matching process

  ###Subfunctions:

  ###A. Compute model output
  ####  The function requires:
        #*a vector of input parameter (1xp), p:number of input parameters
        #*a set of observed response
        #*a bat file to run the hydrological model
        #*a file containing input parameter for the hydrological model
        #*a file of output

  #### The output is the model response
```

```
mod_output=function(input_par,file_to_run_model,file_input_parameter,file_output){
  #Extract input parameters
  input_par[1]<-x[1] #It can be added to number of parameters concerned to be
  input_par[2]<-x[2] #calibrated
  input_par[3]<-x[3]
  input_par[4]<-x[4]
  input_par[5]<-x[5]
  input_par[6]<-x[6]

  #Write input parameters in Lucicat.par
  write.table(file_input_parameter)

  #Run the LUCICAT model
  system(file_to_run_model)

  #Extract the output
  mod_response<-read.csv(file_output)

  #Convert to matrix form
  mod_response<-matrix(mod_response, nrow=1)

  return(mod_response)

}

###B. Compute Jacobian matrix (partial derivative) dy/dx
###The inputs are:
###*a vector of input parameter (input_par) size of 1xp, p:number of
    ###input parameters
###*a set of model output based on input_par (y)
###*a bat file to run the hydrological model (file_to_run_model)
###*a file containing input parameter for the hydrological model
    ###(file_input_parameter)
###*a file of output (file_output)

###The output of function is Jacobian matrix

lm_dydx<-function(input_par,y,dx,file_to_run_model,file_input_parameter,
                  file_output) {
  #Assign variables
```

```
  q<-length(y)                    #number of model response
  p<-length(input_par)            #number of parameter
  xs<-input_par
  dydx<-matrix(rep(0,m*n),nrow=m) #Initialise Jacobian matrix
  del<-matrix(rep(0,n), ncol=1)   #initialise perturbation matrix

  for (j in 1:n) {
    del[j]<-dx[j]*(1+abs(x[j]))   #parameter perturbation
    x[j]<-xs[j]+del[j]            #perturb parameter x[j]

    if (del[j] !=0) {
      y1<-mod_output(x,file_to_run_model,file_input_parameter,file_output)

    }

    if (dx[j]<0) {
      dydx[,j]=(y1-y)/del[j]               #forward difference
    } else {                               #central difference
      x[j]<-xs[j]-del[j]
      y2<-mod_output(x,file_to_run_model,file_input_parameter,file_output)
      dydx[,j]<-(y1-y2)/(2*del[j])
    }
    x[j]=xs[j]
  }


  return(dydx)

}

###C. Compute Hessian matrix
###The inputs are:
###*a vector of input parameter (input_par) size of 1xp,
   ###p:number of input parameters
###*a Jacobian matrix
###*a the weight square

###The output is Hessian matrix

lm_alpha<-function(input_par,dydx,weight_sq) {
```

```
  #Assign the variables
  npar<-length(input_par)       #find number of input parameters
  alpha<-matrix(0, nrow=npar,ncol=npar)  #initialize alpha

  #Compute Hessian matrix
  alpha<-t(dydx)%*%(dydx*matrix(rep(weight_sq,npar),ncol=npar))

  return(alpha)

}

###D. Compute the update parameters according to
###gradient descent method t(J)*W*(observed y-model y)
###The input are:
###*a vector of input parameter (input_par) size of 1xp,
   ###p:number of input parameters
###*a Jacobian matrix (dydx)
###*a diagonal matrix of sigma^2 (variance of the measurement error of
    ###observed data)
###streamflow and model output (delta_y), Jacobian matrix (dydx)
   ###and weight_sq

###The output is a matrix of update parameters based on gradient descent method

lm_beta<-function(input_par,delta_y,dydx,weight_sq) {
  #Assign the variables
  npar<-length(input_par)            #find the number of input parameters
  beta<-matrix(0, nrow=npar,ncol=1)  #initialize beta

  #Compute the update parameters based on beta
  beta<-t(dydx)%*%(weight_sq*delta_y)
  return(beta)

}

###E. compute the objective function (sum square error)
###The input are :
###*a vector of the differences between observed streamflow and
   ###model output (delta_y)
###*a diagonal matrix of sigma^2 (variance of the measurement error
```

```
###   of observed data)

###The output is a sum square error value

lm_phi=function(delta_y,weight_sq) {
  #Compute phi
  phi=t(delta_y)%*%(delta_y*weight_sq)
  return(phi)

