Developing Efficient Strategies for Automatic Calibration of Computationally Intensive Environmental Models

by

Seyed Saman Razavi

A thesis

presented to the University of Waterloo

in fulfillment of the

thesis requirement for the degree of

Doctor of Philosophy

in

Civil Engineering

Waterloo, Ontario, Canada, 2013

AUTHOR'S DECLARATION

Abstract

Environmental simulation models have been playing a key role in civil and environmental engineering decision making processes for decades. The utility of an environmental model depends on how well the model is structured and calibrated. Model calibration is typically in an automated form where the simulation model is linked to a search mechanism (e.g., an optimization algorithm) such that the search mechanism iteratively generates *many* parameter sets (e.g., thousands of parameter sets) and evaluates them through running the model in an attempt to minimize differences between observed data and corresponding model outputs. The challenge rises when the environmental model is computationally intensive to run (with run-times of minutes to hours, for example) as then any automatic calibration attempt would impose a large computational burden. Such a challenge may make the model users accept sub-optimal solutions and not achieve the best model performance.

The objective of this thesis is to develop innovative strategies to circumvent the computational burden associated with automatic calibration of computationally intensive environmental models. The first main contribution of this thesis is developing a strategy called "deterministic model preemption" which opportunistically evades unnecessary model evaluations in the course of a calibration experiment and can save a significant portion of the computational budget (even as much as 90% in some cases). Model preemption monitors the intermediate simulation results while the model is running and terminates (i.e., pre-empts) the simulation early if it recognizes that further running the model would not guide the search mechanism. This strategy is applicable to a range of automatic calibration algorithms (i.e., search mechanisms) and is *deterministic* in that it leads to exactly the same calibration results as when preemption is not applied.

One other main contribution of this thesis is developing and utilizing the concept of "surrogate data" which is basically a reasonably small but representative proportion of a full set of calibration data. This concept is inspired by the existing surrogate modelling strategies where a surrogate model (also called a metamodel) is developed and utilized as a fast-to-run substitute of an original computationally intensive model. A framework is developed to efficiently calibrate hydrologic models to the full set of calibration data while running the original model only on surrogate data for the majority of candidate parameter sets, a strategy which leads to considerable computational saving. To this end, mapping relationships are developed to approximate the model performance on the full data based on the model performance on surrogate data. This framework can be applicable to the calibration of any environmental model where appropriate surrogate data and mapping relationships can be identified.

As another main contribution, this thesis critically reviews and evaluates the large body of literature on surrogate modelling strategies from various disciplines as they are the most commonly used methods to relieve the computational burden associated with computationally intensive simulation models. To reliably evaluate these strategies, a comparative assessment and benchmarking framework is developed which presents a clear computational budget dependent definition for the success/failure of surrogate modelling strategies. Two large families of surrogate modelling strategies are critically scrutinized and evaluated: “response surface surrogate” modelling which involves statistical or data–driven function approximation techniques (e.g., kriging, radial basis functions, and neural networks) and “lower-fidelity physically-based surrogate” modelling strategies which develop and utilize simplified models of the original system (e.g., a groundwater model with a coarse mesh). This thesis raises fundamental concerns about response surface surrogate modelling and demonstrates that, although they might be less efficient, lower-fidelity physically-based surrogates are generally more reliable as they to-some-extent preserve the physics involved in the original model.

Five different surface water and groundwater models are used across this thesis to test the performance of the developed strategies and elaborate the discussions. However, the strategies developed are typically simulation-model-independent and can be applied to the calibration of *any* computationally intensive simulation model that has the required characteristics. This thesis leaves the reader with a suite of strategies for efficient calibration of computationally intensive environmental models while providing some guidance on how to select, implement, and evaluate the appropriate strategy for a given environmental model calibration problem.

Acknowledgements

I would like to thank Dr. Bryan Tolson, my PhD co-advisor, for his scientific supports and stimulating discussions throughout my PhD studies. Also, I have always been grateful to Bryan for his care and advice on things other than research, especially for his care in helping me and my wife smoothly settle down in Waterloo. I would also like to thank Dr. Donald Burn, my PhD co-advisor, for his guidance and comments on my research.

Nasim, my wife and constant companion, showed endless love and patience during my PhD studies, for which I am forever grateful. She sacrificed many weekends and evenings to the cause and did everything she could to make home a warm and lovely place to be.

It is my pleasure to thank my PhD committee members Dr. François Anctil from Université Laval, Dr. Gordon Savage, Dr. Neil Thomson, and Dr. James Craig from the University of Waterloo, and Dr. Shawn Matott from the State University of New York at Buffalo. I would also like to thank the help and comments of Paul Martin, Gord MacMillan, and Doug Anderson on my research during my internship with Matrix Solutions Inc.

The majority of the thesis contents were peer-reviewed in the form of technical journal papers before thesis submission. I would like to thank Dmitri Kavetski, Elena Toth, and one anonymous reviewer who reviewed the paper associated with Chapter 2, Holger Maier, Uwe Ehret, and the two anonymous reviewers who reviewed the paper associated with Chapter 3, and Marco Ratto and the four anonymous reviewers who reviewed the paper associated with Chapter 4. Their constructive comments significantly improved the contents of my thesis.

During my PhD studies, I was partially supported by Ontario Graduate Scholarship (OGS), MITACS Accelerate program, Queen Elizabeth II Graduate Scholarship in Science and Technology (QEIIGSST), Dr. T.E. Unny Memorial Award, nine University of Waterloo Graduate Merit Scholarships, and University of Waterloo president's Graduate Scholarship, which are hereby acknowledged. I would also thank the generous support of Matrix Solutions Inc. I also acknowledge DHI-WASY for providing me with a research license for the FEFLOW groundwater modelling software.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Problem Statement

Advanced technologies have provided detailed spatiotemporal information about a variety of environmental variables and natural processes. Environmental model developers tend to make use of all available data and rigorously model all the detailed processes in their system of interest. Such practice may lead to highly complex models that demand huge computational budgets to simulate the system. For example, a hydrologic or groundwater model representing a detailed conceptualization with a very fine level of discretization may demand hours (or even days) for a single run [*Keating et al.*, 2010; *Mugunthan et al.,* 2005; *Zhang et al.,* 2009]; while many model runs (e.g., thousands or more) are required when calibrating the model, analyzing different uncertainties involved or studying the best management practices. As such, despite the existence of rapidly advancing computing facilities, the computational burden has remained a challenge for many modelling applications in water resources and environmental engineering, especially for automatic model calibration. This thesis develops methodologies aiming to circumvent such computational burdens imposed when calibrating computationally intensive environmental models.

Environmental model calibration may be of two general types: optimization-based calibration and uncertainty-based calibration. Optimization-based calibration refers to the coupling of an environmental model with an optimization engine such that the optimization engine adjusts model parameters in an attempt to minimize differences between observed data and corresponding model outputs (i.e., simulated equivalents). Examples of optimization-based calibration tools include PEST [*Doherty*, 2005], UCODE [*Poeter and Hill*, 1998] and OSTRICH [*Matott*, 2005]. Uncertainty-based calibration refers to the coupling of an environmental model with an uncertainty engine such that the uncertainty engine repeatedly samples model parameter configurations to develop a calibrated probability distribution for the parameters. Unlike optimization-based calibration, which is focused on identifying a single “optimal” parameter set, uncertainty-based calibration identifies numerous “plausible” parameter sets. Examples of tools for uncertainty-based calibration include GLUE [*Beven and Binley*, 1992], Sequential Uncertainty Fitting (SUFI-2 – [*Abbaspour et al.*, 2004]) and various Markov Chain Monte Carlo (MCMC) implementations [*Kuczera and Parent*, 1998].

What is Automatic Calibration?

An environmental model (such as a rainfall-runoff model) is a simplified mathematical representation of a complex real-world system. Environmental models when developed attempt to emulate the real-world systems of interest as closely as possible so that they enable the users to predict the response of the systems under the future/unseen conditions. An environmental model can be mathematically represented as:

$$\hat{\boldsymbol{y}} = f(\boldsymbol{x}, \boldsymbol{p}) \tag{1-1}$$

where f represents the model as a function, $\hat{\boldsymbol{y}}$ is the vector of model responses (outputs) over spatial and/or temporal domains, $\boldsymbol{x}$ is the vector of model forcing data (inputs), and $\boldsymbol{p}$ is the vector of model parameters. For example in a rainfall-run model, $\hat{\boldsymbol{y}}$ can be the vector of simulated daily streamflows at a watershed outlet, $\boldsymbol{x}$ can consist of daily rainfall, temperature, and snow pack data, and $\boldsymbol{p}$ may include the average watershed slope, travel time, and different watershed storage parameters. As an environmental model is never a perfect emulator of the complex real-world system of interest, the vector of real system responses, $\boldsymbol{y},$ is typically represented as:

$$\boldsymbol{y} = f(\boldsymbol{x}, \boldsymbol{p}) + \boldsymbol{e} \tag{1-2}$$

where $\boldsymbol{e}$ is the vector of model errors. Given an environmental model is adequately structured, the utility of the model depends on how well the model parameters are "calibrated". In the process of the so-called "model calibration", the model parameters are adjusted to minimize the discrepancies between the model outputs, $\hat{\boldsymbol{y}}$, and the real-world system responses, $\boldsymbol{y}$. This process is also called "model inversion" or "parameter estimation".

There are typically two types of model parameters: physical parameters and process parameters. Physical parameters directly represent physical properties of the system and can be measurable, while process parameters are typically conceptual or empirical and cannot be measured in the field [*Pechlivanidis et al.*, 2011]. For example in the case of rainfall-runoff models, catchment area and average slope are physical parameters, while parameters controlling baseflow recession curve are process parameters. Process parameters almost always need to be calibrated against the observed data. Moreover, the modellers may opt to calibrate/fine-tune some physical parameters such as average catchment slope as they are measured with some level of uncertainty. There are also some physical parameters that although are measurable are often calibrated. Examples of such parameters include hydraulic conductivity and porosity in groundwater models; the reason is only point measurements of these aquifer parameters can be available, while in a groundwater model such parameters represent the characteristics of a large body of soil (average characteristics) possibly very different from the point measurements.

Early efforts to calibrate environmental models, before the wide spread of powerful computing facilities, were based on a "manual" approach. "Manual calibration" relies heavily on the modeller's understanding of the simulated processes, model structure, and parameters, and is generally a trial-and-error procedure. The applicability of manual calibration is typically *limited* to models with a limited number of parameters. Manual calibration becomes very difficult when calibrating interacting parameters, which commonly exist in environmental models [*Gupta et al.*, 1999]. In addition, there is a great deal of subjectivity involved and different results may be obtained by different modellers attempting to manually calibrate the same model to the same data [*Pechlivanidis et al.*, 2011]. Because of the time-consuming and difficult nature of manual calibration, researchers in the 1960s and early 1970s started investigating more objective and automated approaches to model calibration [*Gupta et al.*, 1999]. These research efforts have led to the development of a variety of methods for "automatic calibration".

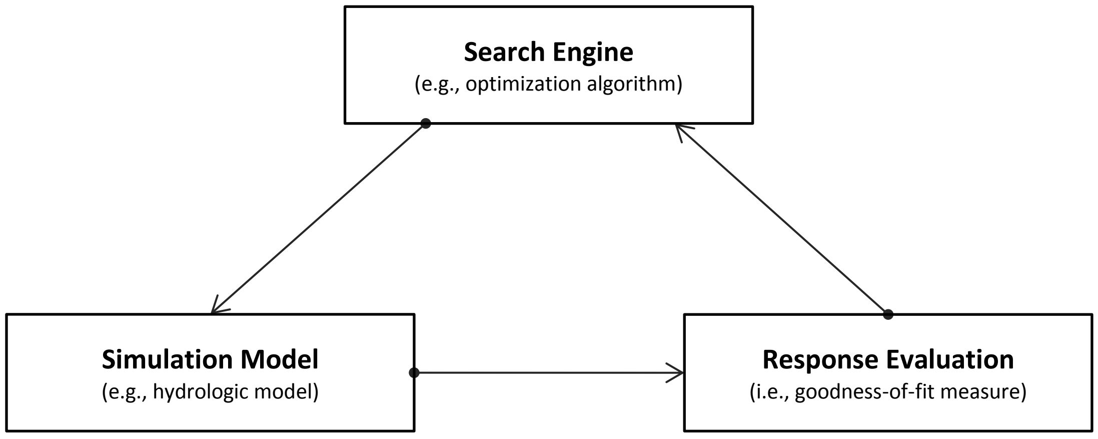


Figure 1-1. The three main components involved in automatic calibration of environmental models

Any automatic calibration procedure consists of the three main components shown in Figure 1-1. The search engine repeatedly generates candidate parameter sets, $\boldsymbol{p}$, within a hyper-cube of feasible parameter space, the simulation model runs with this parameter set and produces the model responses, $\hat{\boldsymbol{y}}$, and then the third component (i.e., typically a function of modelling errors, $g(\boldsymbol{e})$) objectively quantifies the goodness-of-fit between model responses and observed data, $\boldsymbol{y}$. The automatic calibration problem is most often posed in the form of an optimization formulation as:

$$\underset{\boldsymbol{p}}{\text{minimize}}\ g(\boldsymbol{e})$$

$$\text{S.T: } p_i^{\text{lb}} \le p_i \le p_i^{\text{ub}} \quad \text{for } i = 1, \dots, n \tag{1-3}$$

where n is the number of parameters to be calibrated, p_i^{lb} and p_i^{ub} are the lower and upper bound constraints on parameter i, and $g(\boldsymbol{e})$ represents the objective function (i.e., error function) to be minimized. The form of the appropriate objective functions relates to the type of the environmental model to be calibrated and the modelling objectives. Examples of objective functions in automatic calibration include weighted sum of squared errors (WSSE) and root mean squared errors (RMSE). Some goodness-of-fit functions take larger values for better fits, such as Nash-Sutcliffe coefficient of efficiency [*Nash and Sutcliffe*, 1970b] and Kling-Gupta efficiency [*Gupta et al.*, 2009], and as such require to transform the minimization problem in Equation (1-3) into a maximization problem. A variety of optimization algorithms have been developed or used in the literature as the search engine for automatic calibration of environmental models such as Levenberg-Marquardt [*Doherty*, 2005], shuffled complex evolution [*Duan et al.*, 1993], genetic algorithms [*Ndiritu and Daniell*, 2001], and dynamically dimensioned search [*Tolson and Shoemaker*, 2007]. Automatic calibration procedures solving the above optimization formulation typically lead to a single parameter set that best represents the real-world system in terms of the selected goodness-of-fit criterion. Such a practice is referred to as "optimization-based calibration" in this thesis.