}

#########################################################################
####Main Program of Gauss Levenberg Marquardt Algorithm

###Set initial time
ptm <- proc.time()

###Retrieve the observed streamflow files and retriev input parameter
y_data<-obs_resp   #streamflow in m3/day
y_data<-matrix(y_data/86400, ncol=1)    #convert observed streamflow into m3/sec
dt<-file_input_parameter   #retrieve global parameter from Luc

###Assign initial input parameters
x0<-matrix(input_par,ncol=1)

###Assign number of parameters and number response points need to be matched
npar<-length(X0)
num_dim<-length(y_data)

###Assign the weigth of each point to be matched
sigma<-sd(obs_resp)
weight<-matrix(sigma, nrow=num_dim, ncol=1)

###Set the predefined boundary for each input parameter
x_min<-matrix(c(1.8, 15.3,0.156,350,0.156),ncol=1)
x_max<-matrix(c(3.1, 27.2,0.56,1000,0.56),ncol=1)

###Assign parameter perturbation for Jacobian matrix calculation
dx<--0.001 #needs to be calibrated before algorithm
           #used for history matching (HM)
```

```
###Assign p with initial input parameters
x<-x0

### Assign maximum iteration
max_iter<-100

###Assign parameters and termination criteria required for GLM
###These parameters need to be calibrated before algorithm used for HM
epsilon_1<-1e-6    #threshold for convergence in the update parameter
                   #based on gradient descent
epsilon_2<-1e-6    #threshold for convergence in parameter
epsilon_3<-1e-5    #threshold for convergen in objective value
epsilon_4<-1e-6    #threshold for gain factor
lambda_0<-1e-3     #intial lambda value
lambda_UP_fac<-11  #the  increasing factor
lambda_DN_fac<-9   #the decreasing factor

###Create vector of perturbation matrix
if (length(dx)==1) {dx=dx*matrix(rep(1,npar),ncol=1)}

###Assign termination for stop the process of HM
halt<-0

###Create vectors of weight
if (length(weight)<num_dim) {weight_sq=(matrix(rep(weight,num_dim),ncol=1))^2
} else {weight_sq=weight^2}

###Calculate initial model outputs based on initial input parameters (p_init)
y_hat<-mod_output(x,file_to_run_model,file_input_parameter,file_output)

###calculate Jacobian matrix (dydp),
###and model outputs(delta_y), Hessian matrix (alpa),
###gradient descent update (beta),
###and objective function, sum square error (phi)

###Compute Jacobian matrix (dydp)
dydp<-lm_dydx(x,y_hat,dx,file_to_run_model,file_input_parameter,file_output)

###Compute the differences between observed streamflow
```

```
delta_y<-y_dat-y_hat

###Compute Hessian matrix (alpa)
alpha<-lm_alpha(x,dydp,weight_sq)

###Compute the update parameter based gradient descent method (beta)
beta<-lm_beta(x,delta_y,dydp,weight_sq)

###Compute the objective function (phi:sum square error)
phi<-lm_phi(delta_y,weight_sq)

###Assign lambda and old objective function for comparison between iterations
lambda<-lambda_0
phi_old<-phi

###Create empty matrix for storing the convergent history
convg_history<-matrix(rep(1,max_iter*(npar+2)),nrow=max_iter)

###Assign first iteration
iter<-1

###Start iteration process to find the optimal input parameter values (x)
while (!halt && iter<=max_iter) {

  #Update the input parameters by combining Gauss Newton and
  #gradient descent method controlled by Marquardt parameter lambda
  dxlm<-solve((alpha+lambda*diag(diag(alpha))),beta)
  x_try=x+dxlm

  #Check whether the new input parameter values within the ranges of preset
  #input parameters
  for (i in 1:npar) {
    x_try[i,1]<-min(max(x_min[i],x_try[i]),x_max[i])
  }

  #Calculate model outputs based on new input parameters (p_try)
  y_hat<-mod_output(x_try,file_to_run_model,file_input_parameter,file_output)

  #Calculate the differences between observed streamflow and
  #model outputs (y_hat)
```

```
delta_y<-y_dat-y_hat

#Calculate the sum square error based on new input parameters (p_try)
phi_try<-lm_phi(delta_y,weight_sq)

#Calculate the gain factor to determine the update lambda
#according to Nielsen (1999)
rho=(phi-phi_try)/(2*t(dxlm))%*%(lambda*dxlm+beta)