There is another family of automatic calibration procedures that search and collect a large number of high-quality parameter sets, instead of searching for a single optimal parameter set, with the objective of developing probability distributions for the model response and/or parameters. Such procedures attempt to quantify the uncertainties in model parameters and predictions, and as such are referred to as "uncertainty-based calibration" in this thesis. The basic idea behind this type of model calibration is that there are always many different parameter sets (or models) that can emulate the real-world system (almost) equally well; these equally behavioral (acceptable) parameters/models are therefore called equifinal [*Beven and Binley*, 1992]. In uncertainty-based calibration, a statistical (formal or informal) "likelihood function" is used as the goodness-of-fit measure which quantifies how likely a candidate parameter set (or a model) is the *true* parameter set (or the true model) given the observed data. A variety of methods have been developed in the literature for uncertainty-based calibration of environmental models including generalized likelihood uncertainty estimation [GLUE - *Beven and Binley*, 1992], sequential uncertainty fitting [SUFI-2 - *Abbaspour et al.*, 2004], null-space Monte Carlo [*Tonkin and Doherty*, 2009], and various Markov-chain Monte Carlo (MCMC) implementations [*Kuczera and Parent*, 1998; *Vrugt et al*., 2009].

Automatic calibration can take the form of a multi-objective (multi-criteria) optimization problem [*Efstratiadis and Koutsoyiannis*, 2010; *Madsen et al.*, 2002; *Yapo et al.*, 1998]. The motivation for involving multiple criteria in the automatic calibration process is that the single-criterion calibration results are typically biased to the individual aspects of the model response emphasized by the goodness-of-fit criterion used. Examples of possibly conflicting goodness-of-fit criteria in rainfall-runoff modelling include RMSE of peak flow events and RMSE of low flow events. Multi-objective automatic calibration results in a suite of parameter sets that approximate the trade-off relationship between the various calibration objectives. However, *Kollat et al.* [2012] have recently found that meaningful multi-objective trade-offs in watershed model calibration are less frequent than the literature has suggested.

1.2 Overview of Solution Approaches

Figure 1-2 identifies four main solution approaches for alleviating the computational burden associated with calibrating computationally intensive models. There are a variety of methods and strategies in the literature that fall under each of these main approaches. Notably, these approaches are largely complementary and in practice a given calibration exercise may employ a combination of approaches to maximize the calibration computational efficiency (e.g., using surrogate modelling in concert with a parallelized and computationally efficient optimization algorithm). The focus of this thesis is on the first two approaches (i.e., preemption strategies and surrogate modelling) as outlined in Section 1.3, and therefore, significant details on these can be found throughout this thesis. The other two approaches (i.e., parallel computing and more efficient calibration algorithms) are introduced only in this section, and their details are beyond the thesis scope and can be found in the references given. The methods developed in this thesis can be extended such that they can be utilized on parallel computing resources possibly in conjunction with more efficient calibration algorithms.

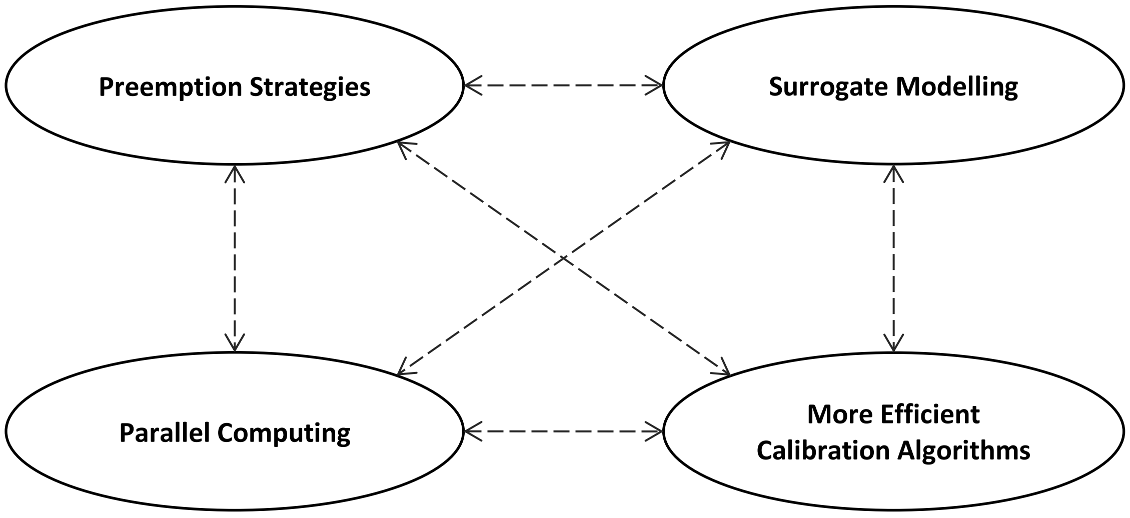


Figure 1-2. Four general solution approaches to calibration of computationally intensive simulation models – various combinations of these four approaches are theoretically possible.

Strategies for opportunistically avoiding model evaluations in the course of simulation-optimization attempts are called “preemption strategies” in this thesis. Relatively few strategies have been proposed in the literature to intelligently avoid unnecessary model evaluations in model calibration (i.e., model evaluations that yield implausible, non-behavioral or non-informative simulation results). Conversely, surrogate modelling (also called metamodelling) is an increasingly common approach to dealing with computationally expensive simulation models and is concerned with developing and utilizing cheaper "surrogates" of expensive simulation models to improve the overall computational efficiency. These surrogates replace original computationally intensive models in calibration experiments. There is a rich literature on different surrogate modelling methods arising from a variety of disciplines. In general, surrogate models may be in the form of statistical or data-driven models such as kriging and neural networks aiming to emulate the response surface of computationally intensive models, or they can be simpler, faster-to-run models of the original environmental system of interest; e.g., a finite difference-based groundwater model with a coarse mesh may be deemed a surrogate of a similar model with a fine mesh.

Different strategies have been developed in the literature to divide computationally intensive problems into multiple sub-problems that can be run concurrently on parallel computing resources. In the model calibration context, depending on the calibration algorithm used, groups of model evaluations can be typically run in parallel to save calibration time. Optimization-based calibration using parallel search algorithms such as parallel SCE-UA [*Feyen et al.*, 2007; *Vrugt et al.*, 2006a], parallel GA [*Cheng et al.*, 2005; *He and Hui*, 2007], and parallel PSO [*Matott and Rabideau*, 2008; *Schutte et al.*, 2004] can result in considerable time savings when compared with the corresponding serial algorithm implementations. Supercomputer networks can also vastly improve the efficiency of uncertainty-based calibration techniques, such as GLUE. For example, *Brazier et al.* [2000] and *Freer et al.* [2004] each conduct more than 3 million model evaluations for GLUE on parallel computing networks.

A significant amount of research has been directed at developing highly efficient algorithms that are suitable for calibrating computationally expensive models. These algorithms are designed to generate optimal or near-optimal solutions through a limited number of model evaluations. Examples of highly efficient algorithms for optimization-based calibration include: a hybrid tabu search – adjoint state method [*Tan et al.*, 2008]; dynamically dimensioned search (DDS - [*Tolson and Shoemaker*, 2007b]); a tuned particle swarm optimizer (PSO - [*Beielstein et al.*, 2002]); stepwise linear search (SLS - [*Kuzmin et al.*, 2008]); and gradient-based methods [*Doherty*, 2005; *Ha et al.*, 2007], if they are applied in conjunction with smoothing strategies that ensure a "well-behaved" objective function [*Kavetski and Kuczera*, 2007]. Examples of efficient algorithms for uncertainty-based calibration include: DDS-approximation of uncertainty (DDS-AU - [*Tolson and Shoemaker*, 2008]), automatic calibration and uncertainty assessment using response surfaces (ACUARS - [*Mugunthan and Shoemaker*, 2006]), and limited-memory MCMC [*Kuczera et al*., 2010].

1.3 Research Contributions and Thesis Structure

The objective of this thesis is to develop and formalize new efficiency-increasing strategies for automatic calibration of computationally intensive environmental models. This thesis is structured around published and submitted articles. Chapters 2 through 4 correspond to three published articles, and Chapter 5 corresponds to an article which is currently under review. Chapter 6 ends this thesis with research summary, conclusions, and future directions. In the following, the main contributions of this thesis along with the associated chapters are outlined.

1- Develop and formalize the "deterministic model preemption" strategy for opportunistically evading unnecessary model evaluations (see Chapter 2). This strategy, which is applicable to a range of optimization-based and uncertainty-based algorithms, can save a significant portion of the computational budget in model calibration practice while being "deterministic" in that it leads to exactly the same calibration results as when preemption is not applied. Notably, the application of this strategy is not limited to model calibration, and we have shown its promise in two water resources systems optimization studies [i.e., *Razavi et at.*, 2013; *Asadzedeh et al*., 2013; not included in the thesis].

2- Review, analyze, and standardize the research efforts and publications on surrogate modelling arising from a variety of disciplines ranging from Mathematics and Computer Science to Water Resources Engineering (see Chapter 3). Surrogate modelling (also called metamodelling) is the most commonly used approach to circumventing the computational burden associated with computationally intensive simulation models. This thesis attempts to address the lack of organization, referencing, and consistencies observed in the literature on surrogate modelling especially in the environmental and water resources community. Taxonomies on surrogate modelling frameworks, practical details, advances, challenges, and limitations are outlined. Two broad families of surrogate modelling strategies namely response surface surrogates and lower-fidelity physically-based surrogates are recognized and scrutinized. Multiple applications of surrogate modelling in optimization-based and uncertainty-based calibration of computationally intensive environmental models are discussed.

3- Develop a comparative assessment framework which presents a clear computational budget dependent definition for the success/failure of surrogate modelling strategies (see Chapter 4). In the literature, evaluation of the algorithms enabled with surrogate models is typically on an ad-hoc basis and sometimes unreliable. The developed assessment framework, which emphasizes appropriate benchmarking and the dependency on the computational budget, enables the user to comprehensively evaluate any calibration algorithm (not only the ones using surrogate modelling) and gives reliable insights into the algorithm performance under different conditions.

4- Critically evaluate response surface surrogate strategies, through numerical experiments, against other common optimization (calibration) strategies not involving surrogate models (see Chapter 4). This assessment challenges the capabilities of response surface surrogates and, in contrast to the common belief in the literature, demonstrates that response surface surrogate modelling is not always an efficient and reliable approach to optimizing computationally intensive problems. Moreover, this research demonstrates that neural networks, although quite common in surrogate modelling, are not appropriate tools for this purpose and can be quite misleading when the computational budget is limited.

5- Develop the concept of “surrogate data” and a framework utilizing this concept for efficient calibration of computationally intensive hydrologic models (see Chapter 5). A key component of this contribution is a mapping system that maps the model performance on surrogate data to the model performance on “full calibration data”. The developed calibration framework is capable of calibrating the model to full calibration data within very limited computational budgets.

In addition, a significant part of my research efforts during PhD was devoted to develop a general purpose, more efficient and transparent neural network model (called Reformulated Neural Network, ReNN) to be potentially used in surrogate modelling. However, as this thesis eventually concludes that neural networks may not be appropriate tools for surrogate modelling (see #4 above), this contribution is not included in the thesis document for the sake of consistency. Full detail of ReNN is published in *Razavi and Tolson* [2011].

Note that the methodologies developed throughout this thesis are typically simulation-model-independent and can be applied to calibration of *any* computationally intensive simulation model that has the required characteristics. A total of five different surface water and groundwater models with seven to 62 calibration parameters are used across this thesis to test the performance of the developed methodologies. Three of these models have been developed by other researchers as referenced in the relevant sections, and only their application with the developed methodologies is within the scope of this thesis.

Chapter 2
Model Preemption: A Simple Strategy for Opportunistically Evading Unnecessary Model Evaluations

This chapter is a mirror of the following published article with minor changes to increase its consistency with the body of the thesis. Changes were only made in the Summary (abstract), Section 2.1, and Section 2.2 and intended to delete the contents that have been presented more appropriately in other parts of the thesis. References are unified at the end of the thesis.

Razavi, S., B. A. Tolson, L. S. Matott, N. R. Thomson, A. MacLean, and F. R. Seglenieks (2010), Reducing the computational cost of automatic calibration through model preemption, *Water Resour*. *Res*., 46, W11523, 17 pages, doi:10.1029/2009WR008957.

Summary

This chapter introduces and formalizes the concept of simulation model preemption during automatic calibration. The proposed model preemption method terminates a simulation model early to save computational budget if it is recognized through intermediate simulation model results that a given solution (model parameter set) is so poor that it will not benefit the search strategy. The methodology proposed here is referred to as deterministic model preemption because it leads to exactly the same calibration result as when deterministic preemption is not applied. As such, deterministic preemption enabled calibration algorithms which make no approximations to the mathematical simulation model are a simple alternative to the increasingly common and more complex approach of surrogate modelling (see Chapter 3) for computationally constrained model calibration. Despite its simplicity, the deterministic model preemption concept is a promising concept that has yet to be formalized in the environmental simulation model automatic calibration literature. The model preemption concept can be applied to a subset of uncertainty-based and optimization-based automatic calibration strategies using a variety of different objective functions. Results across multiple calibration case studies demonstrate actual preemption computational savings ranging from 14-49%, 34-59% and 52-96% for the dynamically dimensioned search, particle swarm optimization, and GLUE automatic calibration methods, respectively.

2.1 Introduction and Objective

This chapter introduces the "model preemption" strategy to alleviate the computational burden of calibrating computationally expensive environmental models and explores, through numerical experiments, the potential advantages of performing model calibration with this strategy that allows for expensive simulation models to be terminated early. This strategy is general in that it can be adapted to a wide variety of existing calibration algorithms (i.e., optimizers or samplers). To demonstrate and evaluate the model preemption strategy, it was linked with four separate model calibration algorithms: Dynamically Dimensioned Search (DDS - [*Tolson and Shoemaker*, 2007b]), particle swarm optimization (PSO - [*Kennedy and Eberhart*, 1995]), GLUE ([*Beven and Binley*, 1992]), and DDS-approximation of uncertainty (DDS-AU - [*Tolson and Shoemaker*, 2008]). While DDS and PSO are suitable for optimization-based calibration, GLUE and DDS-UA are geared towards uncertainty-based calibration. The various preemption enabled algorithms were benchmarked using a suite of calibration case studies involving so-called "black-box" environmental models. Case studies include: (1) a hydrologic model of the Cannonsville Watershed, New York State; (2) a hydrologic model of the Reynolds Creek Watershed, Idaho; and (3) a model of subsurface contaminant transport to support a dipole flow and reactive tracer test in the well-known Borden aquifer in Ontario, Canada.

The structure of this chapter is organized as follows: Section 2.2 outlines related research in the literature; Sections 2.3-2.5 describe the preemption methodology and its applicability in optimization-based and uncertainty-based calibration frameworks. Sections 2.6 and 2.7 summarize case studies that facilitated demonstration and benchmarking of the new method and experimental settings. Preemption results, as applied to the various case studies, are presented in Section 2.8, and Sections 2.9 and 2.10 contain a discussion of the results and concluding remarks, respectively.

2.2 Related Research

Relatively few methods have been proposed to intelligently avoid unnecessary model evaluations (i.e., model evaluations that yield implausible, non-behavioral or non-informative results). As suggested by *Haupt and Haup* [1998] and *Griffin et al*. [2008], a simple way to avoid repeating expensive model evaluations is to store and consult a running history of every parameter set (and corresponding objective function value) that is evaluated. Known as "caching", this strategy has been implemented in freely available optimization codes [*Gray and Kolda*, 2006].

Like caching, model preemption is a general purpose mechanism for opportunistically avoiding model evaluation. The basic concept behind model preemption is that full evaluation of a given candidate solution is unwarranted if the corresponding objective function is predictably poor (i.e., are implausible, non-behavioral or non-informative) relative to previous solutions or some fixed threshold. The key to model preemption is being able to predict poor performance prior to fully evaluating each candidate solution (i.e., prior to completely simulating the entire model simulation time period or spatial domain).

Different variations of model preemption in an optimization framework have been proposed. For example, *Joslin et al.* [2006] describe an optimization framework in which constraint violations are determined through a computationally expensive model while the cost function is inexpensive and evaluated independently. Under this type of framework, the authors demonstrated significant computational savings by first evaluating the cost function and then selectively evaluating the constraints only if some threshold cost criteria was satisfied. In an automatic calibration framework, *Ostfeld and Salomons* [2005] proposed constructing a "hurdle" for accepting/rejecting a candidate set of parameters during a model simulation – where a hurdle corresponds to a predefined threshold objective function value assigned to a specific simulation breakpoint. At a given hurdle, the simulation is aborted if the objective function value of the candidate solution does not exceed the value of the hurdle.

SwarmOps [*Pedersen*, 2008], a recently developed software tool for automating the process of tuning the parameters of a given optimization algorithm, also introduces a type of preemption concept namely "pre-emptive fitness evaluation" to increase optimization efficiency. The algorithm tuning process in SwarmOps involves solving many different optimization problems multiple times each in order to assess average algorithm behavior under all candidate algorithm parameter sets. In this regard, the SwarmOps tool utilizes what we refer to as "trial-based" preemption to avoid unnecessary evaluation of entire optimization trials. Unlike trial-based preemption, the present work employs preemption at a much finer level through a "model-based" approach.

A general preemption strategy has undoubtedly been applied by many modellers performing manual (trial and error) calibration on computationally intensive simulation models. For example, modelers often frequently check intermediate simulation results and will terminate poor simulations before the end of the simulation period. In an optimization-based or uncertainty-based calibration context, the preemption concept has not been formalized and although there are similarities to previous research, the concept as applied to model calibration is a unique contribution of this paper.

2.3 Model Preemption Concept

In traditional optimization-based and uncertainty-based simulation model calibration frameworks the quality of a candidate solution (i.e., some model parameter set configuration) is quantified using an objective (or likelihood) function that measures model prediction errors calculated over the entire model simulation time period. In transient or continuous hydrologic models, however, prediction errors accumulate throughout the simulation time period. Making use of this fact, we propose an approach that monitors the simulation model performance during the simulation time period. If it is recognized through intermediate simulation model results that a given solution (model parameter set) is so poor that it will not contribute to guiding the search strategy, the simulation model is terminated early (i.e., preempted) to save computational budget. In this paper, we introduce a simple form of preemption that we call deterministic model preemption and apply this preemption technique in the context of calibrating environmental simulation models. Deterministic model preemption refers to the termination of model simulations that have demonstrated such poor performance that the solution will definitely not contribute to guiding the search strategy. In other words, the application of the deterministic preemption strategy leads to exactly the same calibration result as when deterministic preemption is not applied. This attractive property of deterministic preemption does not hold for meta-modeling strategies, or the hurdle approach in *Ostfeld and Salomons* [2005] as discussed in Section 2.2.

In the hurdle approach to automatic calibration in *Ostfeld and Salomons* [2005], hurdle magnitudes and breakpoints over the simulation period, referred to as parameters of the methodology, are assigned arbitrarily prior to the optimization. No objective procedure is suggested for specifying these parameters. In contrast, our preemption threshold is determined objectively and adaptively as explained later in Section 2.3.2. Moreover, as the hurdle has a predefined value regardless of the objective function value of the current best solution in the course of an optimization trial, there is no guarantee that an aborted solution is non-informative. In other words, in the hurdle approach, if users select a strict hurdle, they might increase computational saving at the expense of frequently aborting informative solutions and thus change algorithm behaviour and potentially the final calibration solution.

2.3.1 Applicable Objective Functions

Model calibration may be viewed as the process of adjusting or sampling model parameters within prescribed limits (i.e., parameter ranges) to obtain one or more model parameter sets that satisfy some criteria. The main criterion in this context is the deviation of model outputs from measured data. This deviation is usually formulated as either an objective function to be minimized (i.e., in optimization-based calibration) or as a likelihood function to be sampled (i.e., in uncertainty-based calibration). Many calibration objective functions have been proposed in the literature, but not all of them are applicable to preemption enabled model calibration.

The essential characteristic of any applicable objective function within the model preemption framework is that it must monotonically degrade in quality with simulation period length. In other words, the objective function must not improve as more simulation time steps are considered. Objective functions that can be derived from (or are transformations of) any monotonic function are indirectly applicable as well.

For this study, we selected the sum of squared errors (SSE) and its weighted version as a representative objective function suitable for investigating preemption enabled calibration. SSE accumulates the squared errors between simulation results and measurements from the beginning of calibration period up to the current simulation time step, *t*, as:

$$\mathrm{SSE}_{\mathrm{t}} = \sum_{\mathrm{i}=1}^{\mathrm{t}} (\mathrm{sim}_{\mathrm{i}} - \mathrm{obs}_{\mathrm{i}})^2 \qquad \forall t = 1,\ldots,T \tag{2-1}$$

where *T* is the length of the calibration period, sim_i and obs_i are the simulated and observed values at time step i, respectively.

Other commonly used performance metrics in water resources modeling include: the mean squared error (MSE), root mean squared error (RMSE), RMSE of peak flow events, RMSE of low flow events, and their weighted versions, as well as the Nash-Sutcliffe coefficient of efficiency [*Nash and Sutcliffe*, 1970a]. These metrics are all transformations of the SSE objective and are therefore also applicable within the preemption framework. Alternative metrics to SSE that monotonically degrade in quality such as the sum of absolute deviations are also applicable.

An example of using transformations of SSE is demonstrated when the Nash–Sutcliffe (E_{NS}) coefficient is selected as the calibration objective function. An intermediate Nash-Sutcliffe value, E'_{NS}, can be calculated as a transformed and normalized *SSE* measure:

$$E'_{NS} = 1 - \frac{\sum_{t=1}^{\tau}(obs_t - sim_t)^2}{\sum_{t=1}^{T}(obs_t - \overline{obs})^2} = 1 - \frac{SSE_\tau}{\sum_{t=1}^{T}(obs_t - \overline{obs})^2} \tag{2-2}$$

where obs_t and sim_t are the measured and simulated data, $\overline{obs}$ is the average of measured data with the length of *T*, τ is the current time step in the course of a model simulation ($\tau = 1,2,...,T$) and *t* is a timing index. Since the denominator of the fraction is only a function of observed data, it is a constant value and can be calculated over the entire calibration period prior to the calibration process. As such, E'_{NS} can be viewed as a linear function of *SSE* resulting in a monotonically decreasing function. At the end of a given simulation, the current time, τ, will equal the overall simulation time *T* ($\tau = T$) and the intermediate Nash-Sutcliffe coefficient, E'_{NS}, will be equal to the overall Nash-Sutcliffe coefficient, E_{NS}, as desired.

2.3.2 Preemption Threshold

As a general concept, and for a given calibration algorithm, the deterministic model preemption threshold defines a specific objective function value that separates model parameter sets that are known to have no influence on calibration algorithm behaviour (and thus have no impact on the calibration result) from those that are known to influence algorithm behaviour. As such, there are some calibration algorithms which are not suitable for model preemption (as discussed in Section 2.4). In this study, since SSE-based objective functions are utilized, our model preemption threshold is the maximum SSE value a solution can have before it is known to have no impact on the model calibration result. If, during the course of a model simulation, the accumulated (but intermediate) SSE exceeds the preemption threshold, then there is no doubt that continuing evaluation of the current model parameter set is unnecessary because the objective function in question will not change the behavior of the overlying calibration algorithm.

In optimization-based calibration, a given preemption enabled optimizer may involve one or more preemption thresholds depending on whether the algorithm operates on a single candidate solution or a population of candidate solutions. Importantly, a given preemption threshold may be dynamically adjusted as a given optimization algorithm progresses. Dynamic adjustments to the preemption threshold(s) generally coincide with the updates to the best solution(s) found so far in a given search. Accordingly, initial preemption threshold value(s) can be assigned using the initial solution of a given optimization trial.

2.3.3 Illustrative Example of Model Preemption

Figure 2-1 illustrates the model preemption concept and the use of a preemption threshold. The figure plots a representative monotonic increase of the sum-of-squared errors objective function as a function of elapsed simulation time steps. Once the simulation reaches time t_p of the T total time steps, the SSE value already exceeds a particular preemption threshold and at that point the simulation could be aborted to save computational budget. When calibrating continuous hydrologic models, modelers often consider a spin-up or initialization period at the beginning of the simulation to avoid the effect of initial conditions in the watershed. In such cases, it would be inappropriate to compute any portion of the SSE objective function during spin-up. Thus, the preemption concept would not apply until just after the spin-up period.

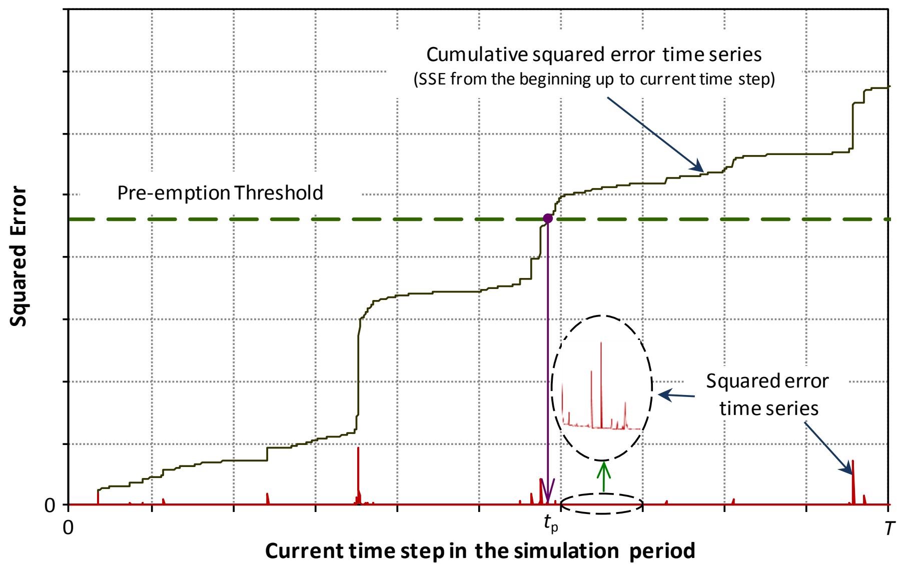


Figure 2-1. Plot of a typical sum of squared errors time series in the course of a model simulation period – the SSE is monotonically increasing over time and may pass a specified preemption threshold (t_p is the earliest possible time of preemption)

2.4 Model Preemption in Optimization-based Calibration Algorithms

Both derivative-based and derivative-free optimizers have been utilized in the calibration of environmental simulation models. Many of these algorithms may be modified to take advantage of the simulation-preemption concept. The following sub-sections highlight a representative sub-set of optimization algorithms and discuss their suitability (or unsuitability) for utilizing the preemption concept. In assessing the suitability of a given algorithm, the key consideration is whether or not a deterministic preemption threshold (i.e., one that definitely does not alter the algorithm behaviour) can be easily defined based on the algorithm search procedure.

2.4.1 Ideal Optimization Algorithms for Model Preemption

Some optimization algorithms are ideal for model preemption in that they stand to benefit considerably in terms of increased computational efficiency. These ideal algorithms have an easily defined preemption threshold that can be utilized to preempt the evaluation of *any* of the candidate solutions generated by the algorithm. Examples of this type of algorithm discussed below include the dynamically dimensioned search [*Tolson and Shoemaker*, 2007b], particle swarm optimization [*Kennedy and Eberhart*, 1995], pattern search [*Torczon*, 1997] and grid search [*Yu et al.*, 2006] algorithms.

Designed for optimization problems with many decision variables, the DDS [*Tolson and Shoemaker*, 2007b] algorithm is a computationally efficient stochastic global search algorithm that requires no algorithm parameter adjustment. In the DDS algorithm, the search dimension is dynamically refined as a function of current iteration number and the user-specified maximum number of function evaluations. DDS is a single-solution based algorithm that always searches from the current "best" solution. It is a greedy type of algorithm since moves that would degrade the current best solution are never accepted and are not utilized in subsequent algorithm decisions. In other words, DDS ignores any candidate solutions which are worse than the current best solution and these inferior solutions do not influence the search strategy. Making use of this fact, an obvious preemption threshold for the DDS algorithm is to utilize the objective function value of the current best solution.

PSO [*Kennedy and Eberhart*, 1995] is a stochastic, population-based, global optimization method inspired by the social behavior of birds and fish. A swarm consists of a population of "particles" distributed throughout the *D*-dimensional parameter space. Each particle has a position vector ($\overline{\mathbf{X}}$)

and a velocity vector (position change) ($\overline{\mathbf{v}}$) and these are updated at each generation, causing the particles to fly through hyperspace. During flight, each particle tracks its own "personal" best solution ($\overline{\mathbf{X}}_{\mathbf{p}}$ and associated objective function F_p), as well as the overall best solution ($\overline{\mathbf{X}}_{\mathbf{g}}$ and associated objective function F_g) discovered by the entire swarm of particles. Velocity updates for each particle are computed each generation using a simple vector calculation that is a random-but-weighted function of (1) the particles current position ($\overline{\mathbf{X}}_{\mathbf{cur}}$); (2) the particles previous velocity (i.e., inertia); (3) the particles personal best position ($\overline{\mathbf{X}}_{\mathbf{p}}$); and (4) the overall best position ($\overline{\mathbf{X}}_{\mathbf{g}}$), Weights are applied to bias particle movement, such that the inertia weight (*w*) biases movement toward the previous velocity, the cognitive weight (c_1) biases movement toward the personal best position, and the social weight (c_2) biases movement toward the overall best position. Detailed descriptions of PSO can be found elsewhere [*Beielstein et al.*, 2002; *Kennedy and Eberhart*, 2001]).