#Check the gain factor is greater than a threshold
if (rho>epsilon_4) {

  #Accept x_try and assign into x
  x=x_try
  #Calculate Jacobian matrix (dydp), the Hessian matrix (alpha),
  #the update based on gradient descent method (beta) and
  #sum square error (phi)

  ###Compute the differences between observed streamflow
  delta_y<-y_dat-y_hat

  ###Compute Hessian matrix (alpa)
  alpha<-lm_alpha(x,dydp,weight_sq)

  ###Compute the update parameter based gradient descent method (beta)
  beta<-lm_beta(x,delta_y,dydp,weight_sq)

  phi<-phi_try

  #Update lambda by decreasing it by a decreasing factor
  lambda<-max(lambda/lambda_DN_fac,1e-7)

  #Assign old sum square error with new phi
  phi_old<-phi

} else {      #if the gain factor is less than a threshold

  phi<-phi_old

  #Update lambda by a increasing factor
```

```
      lambda<-min(lambda*lambda_UP_fac,1e7)
    }

    #Store the optimal input parameters and lambda in conv_history matrix
    convg_history[iter,]<-cbind(t(p),sqrt(phi/num_dim),lambda)

    #Termination criteria proposed by Gavin (2011). There are some choices:
    #convergence in beta
    #1.if (max(abs(beta)) < epsilon_1  &  iter > 2 ) {halt=1}

    #convergence in parameter
    #2.if (max(abs(hlm/p))<epsilon_2 && iter>2) {halt=1}

    #convergence in objective function
    #3.if (sqrt(phi/num_dim)<epsilon_3 && iter>2) {halt=1}

    #This study used criteria no. 3 for HM based multiple objective function
    #but for  single objective function, we used the following criteria
    #based on Blasone (2007)
    #Termination criteria if total sum of differences between 4 consecutive
    #iterations of objective function is less than a threshold,
    #we stop the process
    if (iter_count>=8){
      if (abs(convg_history[iter_count,num_dim+1]-
              convg_history[iter_count-7,num_dim+1])<0.005){
        break
      }
    }

    #Increment the iteration number
    iter<-iter+1
  }

}
```

Robust Parameter Estimation (ROPE)

```
###Robust Parameter Estimation (ROPE)
###The algorithm was developed by Bardossy and Singh (2008)

ROPE_output<-ROPE(input_par,obs_resp, file_to_run_model,file_input_parameter,
                  file_output) {
  #The function requires:
  #*a set of input parameter values (with size of nxp); n:number solutions,
  #p:number of input parameters
  #*a set of observed response
  #*a bat file to run the hydrological model
  #*a file containing input parameter for the hydrological model
  #*a file of output
  #the output of function:
  #*a history of the best optimized input parameters with corresponding RMSE
    #values from each iteration
  #*computational time required for history matching process

  #Rpackages required are DoE.wrapper and depth

  ###Subfunctions:

  ###A. Compute model output
  ####  The function requires:
  #*a vector of input parameter (1xp), p:number of input parameters
  #*a set of observed response
  #*a bat file to run the hydrological model
  #*a file containing input parameter for the hydrological model
  #*a file of output

  #### The output is the model response
```

```
mod_output<-function(input_par,file_to_run_model,file_input_parameter,file_output){
  #Extract input parameters
  input_par[1]<-x[1] #It can be added to number of parameters concerned to be
  input_par[2]<-x[2] #calibrated
  input_par[3]<-x[3]
  input_par[4]<-x[4]
  input_par[5]<-x[5]

  #Write input parameters in Lucicat.par
  write.table(file_input_parameter)

  #Run the LUCICAT model
  system(file_to_run_model)

  #Extract the output
  mod_response<-read.csv(file_output)

  #Convert to matrix form
  mod_response<-matrix(mod_response, nrow=1)

  return(mod_response)

}

###B. Compute RMSE
####  The function requires:
#*a vector of input parameter (1xp) (input_par), p:number of input parameters
#*a set of observed response (obs_resp)
#*a bat file to run the hydrological model (file to run_model)
#*a file containing input parameter for the hydrological model
 #(file_input_parameter)
#*a file of output (file output)

#### The output is the model response
obj_func<-function(input_par,obs_resp,file_to_run_model,file_input_parameter,
                   file_output) {
  # Assign variable
  n_sol<-dim(input_par)[1]  #number of solutions are created
  n_points<-length(obs_resp) #number of points to be matched
  y<-matrix(0, nrow=n_sol, ncol=n_points)
```