The PSO algorithm only needs to keep track of the personal best and overall best particle positions and corresponding objective function values (i.e. F_p for each particle and F_g) as these are the only solutions that influence the path of each particle. Clearly, for each particle, a given personal best objective function value will always be inferior or equal to the overall best objective function value – for minimization problems $F_p \geq F_g$. The trajectory of individual particles will not be influenced by solutions whose objective function values are inferior to their current personal best (F_p). Making use of this fact, a separate preemption threshold can be conveniently defined for each particle – these thresholds correspond to the current personal best solution of each particle.

The pattern search [*Torczon*, 1997] and grid search [*Yu et al.*, 2006] algorithms are deterministic global optimization algorithms that are able to utilize a straightforward model preemption strategy. For example, in the polling process of the pattern search algorithm the preemption threshold can be conveniently defined to correspond to the objective function value of the current best solution – such an assignment will not affect the overall behavior of the algorithm. Similarly, during evaluation of grid points in the grid search algorithm, the preemption threshold can be defined to correspond to the objective function value of the current best solution.

2.4.2 Moderately Suitable Optimization Algorithms for Model Preemption

Some optimization algorithms would stand to benefit in terms of increased computational efficiency from model preemption but the potential benefits are likely reduced relative to the benefits of the ideal algorithms discussed in 2.4.1. The ability or likelihood of pre-empting solutions in these algorithms is relatively small. Examples of moderately suitable algorithms for preemption discussed below include the Nelder-Mead simplex [*Nelder and Mead*, 1965], shuffled complex evolution [*Duan et al.*, 1993] and all derivative-based optimization algorithms.

In the Nelder-Mead simplex algorithm, a candidate preemption threshold is the objective function value of the worst vertex of d+1 solutions in the current simplex (where d is the problem dimension) – model preemption using such a threshold will not have any effect on the algorithm. Accordingly, since the Nelder-Mead algorithm is utilized in the competitive complex evolution (CCE) sub-module of the popular SCE algorithm, this same model preemption strategy can be used in SCE - the evolved complexes that are returned to the main SCE module prior to reshuffling and repartitioning will not be altered. Preliminary testing of model preemption efficiency gains in the SCE algorithm confirmed that SCE was moderately suitable with achievable computational savings typically less than 10%.

Since the derivative-free Nelder-Mead algorithm is generally considered a local search, the expected preemption savings are less relative to a global search method because a local search would concentrate the search too closely around the best solution or set of solutions found so far and the associated preemption threshold is based on the worse vertex in a simplex. Preemption savings are generally a maximum for global optimization algorithms that are more likely to evaluate relatively poor solutions with significant frequency. Similar to Nelder-Mead, the stepwise linear search algorithm [*Kuzmin et al.*, 2008] proposed for efficient distributed hydrologic model calibration can be considered as a moderately suitable optimization algorithm for model preemption.

The proposed deterministic model preemption strategy has limited applicability and efficiency gains when combined with derivative-based optimization algorithms. In such methods, determining the search direction requires evaluating numerical derivatives and obtaining accurate derivatives requires fully evaluated objective function values. Any model preemption implemented during the search direction step of derivative-based algorithms would yield approximate derivative information and as such is incompatible with the proposed deterministic model preemption concept. However, a preemption strategy might improve the efficiency of line-search algorithms that are used by derivative-based algorithms to adjust the step size (given a derivative-based search direction).

2.4.3 Optimization Algorithms Unsuitable for Model Preemption

Some optimization algorithms such as genetic algorithms (GA – [*Goldberg*, 1989]) and ant colony optimization (ACO - [*Maier et al.*, 2003]) might have substantial difficulty adopting the proposed model preemption strategy. For example, a GA with roulette-wheel selection requires fully evaluated objective function values for the entire population – utilizing preemption would alter the behavior of the algorithm. Although some preemption might be possible in GAs with tournament selection, defining preemption thresholds becomes more complex and the relative potential benefits would put these types of GAs in the moderately suitable algorithm class in Section 2.4.2. Similar to a GA with roulette-wheel selection, the ACO algorithm probabilistically evaluates possible paths (solutions) and utilizes the entire colony of fully evaluated objective function values, even if some members of the colony are massively inferior. Thus, utilizing preemption would alter the behavior of the ACO algorithm.

2.5 Preemption in Uncertainty-based Calibration Algorithms

Apart from treating model calibration as an optimization problem, the introduction of the Generalized Likelihood Uncertainty Estimation (GLUE) method proposed by *Beven and Binley* [1992] has refocused many calibration efforts towards quantifying model prediction uncertainty. In contrast to the optimization-based approach of identifying a single parameter set that minimizes some objective function, procedures developed for uncertainty-based calibration are designed to search some prior probability distribution of the parameter space with the goal of elaborating a calibrated (i.e., posterior or behavioral) probability distribution of model parameters. *Kuczera et al*. (2010) introduced a limited-memory MCMC sampler for Bayesian uncertainty analysis that is conceptually similar to preemption. Here, two other uncertainty-based calibration algorithms were investigated as candidates for the model preemption strategy and are described below.

Generalized Likelihood Uncertainty Estimation (GLUE) is a method introduced by *Beven and Binley* [1992] to quantify the uncertainty of model predictions. GLUE is based on the "equifinality" concept [*Beven and Freer*, 2001], which hypothesizes the existence of several different parameter sets (and multiple model structures) that simulate observed system behaviour equally well. When considering parameter uncertainty, GLUE is focused on identifying multiple acceptable or "behavioral" parameter sets via a Monte Carlo sampling experiment. Therefore, the modeller must subjectively define behavioral in terms of the selected likelihood function by a threshold value for each case study. A large number of GLUE studies have utilized transformations of the sum of squared errors (SSE), especially the Nash-Sutcliffe coefficient, to define the likelihood function.

Typical GLUE studies utilize uniform random sampling and report exceedingly high numbers of model evaluations. For example, the vast majority of GLUE studies report using more than 10,000 model evaluations while multiple studies report using millions of model simulations [*Brazier et al.*, 2000; *Freer et al.*, 2004]. There are definite practical sampling efficiency issues in GLUE that require improved sampling procedures [*Beven*, 2006] and very low behavioural sampling frequencies (<1/1000 for example) are reported in various GLUE studies [*Blazkova and Beven*, 2009; *Freer et al.*, 2004]. Therefore, the GLUE procedure typically samples a very high proportion of solutions that could be terminated early in the simulation with the preemption concept.

In the preemption enabled GLUE, the preemption threshold is simply set equal to the subjective behavioral threshold as determined by the modeller. In the GLUE-based preemption experiments in this study, variable behavioural thresholds are defined using the Nash-Sutcliffe coefficient (ranging from 0.0 to 0.5) and the common uniform random sampling approach is applied.

The DDS-AU methodology [*Tolson and Shoemaker*, 2008] is an uncertainty-based calibration algorithm that enumerates multiple high-likelihood solutions (i.e., parameter sets) using independent DDS optimization trials. Using a relatively small number of model evaluations (e.g., ~100), each DDS trial starts from a different initial solution and follows a different randomized search trajectory, resulting in the identification of a variety of behavioral solutions. In the simplest implementation of DDS-AU, only the best solution from each DDS optimization trial is considered as a possible behavioral sample. Therefore, in DDS-AU, each independent DDS trial can conveniently utilize the DDS preemption strategy (and associated preemption threshold based on the current best) described in Section 2.4.

2.6 Case Studies

To demonstrate the model preemption strategy, it was linked with a representative set of optimization-based and uncertainty-based calibration algorithms for the purpose of calibrating several environmental simulation models. Selected case studies include two SWAT2000 hydrologic (Soil and

Water Assessment Tool, version 2000 – [*Neitsch et al.*, 2001]) model applications, a MESH hydrological model (Modélisation Environnementale de la Surface et de l'Hydrologie – [*Pietroniro et al.*, 2007]) application, and an aquifer parameter estimation problem for the DFRTT interpretation model [*Roos*, 2009]. The calibration periods utilized in case studies above were established based on the original studies where the case studies were first introduced [e.g., *Tolson and Shoemaker*, 2007a; *MacLean*, 2009].

2.6.1 SWAT Case Studies

Tolson and Shoemaker [2007a] utilized SWAT2000 to simulate streamflow and sediment and phosphorous transport into the Cannonsville Reservoir in Upstate, New York. The major land uses in this nearly 1200 km^2 watershed are forest and farmland while urban areas comprise less than 0.5% of the watershed. *Tolson and Shoemaker* [2007b; 2008] derived multiple optimization-based and uncertainty-based calibration problems from the Cannonsville case study to demonstrate the efficiency of the DDS and DDS-UA algorithms. Two of the calibration problems from the Cannonsville watershed case study were included in the suite of model preemption benchmark problems utilized in this study.

SWAT-1 Case Study – Streamflow Calibration at the Walton Gauging Station

The SWAT-1 case study is a streamflow calibration problem where the SWAT2000 model for the Cannonsville Reservoir watershed is calibrated to measured flow at the Walton gauging station (drainage area of 860 km^2) by maximizing the Nash-Sutcliffe coefficient for daily flow. The SWAT-1 problem seeks to calibrate 14 SWAT2000 model parameters that are subject to various range constraints (see Table 2 of Tolson and Shoemaker, [2007a]). The relevant calibration time period is 2191 days (January 1990 to December 1995) and this is preceded by a model initialization period of 1095 days – resulting in a total simulation time period of 2191+1095= 3286 days (9 years). A single SWAT2000 model evaluation of the SWAT-1 case study requires about 1.8 minutes on average to execute on a 2.8GHz Intel Pentium processor with 2 GB of RAM and running the Windows XP operating system.

SWAT-2 Case Study – Streamflow, Sediment, and Phospohorous Calibration

A second calibration problem (i.e., SWAT-2) involving the Cannonsville Reservoir watershed was also included in the benchmarking studies. The SWAT-2 case study differs from the SWAT-1 case study in that the model is calibrated to flow at the Walton gauging station as well as total suspended sediment (TSS) and total phosphorous (P) at the Beerston monitoring station, located few kilometers downstream of Walton. For the SWAT-2 case study a weighted combination of Nash-Sutcliffe coefficients was utilized as the objective function – weights of 0.5, 0.2 and 0.3 were assigned to the Nash-Sutcliffe coefficients for streamflow, total suspended sediments, and total phosphorous, respectively.

The SWAT-2 case study seeks to calibrate 30 SWAT2000 model parameters subject to various side constraints (see Table 3 of *Tolson and Shoemaker* [2007b]). The calibration time period for SWAT-2 is 1553 days (October 1991- December 1995) and this is preceded by an initialization period of 365 days – resulting in a total simulation time period of 1918 days (5.25 years). A single evaluation of the SWAT-2 case study takes about 1 minute on average to execute on a 2.8GHz Intel Pentium processor with 2 GB of RAM and running the Windows XP operating system.

Both the SWAT-1 and SWAT-2 case studies were utilized to benchmark model preemption in an optimization-based calibration context. The SWAT-2 case study was also utilized to demonstrate the model preemption concept in an uncertainty-based calibration context. In this regard, two SWAT-2 uncertainty-based calibration problems were evaluated – one sampled from default parameter bounds while the other utilized reduced parameter bounds as previously defined in Table 3-1 of *Tolson and Shoemaker* [2008].

2.6.2 MESH Reynolds Creek Watershed Model Calibration

The MESH (version 1.2.1) model, currently under development by Environment Canada, is a coupled land-surface and hydrological model. MESH combines the vertical energy and water balance of the Canadian Land Surface Scheme (CLASS) [*Verseghy*, 1991; *Verseghy et al.*, 1993] with the horizontal routing scheme of the WATFLOOD hydrological model [*Kouwen et al.*, 1993]. *MacLean* [2009] applied the MESH model to the Reynolds Creek Experimental Watershed – a research basin maintained by the United States Department of Agriculture, located in south western Idaho. The Reynolds Creek model was calibrated to the measured flow at the Tollgate weir. This sub-watershed (area = 54.5 km^2) encompasses the headwaters of the watershed and receives the highest annual precipitation in the watershed [*Pierson et al.*, 2000]. The MESH model was initialized using soil moisture and soil temperature for September 2, 1986 to minimize the initialization period for the model. For the MESH case study 62 model parameters were calibrated and these are listed along with corresponding ranges in Appendix E of *MacLean* [2009]. Similar to the SWAT-1 case study, the calibration error function for the MESH case study was the Nash-Sutcliffe coefficient for daily flow. Model runs were started on September 2, 1986 and ended on December 31, 1988. The first 120 days were considered as the initialization period and, therefore, a 731 day calibration period started on January 1, 1987. A single evaluation of this model takes about 5.3 minutes on average to execute on a 2.8 GHz Intel Pentium processor with 2 GB of RAM and running the Windows XP operating system.

2.6.3 Dipole Flow and Reactive Tracer Test for Aquifer Parameter Estimation

The final benchmark problem was based on a recently introduced groundwater flow and reactive transport model that is designed to aid in the interpretation of aquifer tests. Because the model is less well-known, we have included below a detailed description of the aquifer test, the corresponding interpretation model, and the calibration case study.

The DFRTT is a single-well test proposed for in-situ aquifer parameter estimation to aid in the design of remedial systems for contaminated sites. The DFRTT circulates groundwater between isolated injection (source) and extraction (sink) chambers within a single well. Once steady-state flow has been reached, a suite of conservative and reactive tracers are added to the injected solution. The concentration of the tracers and their reaction products are monitored in the extracted solution and tracer breakthrough curves (BTCs) are generated. Movement of the tracer through the aquifer is usually preceded by some tracer short-circuiting through the disturbed zone or well skin causing more than one peak in field-measured tracer BTCs. Therefore, the overall shape of a BTC is defined by the magnitude and time of the skin and bulk aquifer peak as well as the shape of the rising and falling limbs. The observed BTCs are analyzed by a DFRTT interpretation model (DFRTT-IM) to estimate aquifer parameters.

The DFRTT interpretation model was developed as a high-resolution two-dimensional radially symmetric finite volume model consisting of two major components: a steady-state groundwater flow component and a reactive transport component. The model was designed so that it could provide an accurate representation of key first-order processes (e.g., biodegradation rate), have the ability to conform to a variety of field configurations and site conditions, and be able to handle a range of input parameters. The major assumptions used to develop this model are: (1) homogeneous aquifer parameters in the vicinity of the test well, and (2) the ambient groundwater flow field does not affect the dipole flow field. To estimate the required aquifer parameters, the simulated BTCs produced by the DFRTT interpretation model must be fit to the observed BTCs using an automated calibration process.

The field test used in this paper was conducted in the unconfined sand aquifer at the Canadian Forces Base (CFB) Borden near Alliston, ON, Canada. There are 7 parameters to be adjusted by the automatic calibration in this study as listed in Table 2-1. The DFRTT model is calibrated according to a Nash-Sutcliffe criterion:

$$E_{NS}(x) = 1 - \frac{\sum_{t=1}^{T} w_t^2 \left(obs_t - sim_t(x)\right)^2}{\sum_{t=1}^{T} w_t^2 (obs_t - \overline{obs})^2} \tag{2-3}$$

where E_{NS} is the weighted Nash-Sutcliffe coefficient and w_t is the weighting function for the BTC with the duration of T (i.e., 240 minutes in this experiment). *x* is a vector of 7 model parameters that are subject to bound constraints in Table 2-1. obs_t and sim_t are the measured and simulated tracer concentrations at time t and $\overline{obs}$ is the average of measured concentrations. Due to the importance of capturing the skin effect, w_t associated with the skin was set at 0.8 while a value of 0.2 was used for the remaining portion of the BTC. A single evaluation of this model takes about 37 minutes on average to execute on a 2.8 GHz Intel Pentium processor with 2 GB of RAM and running the Windows XP operating system.

Table 2-1. DFRTT parameters to be optimized and their ranges

Parameter (units)	Minimum	Maximum
Radial hydraulic conductivity (m/s)	5.00E-06	7.00E-05
Vertical hydraulic conductivity (m/s)	5.00E-06	7.00E-05
Porosity (-)	0.32	0.45
Longitudinal dispersivity (m)	0.004	0.01
Radial hydraulic conductivity of skin zone (m/s)	5.00E-06	7.00E-04
Vertical hydraulic conductivity of skin zone (m/s)	5.00E-06	7.00E-04
Porosity of skin zone (-)	0.3	0.5

2.7 Numerical Experiments

A series of numerical experiments were performed in order to explore the model preemption concept and benchmark potential computational savings in the context of several calibration problems involving real-world environmental modeling case studies. Consistent with the goal of addressing the challenges of calibrating computationally expensive models, all of the selected benchmark problems utilized models with average run-times of at least one minute (i.e., SWAT-1 and SWAT-2) and as high as 37 minutes (i.e., DFRTT-IM). Due to computational considerations, the selected numerical experiments involved only a representative subset of calibration algorithms that were modified to adopt a model preemption strategy. In this regard, PSO and DDS were selected as representative optimization-based calibration algorithms and GLUE and DDS-UA were selected as representative uncertainty-based calibration algorithms. Importantly, the focus of these experiments was *not* to compare results across different algorithms (e.g., PSO vs. DDS), but to explore the potential savings afforded by model preemption for each particular algorithm (e.g., PSO with preemption vs. "standard" non-pre-emptive PSO).

The experimental setup also addressed the potential for variability of results within a given algorithm (e.g., due to stochastic or random nature of the algorithm or alternative treatments of algorithm parameters). For example, unlike the DDS algorithm, several PSO algorithm parameters (i.e., swarm size, number of generations, inertia weight, cognitive weight and social weight) must be defined before applying the PSO to a given problem. Following the recommendations of [*Beielstein et al.*, 2002] our PSO implementation was configured to linearly reduce the inertia weight (w) from a value of 1.2 in the first generation to a value of 0 in the last generation. Conversely, two separate strategies for assigning the cognitive and social weights (c_1 and c_2) were investigated: one strategy utilized constant weights ($c_1= c_2=2$) and another strategy linearly varied the values from $c_1=3$ and $c_2=1$ in the first generations to $c_1=1$ and $c_2=3$ in the last generation. This strategy encourages a gradual transition within the swarm from personal (i.e., global) exploration to local exploration around the overall best solution. As recommended by [*Beielstein et al.*, 2002], both strategies constrain the weights such that $c_1+c_2=4$.

The stochastic nature of the DDS and PSO algorithms may affect the preemption computational savings. For DDS, 5 optimization trials with different random seeds, and in all but the MESH case study, different random initial solutions, were conducted. Although the stochastic nature of PSO might also affect the preemption efficiency, due to computational limitations, only results based on one trial for each of the two PSO parameter configurations were evaluated.

In both optimization-based (i.e., DDS and PSO) and DDS-AU uncertainty-based benchmarks, preemption thresholds are set objectively based on the algorithm search histories and they vary dynamically as the algorithms proceed. In the GLUE uncertainty-based algorithm, the preemption threshold is set equal to the subjectively determined behavioral threshold (as determined by the modeller). Therefore, in the GLUE experiments, different preemption thresholds over the range of 0.0 to 0.5 (in Nash-Sutcliffe scale) were applied to capture the variability and sensitivity of results with respect to the behavioral threshold.

Two parameter ranges (hereafter termed "default" and "reduced") were used for the SWAT-2 case study. The use of default parameter ranges replicates a calibration example where the modeller has little prior knowledge (i.e., expert judgment) and the reduced ranges simulates a scenario in which some prior knowledge was available (e.g., through previous case study or simulation model experience).

Importantly, the experimental setup and associated measure of computational savings leveraged the definition of "deterministic model preemption" (which requires the use of a preemption threshold that has absolutely no effect on algorithm behavior) in order to save the total computation time required to conduct all experiments. As a result, a given experiment could be efficiently performed using *only* preemption enabled algorithms – the computational costs of corresponding non-preemptive algorithms were then inferred using the assumption that for each case study, the average computation time required per pre-empted model time step is equal to the overall average computation time of each model time step. In addition, the model simulation time was assumed to be constant across all model parameter sets. These assumptions are consistent with other studies comparing relative computational burdens of alternative model calibration methods [*Khu and Werner*, 2003; *Tolson and Shoemaker*, 2007b]. However, to verify this assumption, the times needed for evaluating 500 samples for SWAT-2 and 500 samples for MESH generated by Latin hypercube sampling were measured. The model simulation time coefficients of variation obtained for the SWAT-2 and MESH case studies were quite small at 0.06 and 0.02, respectively.

The computational time each model spends on calculations in the initialization period (involves SWAT-1, SWAT-2 and MESH case studies) are ignored in the presented saving results due to the fact that the length of any model initialization period is problem specific and in some calibration problems such as DFRTT-IM, an initialization period is unnecessary. Therefore, the term “calibration period” in this paper is referred to as the period used in the calculation of objective function. Assuming computational cost is the same for all model time-steps, the preemption computational saving for each solution or objective function evaluation is calculated as follows:

$$\text{Saving} = 100 \times \left[\frac{n - n_p}{n} \right] \tag{2-4}$$

where n is the total number of time steps in (or length of) the calibration period, and n_p is the number of time steps simulated in that calibration period before the simulation is terminated (either due to preemption or because the entire calibration period was simulated). In non-pre-emptive algorithms, the calibration period would be fully simulated with each model parameter set using a computational cost of n time steps.

For the SWAT-1, SWAT-2 and DFRTT case studies model preemption was achieved by a separately created monitoring program. Each of the selected calibration algorithms was then configured to call upon this monitoring program whenever a new parameter set needed to be evaluated. The monitoring program would first launch a given simulation and then track simulation progress by periodically reading intermediate model output files and computing a corresponding SSE value. If the computed SSE exceeded a given threshold supplied by the preemption enabled calibration algorithm, then the monitoring program would terminate the simulation using a readily available operating system command (e.g., “TaskKill” in Microsoft Windows or “kill” in the Linux and Mac OS X operating systems). For the MESH case study, an alternative strategy was utilized in which monitoring and terminating a simulation was embedded within the MESH model source code. Each calibration algorithm was then modified to directly provide MESH with the preemption threshold for a given model evaluation and the MESH model would monitor its own progress and as appropriate halt a given simulation at the earliest possible time step of the simulation.

In all but the MESH calibration experiments, two different measures of n_p were calculated, the “actual” n_p value and the “theoretical minimum” n_p value. The actual n_p for each objective function evaluation was recorded as the total number of time steps in the calibration period that were actually simulated before termination of the model and thus measures the actual savings due to preemption. The theoretical minimum n_p was established by post-processing the results of each saved model output file. Measuring savings as a function of the theoretical minimum n_p reflects the maximum savings achievable if the model could be pre-empted at the first possible moment (i.e., the simulation model was developed with an internal preemption capability, as in the MESH case study). Thus, the difference between actual and maximum savings reflect limitations of using a separate preemption monitoring program which might not pre-empt a given simulation until several time-steps after the preemption threshold is exceeded. The computational savings reported in this paper for each calibration experiment or optimization trial is the average savings for all model parameter sets evaluated in that experiment or trial.

2.8 Results

The results are presented in four sections. Sections 2.8.1 and 2.8.2 deal with the performance of the model preemption strategy employed within the DDS and PSO optimization algorithms, respectively. These preemption enabled optimization algorithms were applied to the SWAT-1, SWAT-2 and DFRTT-IM case studies. The preemption enabled DDS was also applied to the MESH case study. Sections 2.8.3 and 2.8.4 present the results of model preemption strategy applied in the GLUE and DDS-AU uncertainty analysis frameworks. The efficiency of these preemption enabled algorithms is demonstrated through the MESH and SWAT-2 case studies.

2.8.1 DDS with Preemption

DDS was applied with the preemption strategy (hereafter termed “DDS with preemption”) and tested on all four case studies. As discussed in Section 2.7 (Numerical Experiments) and due to the effects of randomness on DDS performance, the presented results for all case studies are based on 5 optimization trials. The number of objective function evaluations per optimization trial was set to 600, 1000, 1000 and 250 for SWAT-1, SWAT-2, MESH and DFRTT, respectively. When linked with DDS, the model preemption strategy yielded average maximum theoretical computational savings of 19%, 24%, 49% and 50% of the computational budget for SWAT-1, SWAT-2, MESH and DFRTT, respectively. The actual preemption savings for SWAT-1, SWAT-2 and DFRTT were 14%, 21% and 37%. Since the MESH model utilized an embedded preemption capability, the actual savings were the same as the corresponding maximum theoretical savings (i.e., 49%). Table 2-2 summarizes all preemption savings.

Table 2-2. Summary of maximum theoretical (Max.) and actual computational savings (as a percentage, according to Equation 2-4) due to model preemption strategy employed in DDS, PSO, GLUE and DDS-AU.

	DDS		PSO (Constant weights)		PSO (Dynamic weights)		GLUE Behavioral threshold E_{ns}=0		GLUE Behavioral threshold E_{ns}=0.5		DDS-AU	
	Max.	Actual	Max.	Actual	Max.	Actual	Max.	Actual	Max.	Actual	Max.	Actual
SWAT-1	19	14	39	34	44	40	---	---	---	---	---	---
SWAT-2	24	21	57	53	56	52	58 (95)[a]	52	75 (97)[a]	70	22	18
MESH	49	49	---	---	---	---	95	95	96	96	---	---
DFRTT	50	37	68	59	---	---	---	---	---	---	---	---

a) Using wide or default SWAT2000 parameter ranges.

Figure 2-2, Figure 2-3, and Figure 2-4 show selected measured vs. calibrated (using DDS) results for the SWAT-1, MESH and DFRTT-IM case studies – with corresponding Nash-Sutcliffe coefficients of 0.86, 0.77 and 0.98, respectively. These figures qualitatively demonstrate that even with 1000 or fewer objective function evaluations, good quality calibration results were achieved across the various case studies. The average calibrated Nash-Sutcliffe coefficients over all the DDS trials for the SWAT-1, SWAT-2, MESH and DFRTT case studies are 0.85, 0.66, 0.73 and 0.98, respectively.

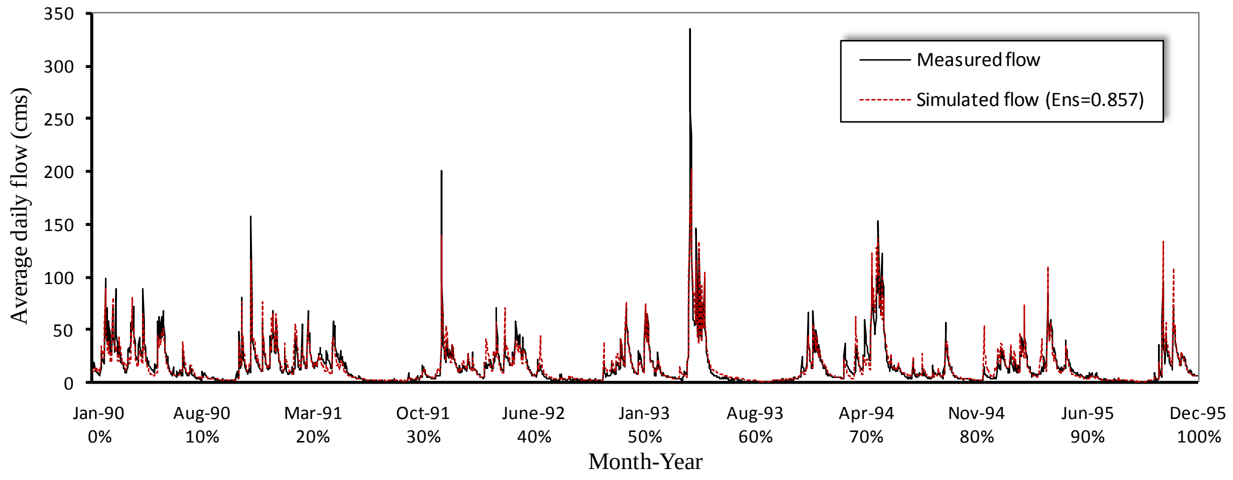


Figure 2-2. Measured and best simulated flow time series of SWAT-1 case study at the Walton station over calibration period (found by DDS).