```
  #Find the output by running the LUCICAT for every set of input parameter
  for (i in 1:n_sol) {
    single_x<-input_par[i,]
    y[i,]<-mod_output(single_x, obs_resp, file_to_run_model,file_input_parameter,
                      file_output)

  }

  #Calculate RMSE
  RMSE<-matrix(0,nrow=n_sol, ncol=1)

  #Replicate the observed response
  y_dat<-matrix(rep(obs_resp,each=dim(y)[1]),nrow=dim(y)[1])

  #Calculate RMSE
  RMSE<-sqrt((rowSums((y-y_dat)^2))/n_points)

  return(RMSE)
}

#############################################################################
####Main Program
###Call the package to generate initial input parameter values using LHD
require(DoE.wrapper)

###Retrieve the observed streamflow files and retrieve input parameter
y_data<-obs_resp          #streamflow in m3/day
y_data<-y_data/86400      #convert observed streamflow into m3/sec
dt<-file_input_parameter  #assign input parameter

###Generate initial input parameter values using LHD
x0<-input_par

###Find the number of solutions and number of parameters of x0
n<-dim(x0)[1]
num_dim<-dim(x0)[2]

###Assign variables
m=1000                 #the number of solutions for next iteration
```

```
max_iters <- 5          #maximum number of iterations
mse_tolerance <- 0.005 #a threshold for convergence

###Take x0 as a matrix
Xn=as.matrix(x0)

#nvar=dim(Xn)[2]
#nrun=dim(Xn)[1]

###Set the predefined boundary for each input parameter
X_min=matrix(c(1.8,15.3,0.156,400,0.156),nrow=1)
X_max=matrix(c(3.1,27.2,0.56,1000,0.56),nrow=1)

###Create empty matrix to store convergence history of the best solution
###obtained of each iteration
convg_history<-matrix(1,nrow=max_iters,ncol=num_dim+1)

for (i in 1:max_iters) {

  #Find the objective values for Xn
  obj_func_value=obj_func(Xn,obs_resp,file_to_run_model,file_input_parameter,
                          file_output)

  if(i == 1) {
    if (n!=m){
      prev_mean<-mean(obj_func_value)
      rank_list<-rank(obj_func_value)
      index_rank<-which(rank_list<(m+1))
      prev_obj_val<-obj_func_value[index_rank]
      prev_Xn<-Xn[index_rank,]

    } else {
      prev_mean<-mean(obj_func_value)
      prev_obj_val<-obj_func_value
      prev_Xn<-Xn
    }

  } else {

    #take the samples that show improvement over the previous ones
```

```
  idx <- which(prev_obj_val < obj_func_value)
  obj_func_value[idx] <- prev_obj_val[idx]
  Xn[idx, ] <- prev_Xn[idx, ]
  mean_obj=mean(obj_func_value)
  prev_obj_val <- obj_func_value
  prev_Xn <- Xn

}

###combine the objective fn and the set of input values
dsgn_obj=cbind(Xn,obj_func_value)

###Sort the objective values in increasing order
dsgn_obj_inorder<-dsgn_obj[order(dsgn_obj[,nvar+1]),]

###Store the best model solution x in convg_history matrix
###for each iteration
convg_history[i,]<-dsgn_obj_inorder[1,]

###Termination criteria when the differences between current mean of
###objective value and previous mean of objectve value is less than
###a threshold

if (i>1) {
  if ((abs(mean_obj-prev_mean))< mse_tolerance) {
    print(paste("Total iters: ", i))
    break
  } else {
    prev_mean=mean_obj
  }
}

###Select the best 10% performance (the selected percentage needs
###to be optimized)

Xn_star=dsgn_obj_inorder[1:round(0.1*dim(Xn)[1]),1:nvar]

###New generation of solution is sampled from random normal
###with mean and sigma are determined from the best 10% performances
###of input parameters (Xn_star)
```

```
Xm=matrix(0,nrow=second_iter,ncol=nvar) #Intialize matrix for new solutions

for (j in 1:second_iter) {
  test=0
  while (test!=1) {

    Xm_temp<-matrix(0,nrow=1,ncol=nvar)

    for(k in 1:nvar) {
      ###Generate new sample around Xn_star
      Xm_temp[k] <- rnorm(1,mean(Xn_star[, k]),sd(Xn_star[, k]))

      ###Check the boundary of new sample
      Xm_temp[k] <-min(max(X_min[k], Y[k]),X_max[k])
    }



    ###Application of data depth to check whether the new sample has depth
    ###is greater than 0 with respect to Xn_star.
    ###Depth is defined as how depth a point with respect to a set of data
    ###which is in this case, Xn_star

    depth_Xm_temp=depth(Xm_temp,Xn_star,method="Tukey", approx=T)

    ###Filter which input parameters have depths are greater than 0
    ###and store them in Xm
    if (depth_Xm_temp>0) {
      Xm[j,]=Xm_temp        #and store it within new matrix Ym
      test=1

    }
  }

}

#Assign Xn with Xm and start iteration until the average performances
###between Xm and Xn are within a threshold or maximum iteration is
###achieved
Xn=Xm
```

```
  }

  ###Compute computational time
  comp_time<-proc.time() - ptm

  ###Create the ouput storage
  output<-list()

  output[[1]]<-convg_history
  output[[2]]<-comp_time

  return(output)

}
```

Kriging

```
###Kriging Model
###The algorithm was developed by Na-Udom (2007) based on description of
###Welch (1992)

Kriging_theta<-function(Design, y_output) {
  #This algorithm constructs the surrogate model based on Kriging
  #Input arguments are:
  #* Design is generated using latin hypercube design with size nxp
      #where n is number of runs (rows) and p is number of input parameter
  #* y_output is obtained from hydrological model outputs
  #
  #The output of function :
  #*a matrix of optimal theta with size (px1)
  #*computational time

  ##########################################################################
  # nsample - Number of Samples
  # nvar - Number of Variables
  # nrun - Number of Design Runs

  nrun<-dim(Design)[1]
  nvar<-dim(Design)[2]

  #Set the tolerance for the optimize process
  otol<-0.0001

  # tolerance for stopping the loop
  dtol<-0.0001

  # Determine the possible ranges of theta
```

```
lower<-0
upper<-100


#Assign x with design generated by LHD
x<-Design

#Assign y with y_output
y<-y_output

#onematrix<-matrix(1,nrow=nrun,ncol=1)
onematrix<-matrix(1,nrow=nrun,ncol=1)
p<-2    #using Gaussian correlation function

###Optimizing initial theta
###Obtaining theta values by optimizing log likelihood function
###using optimize R function

### The aim of this step is to find loglikehood function (llo)
    ###and theta (thetaMLE)

###A. Function of log likelihood function MLE
### The input is the initial theta values (a matrix of px1)
### The output is a matrix of optimal theta size of px1

MLE<-function(theta){

  theta<-matrix(1,nrow=nvar,ncol=1)

  ###This function is based on Gaussian correlation function
  ###The inputs are:
  ###*design matrix (x)
  ###*theta values
  ###*number of rows (nrun) of design matrix (x)
  ###*number of input parameters (p)--the number of column of
   ###design matrix (x)

  ###The output is correlation matrix size of nrun x nrun
  calR<-function(x,theta,nrun,nvar){
    COV<-matrix(0,nrow=nrun,ncol=nrun)
```

```
    for(i in 1:nrun){
      for(j in 1: nrun){
        sum<-0
        for(k in 1:nvar){
          sum<-sum+theta[k]*(abs(x[i,k]-x[j,k]))^p
        }
        COV[i,j]<- exp(-1*sum)
      }
    }
    R<-COV
  }

  ###compute correlation matrix
  R<-calR(x,theta,nrun,nvar)

  ###Compute betahat according to equation 2.25
  Rinv<-solve(R)
  detR<-det(R)
  RinvY<-solve(R,y)
  Rinvone<-solve(R,onematrix)
  term<-t(onematrix)%*%Rinvone
  betahat<-solve(term,t(onematrix))%*%RinvY

  ###Compute sigmahat according to equation 2.26
  term1<-(y-(onematrix%*%betahat))
  term2<-solve(R,term1)
  sigmahat<-(1/nrun)*t(y-(onematrix%*%betahat))%*%term2

  ###Compute the log likehood function for Gaussian stochastic process
  ###according to equation 2.27
  ans<-(-0.5)*(nrun*log(sigmahat)+log(detR))
}

###Optimize the theta and obtain maximum log likelihood value
###using optimize R function
opt_value<-optimize(MLE,c(lower,upper),tol=otol,maximum=T)

#Extract the values of maximum log likelihood
ll0<-opt_value$objective
```

```
#Extract the values of theta
theta<-matrix(opt_value$maximum,nrow=nvar,ncol=1)