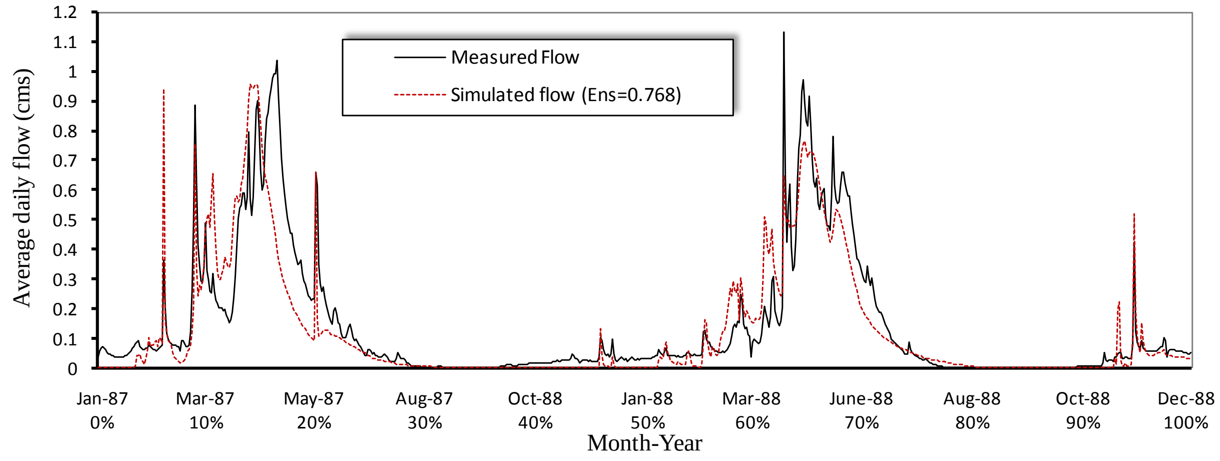


Figure 2-3. Measured and best simulated flow time series of MESH case study at the Tollgate station over calibration period (found by DDS).

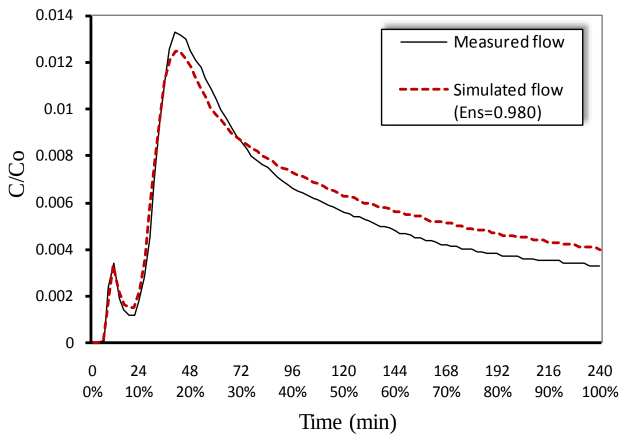


Figure 2-4. Measured and best simulated breakthrough curve of DFRTT case study (found by DDS)

With respect to DDS with preemption, Figure 2-5 shows the empirical cumulative distribution function of the time of model preemption during the calibration period. Model preemption times in Figure 2-5 are the earliest theoretical times at which the model could be pre-empted. According to Figure 2-5a, in the case of SWAT-1, the frequency of model preemption before 55% of the calibration period is simulated is negligible. However, about 40% of model evaluations were preempted before 75% of the calibration period is simulated. Figure 2-5b demonstrates that, when linked with the DDS algorithm, model preemption is active in the early steps of the SWAT-2 simulation so that, for example, about 5% of simulations were pre-empted in the first 15% of the calibration period. Furthermore, the frequency of model preemption before 60% of the simulation is greater than 30%.

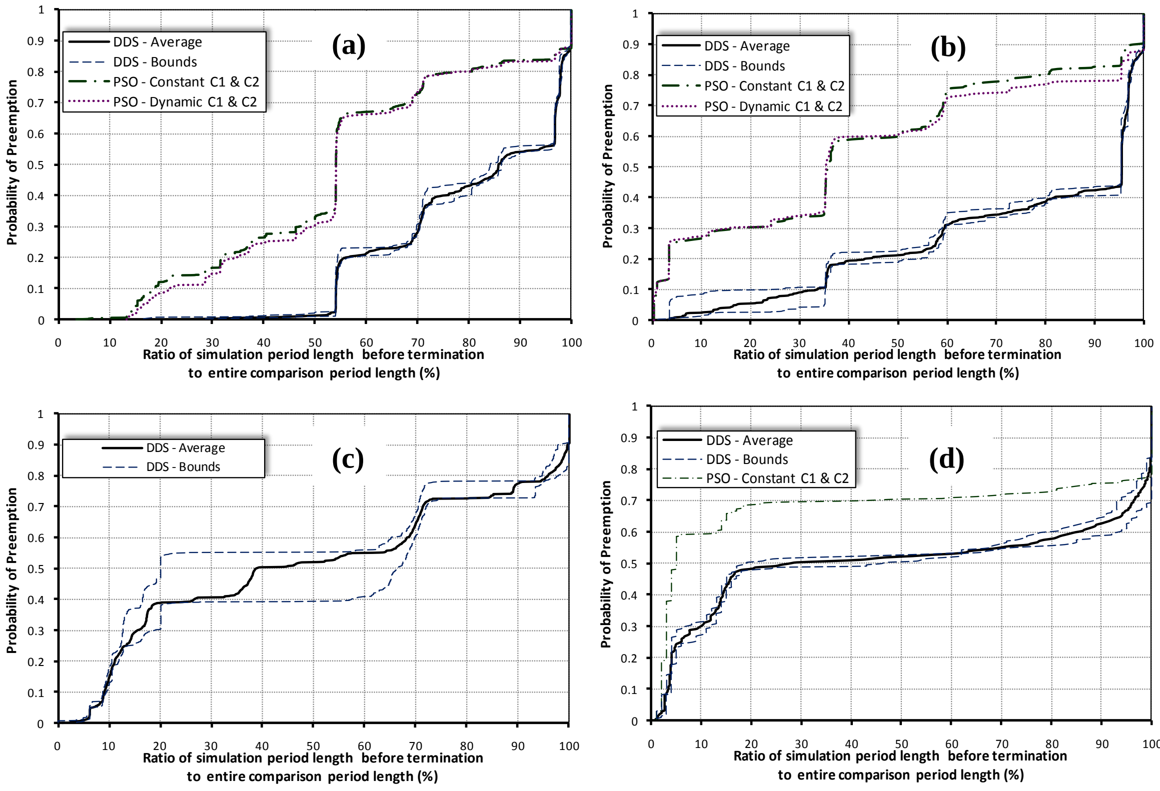


Figure 2-5. Empirical cumulative distribution function of model preemption in DDS and PSO over the course of simulation of (a) SWAT-1, (b) SWAT-2, (c) MESH (without PSO results) and (d) DFRTT – based on earliest theoretical time of termination

For the MESH study, and with respect to DDS with preemption, Figure 2-5c shows that on average about 40% of simulations were terminated in the first 18% of the calibration period and 73% were terminated before simulating 72% of the period. For the DFRTT (Figure 2-5d) case study, model preemption in the DDS algorithm is active from the beginning of the calibration period and the frequency of model preemption within the first 5% and the first 20% of the calibration period were about 25% and 48%, respectively. Note that the high weight of the skin part of breakthrough curve in the DFRTT error function significantly contributed to these early time-step model preemptions.

The narrowness of bounds in Figure 2-5 demonstrates that randomness in the DDS search strategy does not have a considerable effect on the efficiency of the model preemption approach. It is worth noting that the savings afforded by model preemption depends on the model under consideration, the associated parameters being calibrated and the objective (error) function. Furthermore, as can be observed in Figure 2-5, there are a couple of jumps in the empirical CDFs and these jumps represent the active parts of the calibration period with respect to model preemption. For instance, by comparing Figures 2-2 and 2-5a, it is evident that the flood event of April 1993 had a significant influence on model preemption in the SWAT-1 case study. Hence, modellers may increase the potential of model preemption substantially by carefully choosing the calibration period. A more general discussion of this topic is given below, in Section 2.9.1.

Since the potential savings afforded by model preemption depends on the search strategy of the optimization algorithm, Figure 2-6 quantitatively clarifies this behavior over the course of the 5 DDS trials (with 1000 function evaluations) for the SWAT-2 case study. According to this Figure, the cumulative computational savings are relatively high in the early iterations of DDS and these savings decrease gradually as the algorithm progresses toward its final iteration. For instance, an average trial of DDS with preemption saved 16.3% of the total computational budget in the first half of the DDS iterations, but the algorithm achieved only 7.7% savings in the second half of its iterations. This decrease is due to the fact that DDS searches globally (perturbs in all dimensions) in the early iterations and as a function of iteration number it gradually tends to be a more local search (perturbs in only one dimension in the final iterations). Relative to the global search performed in early iterations, during the local search portion of the DDS algorithm it is more likely that model simulations must proceed to near-completion before the preemption threshold is exceeded.

2.8.2 PSO with Preemption

PSO was applied with the model preemption strategy (hereafter termed “PSO with preemption”) and benchmarked using the SWAT-1, SWAT-2 and DFRTT case studies. As explained previously, two PSO strategies (i.e., with and without dynamic cognitive and social weights) were investigated in order to assess potential variability due to different treatments of PSO algorithm parameters. The first strategy using constant weights was applied to all three of the aforementioned case studies, while the second strategy (i.e., dynamic weights) was applied to only the SWAT-1 and SWAT-2 case studies. PSO swarm sizes were set equal to 18, 21 and 16 for the SWAT-1, SWAT-2 and DFRTT case studies, respectively, based on the formula (Swarm_Size = 10+2*(number of decision variables)$^{0.5}$) suggested by *Clerc* [2006]. Based on preliminary "tuning" experiments, PSO trials were run for 100, 150 and 30 generations for the SWAT-1, SWAT-2 and DFRTT case studies, respectively.

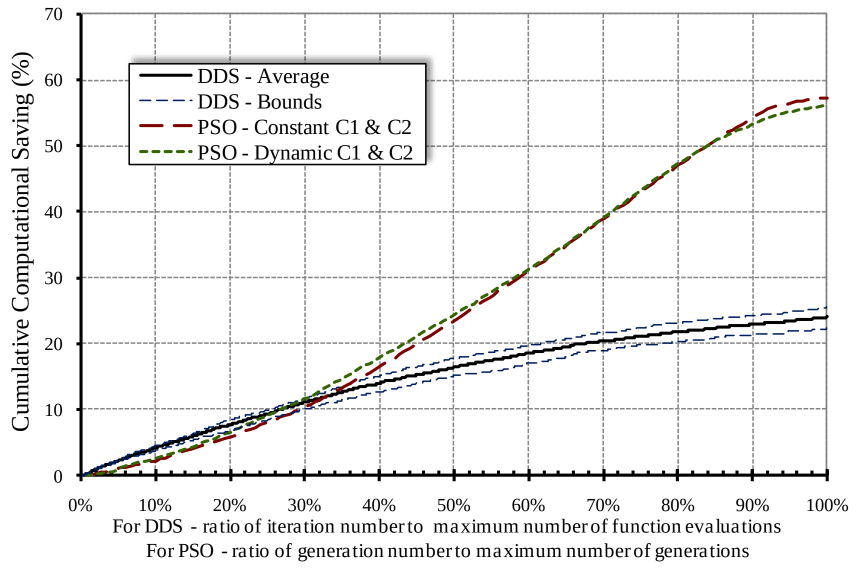


Figure 2-6. Cumulative computational saving (maximum theoretical) over the course of the DDS and PSO trials on SWAT-2 (maximum number of function evaluations in DDS=1000, maximum number of generations in PSO=150 & swarm size=21)

As shown in Table 2, PSO with preemption and using constant cognitive and social weights achieved maximum theoretical savings of 39%, 57% and 68% of the computational budget for the SWAT-1, SWAT-2 and DFRTT case studies, respectively. The actual savings for SWAT-1, SWAT-2 and DFRTT were 34%, 53% and 59%, respectively. Maximum theoretical savings when using dynamic cognitive and social weights were 44% and 56% for the SWAT-1 and SWAT-2 case studies, respectively. Selected "best-fit" PSO calibrations (considering both the constant and dynamic weight variants) yielded Nash-Sutcliffe coefficients of 0.82, 0.61 and 0.98 for the SWAT-1, SWAT-2 and DFRTT case studies, respectively.

With respect to PSO with preemption, Figure 2-5 shows the empirical cumulative distribution function of the time of model preemption during the calibration period. Model preemption times in Figure 2-5 are the earliest theoretical times at which the model could be pre-empted. Because the choice of dynamic vs. constant cognitive and social weights did not demonstrate a significant effect on empirical CDFs, the following analysis applies to both PSO parameter settings. Figure 2-5a illustrates the cumulative behavior of PSO with preemption, as applied to the SWAT-1 case study and illustrates that after the first 14% of the comparison period, the frequency of model preemption increased rapidly so that more than 25% of simulations were pre-empted in the first 40% of the calibration period. Moreover, the probabilities of model preemption before 55% and 75% of comparison period are about 66% and 80%, respectively. According to Figure 2-5b, PSO with preemption was also highly efficient when applied to the SWAT-2 case study. For example, about 12% of simulations were pre-empted in the first 1% of the calibration period. Moreover, in this case, the frequency of model preemption before simulation of 15% and 60% of the comparison period was about 30% and 72%, respectively.

When PSO with preemption was applied to the DFRTT case study (Figure 2-5d), model preemption was active from the start of the calibration period and the frequency of model preemption within the first 5% and 20% of the calibration period was about 58% and 68%, respectively.

Figure 2-6 demonstrates the cumulative maximum theoretical computational savings as a function of generation number over the course of the “PSO with preemption” optimization trials as applied to the SWAT-2 case study. As illustrated in the figure, the alternative PSO treatments had little influence on the amount of computational savings that may be gained through model preemption. In addition, cumulative computational savings increase gradually during the first 85% of PSO generations and then reach a turning point at which the savings are to flatten out. This behavior reflects the fact that the PSO algorithm searches aggressively for the global optimal and transitions only in later generations to a more localized search. The extensive duration of the PSO global search results in relatively frequent evaluation of model parameter sets that are significantly inferior to the “personal best” preemption threshold of individual particles. This characteristic makes the algorithm particularly suitable for the model preemption concept, and the results confirm that significant computational savings can be realized.

2.8.3 GLUE with Preemption

The SWAT-2 and MESH case studies were utilized to demonstrate the efficiency of the uncertainty-based GLUE calibration procedure when it was applied with the model preemption strategy (hereafter termed GLUE with preemption). 10,000 SWAT-2 model evaluations and 10,000 MESH model evaluations were conducted using GLUE with preemption. Uniform random sampling (the most common sampling approach utilized in GLUE applications) was applied to both case studies. In the MESH case study parameter ranges specified in *MacLean* [*MacLean*, 2009] were utilized. In the SWAT-2 case study, the GLUE experiment was repeated twice using both default and reduced model parameter ranges. The use of default parameter ranges replicates a calibration process in which the modeller has little prior knowledge (i.e., expert judgment) of the model as applied to the particular case study. The range reductions simulated a scenario in which some prior knowledge was available (e.g., through previous case study modelling experience).

As reported in Table 2, using a behavioral threshold of 0.5 (Nash-Sutcliffe coefficient), GLUE with preemption yielded maximum theoretical computational savings of 96% and 75% for SWAT-2 with default parameter ranges and SWAT-2 with reduced parameter ranges, respectively. The actual preemption savings for SWAT-2 with reduced parameter ranges was 70%. With the same behavioural threshold of 0.5 for the MESH case study, GLUE with preemption yielded a maximum theoretical computational saving (which is also the actual computational saving) of 96%. However, in all three of these experiments, the GLUE approach was unable to identify a single behavioral sample – indicating that a behavioral threshold of 0.5 is not compatible with the selected number of model evaluations (i.e., 10,000).

Since the choice of behavioural threshold in GLUE studies is a subjective modelling decision, Figure 2-7 shows the total computational savings yielded by GLUE with preemption over a range of behavioral thresholds (i.e., 0.0 to 0.5), as applied to the different case study configurations. The results indicate that the computational savings achieved for two of the case studies (i.e., MESH and SWAT-2 with default parameter ranges) was fairly insensitive to changes in the behavioral threshold. Even with a very relaxed behavioural threshold of 0.0, the GLUE procedure infrequently samples behavioral solutions as the behavioural sampling frequency in the experiments was 894/10000 at best and 0/10000 at worst. This resulted in extremely frequent model preemption and considerable computational savings. When expert judgment was applied to the parameter ranges (i.e., the SWAT-2 case study with reduced parameter ranges), the amount of savings achieved through model preemption was reduced compared to default parameter ranges and became sensitive to the selected behavioral threshold (see Figure 2-7).

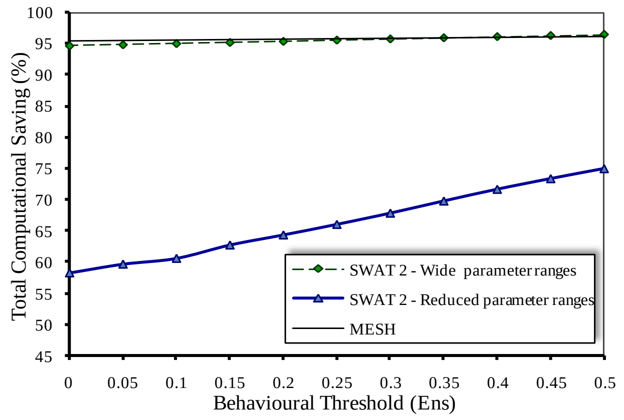


Figure 2-7. Maximum theoretical computational saving due to model preemption strategy in GLUE framework versus behavioural threshold.

2.8.4 DDS-AU with Preemption

The SWAT-2 case study was used to demonstrate the efficiency of the uncertainty-based DDS-AU method with the model preemption strategy (hereafter termed DDS-AU with preemption). Consistent with the "GLUE with preemption" trials, the maximum total number of model evaluations was set equal to 10,000 and DDS trials were run using the reduced parameter ranges configuration of the case study. The number of function evaluations for each independent DDS trial was set equal to 100 based on the recommendation of *Tolson and Shoemaker* [2008]. Therefore, 100 DDS trials with random starting solutions and random seeds were conducted. The total maximum theoretical computational savings yielded by model preemption in the DDS-AU trial was 22% (see Table 2-2). The actual preemption savings for DDS-AU was 18%. Using a behavioral threshold of 0.5, DDS-AU identified 40 behavioral samples out of the 100 DDS trials.

2.9 Discussion

2.9.1 Selecting the Calibration Period

Modellers often have some flexibility in defining their model calibration time period, particularly when the available system response data for calibration is plentiful (i.e., model validation is also to be conducted) and/or when the model is computationally intensive enough to warrant not utilizing the entire set of system response data for calibration. With such flexibility, carefully choosing the calibration period in a given optimization-based or uncertainty-based model calibration problem can significantly improve the computational savings associated with preemption enabled calibration algorithms. As demonstrated by the SWAT-2 case study, some parts of the calibration period are likely to be more fertile than others with respect to triggering model preemption. For example, in rainfall-runoff model calibration, extreme flood events are likely to have a significant influence on overall model performance (i.e., the error function).

For clarification, in Figure 2-5 (which contain empirical CDFs of model preemption as linked with the DDS and PSO algorithms), several jumps in the CDF curves can be observed and these indicate fertile portions of the corresponding calibration period. For instance, comparing Figure 2-2 (measured time series of the SWAT-1 case study) and Figure 2-5a (the corresponding empirical CDF of DDS with preemption), reveals that the flood event of April 1993 corresponds to a jump in the CDF – indicating that this event significantly contributed to model preemption. Therefore, if the calibration period were instead aligned to begin closer to the flood event of April 1993, many model preemptions would generally occur sooner in the calibration period and thus lead to an increase in computational savings (as well as a different calibration result). Although it is impossible to know a priori the fertile periods or events for preemption, it should be clear that it is very likely advantageous with respect to computational efficiency to select (if possible) the calibration period such that the largest magnitude events/responses are not positioned near the end of the calibration period.

2.9.2 Comments on the Model Preemption Strategy

With respect to calibrating environmental models, our results suggest that the computational savings afforded by model preemption can be substantial and the amount of achievable savings depends on the behavior of the selected search (or sampling) strategy. In general, the more global a search (or sampling) strategy is, the greater the expected computational savings that can be achieved through model preemption. This is because a global search, as well as a uniform random sampler, will tend to visit low quality regions of parameter space more frequently than a more localized search that concentrates on portions of the parameter space that are in close proximity to previously identified high quality solutions. Accordingly, it is not surprising that the PSO with preemption algorithm resulted in computational savings that were nearly twice as high as those obtained by the DDS with preemption algorithm.

Overall, and as shown in Table 2, the model preemption strategy is capable of yielding modest (19%) to incredible (97%) computational savings across the various case studies and calibration methodologies considered. When linked with model preemption, the computational efficiency of the PSO, DDS and DDS-UA algorithms were improved by up to 68%, 50%, and 22%, respectively. Of all the considered calibration algorithms, the GLUE approach benefited the most from model preemption – with computational efficiency gains of between 58% and 97% depending on case study and behavioral threshold. In contrast, *Khu and Werner* [2003] demonstrate a meta-modelling approach that incorporates both an artificial neural network and a genetic algorithm to enhance GLUE efficiency and report computational savings of 61% and 80% for two case studies. Unlike *Khu and Werner* [2003], who show that their more complex meta-modelling approach introduces some approximation error and thus modifies the GLUE sampling results, our computational efficiency gains are achieved without introducing any model approximation error and do not change the GLUE calibration result. It is important to note that one factor driving our efficiency gains so high is the fact that the behavioral solution sampling frequency is so low (1/1000 or less) in some of our case studies. Such low behavioral sampling frequencies are not an uncommon phenomenon in GLUE analyses [*Blazkova and Beven*, 2009; *Freer et al.*, 2004].

2.9.3 Model Preemption Limitations

As explained in Section 2.3.1, the deterministic model preemption approach is applicable only when the objective function used is monotonic (e.g., sum of squared errors) or when the objective function can be derived from a monotonic objective function (e.g., Nash-Sutcliffe coefficient). Non-monotonic error metrics used in water resources modeling include: overall volume error or bias, R-squared values and the correlation coefficient. For these types of metrics, the quality of a candidate solution can improve (as well as degrade) during the model simulation period. As a result, intermediate calculations of these types of objective functions are not reliable predictors of solution quality and decisions about solution quality cannot be made with certainty until the given simulation is fully evaluated (i.e., model run is completed). The deterministic preemption framework is not suitable for these types of non-monotonic performance metrics.

A variety of optimization algorithms have been utilized for automatic calibration of environmental models. However, as explained in Section 2.4, only certain algorithms, e.g., DDS and PSO, are ideal for deterministic model preemption. Some optimization algorithms, e.g., SCE, are moderately suitable and some, e.g., ACO and some variations of GAs, are unsuitable for deterministic model preemption.

2.9.4 Guidelines for Implementing Model Preemption

Implementing model preemption with the optimization-based algorithms selected in this study required minor changes to the algorithm source codes since the model preemption thresholds are adaptive by nature to the algorithm progress. However, model preemption can be implemented independent of any GLUE sampling software since the preemption threshold in GLUE is pre-determined by the modeler so that it matches a desired (albeit subjective) behavioral threshold. However, GLUE practitioners may wish to consider a less strict preemption threshold (i.e., greater than the behavioural threshold in minimization problems) to have more fully evaluated solutions, especially when an appropriate behavioural threshold is not clear before sampling.

When a separately created monitoring program is utilized (as in the SWAT-1, SWAT-2 and DFRTT case studies), the frequency at which intermediate simulation results are written to disk is another factor dictating potential computational savings as well as proper monitoring time intervals. Given infrequent updates to intermediate simulation results, a preemption threshold may be exceeded within a model simulation some time before being detected by the preemption monitor, resulting in limited preemption efficiency gains. Our particular model preemption implementation for SWAT-1 and SWAT-2 utilized a monitoring interval of 2 seconds, and as shown in Table 2, the actual computational savings were only 4-5% less than the maximum achievable computational savings. Conversely, DFRTT-IM writes its outputs very infrequently to the disk (only two intermediate writes per simulation), and accordingly, a monitoring interval of 1 minute was utilized. As a result, the actual computational savings for the DFRTT case study were significantly lower than the maximum achievable savings (up to 13% less in DDS with preemption). It should be clear that when implementing model preemption in this way, there is little benefit (less than 1% additional efficiency gain) to monitoring simulation results at more than 100 evenly spaced intervals during a model simulation. Our experience with preemption via a separate monitoring program suggests that this preemption approach is really only beneficial for simulation models requiring one or more minutes for each simulation – otherwise efficient software implementation becomes an important consideration.

For environmental simulation model developers interested in maximizing the computational savings achievable from model preemption, we recommend embedding an internal preemption capability to their model source code, as we implemented in the MESH model. Embedding preemption within the source code means the model would receive the preemption threshold prior to initiating the simulation (e.g., via a text file produced by the automatic calibration algorithm) and could continuously monitor its own progress, terminating as appropriate at the earliest possible moment. In fact, an embedded preemption approach is even more straightforward in environmental modelling software that combines the simulation model source code with an automatic calibration tool and associated model performance metrics. Widely distributed environmental modelling software that combines these capabilities and thus allows users to perform "push-button" automatic calibration include MODFLOWP [*Hill*, 1992], SWAT2005 [*Neitsch et al*, 2004], and MIKE-SHE [*Refsgaard and Storm*, 1995].

2.10 Conclusions

The approach of deterministic model preemption has been formalized to reduce the computational burden associated with both optimization-based and uncertainty-based calibration of computationally expensive environmental simulation models. The approach monitors the intermediate results of a given model simulation and terminates the simulation early if a given problem- and algorithm-specific preemption threshold is exceeded. To formally evaluate the benefits of the approach, it was linked with several readily available calibration algorithms and applied to a suite of calibration case studies. While we expect that a variety of calibration algorithms can make use of the preemption concept, for this study the dynamically dimensioned search (DDS) and particle swarm optimization (PSO) algorithms were selected as representative optimization-based calibration algorithms and the GLUE and DDS-UA algorithms were selected as representative uncertainty-based calibration algorithms.

For the range of calibration case studies considered, numerical results demonstrate that the model preemption strategy can significantly reduce computational costs – with actual savings ranging from 14-49%, 34-59%, 52-96%, and 18% for the DDS, PSO, GLUE and DDS-UA calibration algorithms, respectively, across the multiple calibration case studies considered here. The new model preemption strategy is deterministic, and these computational savings can be realized with absolutely no change to the search algorithm behavior or calibration results.

Model preemption computational efficiency gains are likely to be maximized if it is possible to select the calibration period such that the largest magnitude events/responses are positioned near the beginning of the calibration period. Additionally, maximum theoretical savings were estimated to be higher (4-13% more) than the actual savings that were achievable using a separate preemption monitoring program. Thus, to realize the full potential efficiency savings of the preemption strategy it may be necessary to make modifications to the environmental model being calibrated to include an embedded monitoring and preemption strategy, as was done in this study for the MESH model.

When combined with efficient search algorithms and parallel computing facilities, the model preemption concept has the potential to vastly improve overall model calibration efficiency, relative to corresponding serial computing efforts that do not use preemption. Importantly, these vast improvements would be achieved without the need to approximate and replace the simulation model of interest.

Chapter 3
Surrogate Modelling: Review of the Concepts, Strategies, and Applications in Water Resources

This chapter is a mirror of the following published article. Only minor organizational changes were made to be consistent to the body of the thesis. References are unified at the end of the thesis.

Razavi, S., B. A. Tolson, and D. H. Burn (2012), Review of surrogate modeling in water resources, *Water Resour. Res*., 48, W07401, 32 pages, doi:10.1029/2011WR011527.

Summary

Surrogate modelling, also called metamodelling, has evolved and been extensively used over the past decades. A wide variety of methods and tools have been introduced for surrogate modelling aiming to develop and utilize computationally more efficient surrogates of high-fidelity models mostly in optimization frameworks. This chapter reviews, analyzes, and categorizes research efforts on surrogate modelling and applications with an emphasis on the research accomplished in the water resources field. The review analyzes 48 references on surrogate modelling arising from water resources and also screens out more than 100 references from the broader research community. Two broad families of surrogates namely response surface surrogates, which are statistical or empirical data-driven models emulating the high-fidelity model responses, and lower-fidelity physically-based surrogates, which are simplified models of the original system, are detailed in this chapter. Taxonomies on surrogate modelling frameworks, practical details, advances, challenges, and limitations are outlined. Important observations and some guidance for surrogate modelling decisions are provided along with a list of important future research directions that would benefit the common sampling and search (optimization) analyses found in water resources.

3.1 Introduction

Computer simulation models, which simulate abstract representations of physically-based systems using mathematical concepts and language, are playing a key role in engineering tasks and decision making processes. There are various types of problems utilizing computer simulation models (for simplicity referred to as “simulation models” hereafter) including prediction, optimization, operational management, design space exploration, sensitivity analysis, and uncertainty analysis. There are also problems such as model calibration and model parameter sensitivity analysis dealing with simulation models to enhance their *fidelity* to the real-world system. Fidelity in the modelling context refers to the degree of the realism of a simulation model. Modern simulation models tend to be computationally intensive as they rigorously represent detailed scientific knowledge about the real-world systems [*Keating et al.*, 2010; *Mugunthan et al.*, 2005; *Zhang et al.*, 2009]. Many model-based engineering analyses require running these simulation models thousands of times and as such demand prohibitively large computational budgets.