################################################################################

###Second section:

# Assign theta
theta1<-matrix(0,nrow=nvar,ncol=1)
thetaMLE<-theta

###B. Function of log likelihood function MLE1
###This function is aimed for optimizing individual theta (keeping other theta
###that is not focused on being optimized fixed)
###The input is individual theta values
###The output is a matrix of optimal theta size of px1

MLE1<-function(otita1){
  ###Assign a theta at j in theta matrix
  thetaMLE[j]<-otita1
  theta1<-thetaMLE

  ###This function is based on Gaussian correlation function
  ###The inputs are:
  ###*design matrix (x)
  ###*theta values
  ###*number of rows (nrun) of design matrix (x)
  ###*number of input parameters (p)--the number of column of design matrix (x)
  calR1<-function(x,theta1,nrun,nvar){
    COV<-matrix(0,nrow=nrun,ncol=nrun)
    for(i in 1:nrun){
      for(j in 1: nrun){
        sum<-0
        for(k in 1:nvar){
          sum<-sum+theta1[k]*(abs(x[i,k]-x[j,k]))^p
        }
        COV[i,j]<- exp(-1*sum)
      }
    }
```

```
    R<-COV
  }

  ###compute correlation matrix
  R1<-calR1(x,theta1,nrun,nvar)

  ###Compute betahat according to equation 2.25
  R1inv<-solve(R1)
  detR1<-det(R1)
  R1invY<-solve(R1,y)
  R1invone<-solve(R1,onematrix)
  term<-t(onematrix)%*%R1invone
  betahat1<-solve(term,t(onematrix))%*%R1invY

  ###Compute sigmahat according to equation 2.26
  term1<-(y-(onematrix%*%betahat1))
  term2<-solve(R1,term1)
  sigmahat1<-(1/nrun)*t(y-(onematrix%*%betahat1))%*%term2

  ###Compute the log likehood function for Gaussian stochastic process
  ###according to equation 2.27
  ans<-(-0.5)*(nrun*log(sigmahat1)+log(detR1))
}

###Initialize matrix MAXI to store which theta produce the largest
###differences of loglikelihood function and remove
###it for the next iteration.
MAXI<-matrix(0,nrow=nvar,ncol=1)

# Algorithm as described in Welch et al. (1992) Step 2 to Step 7
for(i in 1:nvar){

  ###Initialize the maxdiff variable
  maxdiff<-0

  ###Check whether j has been stored in MAXI. j which is optimized will not
  ###be optimized for the next iteration
  for(j in 1:nvar){
    check<-1
    for(k in 1:i){
```

```
  if(j==MAXI[k]){
    check<-0
  }
}

###Compute the initial differences between one obtain from first section
###and current section
if(check==1){
  ###Optimize theta to maximize log likelihood values
  opt_value_1<-optimize(MLE1,c(lower,upper),tol=otol,maximum=T)

  ###Assign loglikelihood and theta


  #Assign number of iteration at j
  iterJ=1

  if(j==1){
    ###Extract the values of maximum log likelihood
    ll1<-opt_value$objective

    #Compute the difference
    maxdiff<-ll1-llo

    ###Assign current j at maxvar
    maxvar<-j

    ###Extract the values of theta
    theta_temp<-opt_value$maximum

    #Increase iterJ by 1
    iterJ<-iterJ+1
  }
  else
  {
    ###Extract the values of maximum log likelihood
    ll1<-opt_value$objective

    ###New difference
    diff1<-ll1-ll0
```

```
        if(diff1>maxdiff){

          ###Assign maxdiff with bigger difference
          maxdiff<-diff1

          ###Assign current j at maxvar
          maxvar<-j

          ###Extract the values of theta
          theta_temp<-opt_value$maximum

          #Increase iterJ by 1
          iterJ<-iterJ+1
        }
      }
    }
  }

  if (maxdiff>dtol){
    ###Assign j at MAXI that provides the maximum difference
    MAXI[i]<-maxvar

    ###Assign llo with current log likelihood values
    llo<-ll1

    ###Store theta_temp in thetaMLE
    thetaMLE[maxvar]<-theta_temp
  } else {
    break
  }

}

###Compute computational time
comp_time<-proc.time() - ptm

##Create the ouput storage
output<-list()
```

```
  output[[1]]<-thetaMLE
  output[[2]]<-comp_time

}
```