Surrogate modelling, which is a second level of abstraction, is concerned with developing and utilizing cheaper-to-run "surrogates" of the "original" simulation models. Throughout this thesis, the terms "original functions", "original simulation models", and "simulation models" are used interchangeably. A wide variety of surrogate models have been developed to be intelligently applied in lieu of simulation models. There are two broad families under the large umbrella of surrogate modelling, response surface modelling and lower-fidelity modelling. Response surface surrogates employ data-driven function approximation techniques to empirically approximate the model response. Response surface surrogates may also be referred to as "metamodels" [*Blanning*, 1975; *Kleijnen*, 2009] as a response surface surrogate is a "model of a model". "Model emulation" is another term referring to response surface surrogate modelling [*O'Hagan*, 2006]. The term "Proxy models" has also been used in the literature to refer to response surface surrogates [*Bieker et al.*, 2007]. Unlike response surface surrogates, lower-fidelity surrogates are physically-based simulation models but less-detailed compared to original simulation models, which are typically deemed to be high-fidelity models; they are simplified simulation models preserving the main body of processes modeled in the original simulation model [*Forrester et al.*, 2007; *Kennedy and O'Hagan*, 2000].

In a surrogate modelling practice (response surface surrogates or lower-fidelity physically-based surrogates), the goal is to approximate the response(s) of an original simulation model, which is typically computationally intensive, for various values of explanatory variables of interest. The surface representing the model response with respect to the variables of interest (which is typically a non-linear hyper-plane) is called "response surface" or "response landscape" throughout this thesis. For the majority of response surface surrogate modelling techniques, different response surfaces must be fit to each model response of interest (or each function aggregating multiple model responses). The neural network technique is one exception capable of fitting multiple model responses. In contrast, since lower-fidelity surrogates retain some physically-based characteristics of the original model, one lower fidelity surrogate model is typically capable of approximating multiple model responses of interest.

The main motivation of developing surrogate modelling strategies is to make better use of the available, typically limited, computational budget. *Simpson et al*. [2008] report that the common theme in six highly-cited metamodelling (or design and analysis of computer experiments) review papers is indeed the high cost of computer simulations. Global optimization algorithms based on response surface surrogates such as EGO [*Jones et al.*, 1998], GMSRBF and MLMSRBF [*Regis and Shoemaker*, 2007b], and Gutmann's method [*Gutmann*, 2001] and also uncertainty analysis algorithms such as ACUARS [*Mugunthan and Shoemaker*, 2006] and RBF-enabled MCMC [*Bliznyuk et al.*, 2008] all have been developed to circumvent the computational budget limitations associated with computationally intensive simulation models. In this regard, surrogate modelling may only be beneficial when the simulation model is computationally intensive, justifying the expense of moving to a second level of abstraction (reduced model fidelity) which typically leads to reducing the accuracy of analyses. Therefore, even though *Jones* [2001] and *Simpson et al.* [2008] both point out that surrogate modelling is more than simply reducing computation time, reviewing other possible motivations for surrogate modelling is beyond the scope of this thesis.

Many iterative water resources modelling analyses potentially stand to benefit from surrogate modelling. Benefits are only potential because any surrogate-enabled modelling analysis provides an approximation to the analysis with the original model and the error of the analysis result seems difficult or impossible to assess without repeating the exact analysis with the original simulation model. For example, there is no guarantee that a model parameter deemed insensitive on the basis of surrogate modelling analysis is truly insensitive in the original simulation model. An incomplete list of classic or popular iterative modelling analyses in water resources, which are candidates for efficiency enhancement with surrogate modelling, include deterministic model parameter optimization (calibration) studies with evolutionary algorithms [e.g., *Duan et al.*, 1992; *Wang*, 1991], uncertainty-based or Bayesian model calibration studies [e.g., *Beven and Freer*, 2001; *Kavetski et al.*, 2006; *Vrugt et al.*, 2009b], management or design optimization with evolutionary algorithms [e.g., *McKinney and Lin*, 1994; *Savic and Walters*, 1997], multi-objective optimization algorithms [e.g., *Cieniawski et al.*, 1995; *Reed et al.*, 2003], global sensitivity analysis methods [e.g., *Hornberger and*

Spear, 1981; *Saltelli et al.*, 2000], and any traditional Monte Carlo-based reliability or uncertainty analysis [e.g., *Melching et al.*, 1990; *Skaggs and Barry*, 1997].

This chapter aims to review, analyze, and classify the research on surrogate modelling with an emphasis on surrogate modelling efforts arising from the water resources modelling field. *Simpson et al.* [2001] and *Wang and Shan* [2007] also review the literature on response surface surrogates for engineering design optimization problems. *Simpson et al.* [2004] summarize a discussion panel on response surface surrogate modelling held at the 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis & Optimization. *Simpson et al.* [2008] review the literature on response surface modelling and motivations from a historical perspective and also emphasise the appeal of lower-fidelity physically-based surrogate modelling. *Forrester and Keane* [2009] review recent advances in surrogate modelling including advances in lower-fidelity physically-based surrogates in the field of optimization. Special journal issues on surrogate modelling summarize the first and second International Workshops on Surrogate Modeling and Space Mapping for Engineering Optimization (see *Bandler and Madsen* [2001] and *Bandler et al.* [2008]). Another special issue publication on surrogate modelling is a recent thematic journal issue on surrogate modelling for the reduction and sensitivity analysis of complex environmental models (see *Ratto et al*. [2012]). In addition, there are more specific review papers focusing on specific tools/strategies involved in surrogate modeling. *Kleijnen* [2009] reviews kriging and its applications for response surface surrogate modeling. *Jin et al.* [2001] and *Chen et al.* [2006] review and compare multiple function approximation models acting as response surface surrogates. *Jin* [2005] focuses on response surface surrogate modelling when used with evolutionary optimization algorithms.

Surrogate modelling has been increasingly more popular over the last decade within the water resources community and this is consistent with the increasing utilization of metamodels in the scientific literature since 1990 as documented by *Viana and Haftka* [2008]. A research database search of formal surrogate modelling terminology in mid-2011 (search of Thomson Reuters (ISI) Web of Knowledge) in 50 journals related to surface water and groundwater hydrology, hydraulic, environmental science and engineering, and water resources planning and management returned 110 articles on surrogate modelling in water resources. We believe that the actual number of articles on surrogate modelling in water resources is higher as there are articles not using the formal terminology of surrogate modelling and/or not published in water resources related journals. Forty eight of the available surrogate modelling references published until mid-2011 dealing with water or environmental resources problems were selected for detailed review based on their relevance, our judgement of their quality and clarity of reporting and the significance of surrogate modelling to the contribution of the publication. The phrase “water resources literature” used throughout this chapter refers to the 50 water resources journals and these 48 surrogate modelling references.

3.1.1 Goals and Outline of Review

The primary objective (1) is to provide water resources modellers considering surrogate modelling with a more complete description of the various surrogate modelling techniques found in the water resources literature along with some guidance for the required subjective decisions when utilizing surrogate models. The depth of the review of the topics covered here generally varies with the popularity of the topic in the water resources literature and as such, discussion largely revolves around optimization applications. Additional more specific objectives are as follows: (2) describe each of the components involved in surrogate modelling practice as depicted in Figure 3-1; (3) provide a categorization of the different surrogate-enabled analysis frameworks (i.e., the different ways the components in Figure 3-1 can interact); (4) relate existing surrogate modelling efforts in the water resources literature with similar efforts in the broader research community; and (5) identify relevant underutilized ideas for consideration in future water resources studies.

Figure 3-1 presents a diagram that shows all the components involved in the surrogate modelling analysis framework and the sections in the paper that are directly related to each component. Conventional frameworks not involving surrogate models, such as different simulation-optimization frameworks, consist of only the original model and the search or sampling algorithm components being directly linked together. In surrogate-enabled frameworks, however, three new components, design of experiments, response surface surrogate, and/or lower-fidelity surrogate, may also be involved. These three components, and the framework through which all the components interact, are of particular interest in this paper. Such frameworks generally begin with a design of experiments to generate a sample with which to train or fit a response surface or lower-fidelity surrogate model; then the sampler/search algorithm repeatedly runs the original computationally expensive model and/or the surrogate and collects their response. During this metamodel-enabled analysis, the surrogate model can be static or dynamically updated. Any original model evaluation which is utilized to fit the surrogate model is referred to as a design site. Section 3.2 details the elements associated with the response surface surrogates and presents their advances, considerations, and limitations. Section 3.3

presents the motivation and different types and frameworks for lower-fidelity surrogate modelling. Section 3.4 discusses how the performance of a surrogate-enabled framework should be evaluated and benchmarked against other alternatives. The paper ends with summary and concluding remarks in Section 3.5.

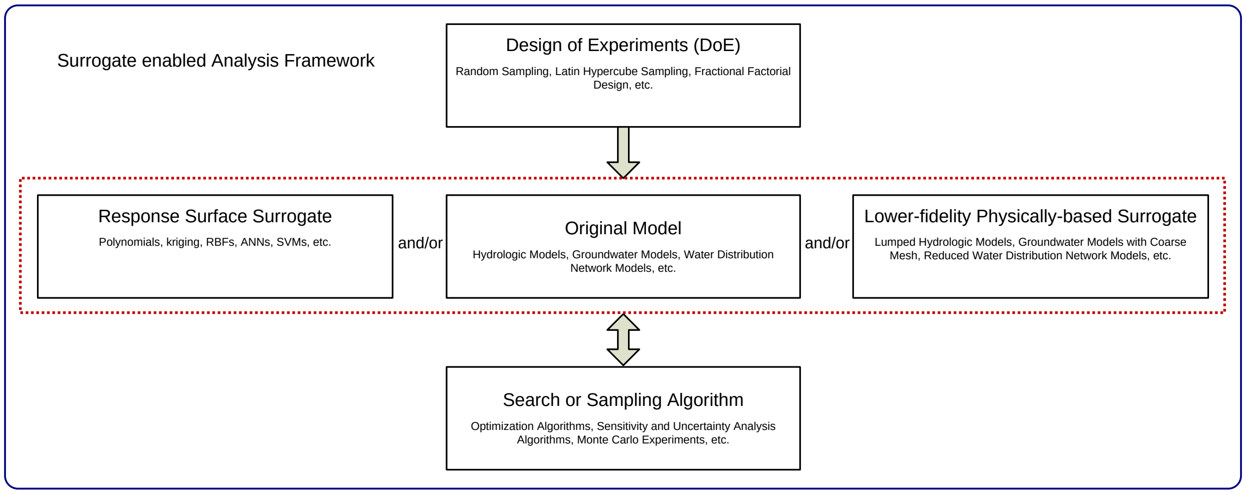

Surrogate enabled Analysis Framework
Design of Experiments (DoE)
Random Sampling, Latin Hypercube Sampling, Fractional Factorial Design, etc.
Response Surface Surrogate
Polynomials, kriging, RBFs, ANNs, SVMs, etc.
and/or
Original Model
Hydrologic Models, Groundwater Models, Water Distribution Network Models, etc.
and/or
Lower-fidelity Physically-based Surrogate
Lumped Hydrologic Models, Groundwater Models with Coarse Mesh, Reduced Water Distribution Network Models, etc.
Search or Sampling Algorithm
Optimization Algorithms, Sensitivity and Uncertainty Analysis Algorithms, Monte Carlo Experiments, etc.

Figure 3-1. Diagram of the elements involved in surrogate (metamodel) enabled analysis framework.

3.1.2 Case Study or Problem Characteristics influencing Surrogate Model Design

The most critical problem characteristics that should influence surrogate model/technique selection are as follows:

1. Model analysis type to be augmented by the surrogate model – search or sampling. For the remainder of this paper, search analysis is meant to refer to optimization (management, calibration, single- or multi-objective) or uncertainty-based/Bayesian model calibration procedure while all other modelling analyses are referred to as sampling analyses.

2. Computational budget constraints. This refers to how many original model evaluations can be utilized to build the surrogate model and ultimately perform the model analysis of interest. In applications where a surrogate model is to be repeatedly utilized after it is initially constructed (i.e., optimize real-time operational decisions), the time available for each utilization can be critical.

3. Dimensionality of the problem. In general, as the number of explanatory variables increases, surrogate modelling becomes less advantageous and even infeasible.
4. Single-output versus multi-output surrogates. This is a key distinction in the context of environmental simulation modelling, where model outputs are typically variable in both time and space. Single-output surrogates are common where the original model output of interest is a function calculated from a large number of model outputs (i.e., calibration error metric).
5. Exact emulation versus inexact emulation. In other words, should the surrogate model exactly predict the original model result at all design sites?
6. Availability of original model developers/experts. Some surrogate modelling techniques require these experts (lower fidelity modelling), and *Gorissen* [2007] notes that they can provide valuable insight into the significance of surrogate modelling errors relative to original model errors.

Although not an original problem characteristic, the availability of surrogate modelling software and experts also has an impact on surrogate model design. The aforementioned surrogate modelling reviews and the literature in general do not precisely map all problem characteristics to specific or appropriate types of surrogate models. We also do not attempt this and instead only make periodic observations and judgements as to when certain types of surrogate modelling techniques might be more or less appropriate than others. Properly considering the case study specific factors above is the first key to avoid implementing a poor surrogate modelling technique.

3.2 Response Surface Surrogates

Response surface surrogate modelling as a research field arising from various disciplines has been in existence for more than six decades and has become very active since the beginning of 1990s [*Simpson et al.*, 2008]. The first generation of response surface surrogates, initiated by *Box and Wilson* [1951], relied heavily on polynomials (typically second-order) and have been the basis of the so-called response surface methodology (RSM). Response surface surrogates do not emulate any internal component of original simulation models; instead they approximate the relationships between several explanatory variables, typically the simulation model parameters and/or variables affecting model inputs, and one or more model response variables. In other words, a response surface surrogate is an approximation or a model of the "original" response surface defined in a problem domain –

response surface surrogates are metamodels of original models. The terms “metamodel” and “response surface surrogate” are used interchangeably throughout this paper. Notably, the existence of the response surface concept dates back to well before formalizing modern response surface approaches as, for example, the traditional non-linear local optimization techniques based on the Taylor series expansion (i.e., different variations of Newton’s method) use simple approximations. In such methods, a response surface surrogate, typically a variation of polynomials, is locally fitted on the (single) current best solution through the use of first- and/or second-order derivative information of the original function, unlike the formalized response surface approaches in which surrogates are fitted on multiple design sites usually regardless of derivatives. Trust-region methods are another example family of traditional local optimization strategies based on the response surface concept, as they iteratively approximate a certain region (the so-called trust region) of the original function typically using polynomials.

Research and advances in response surface surrogate modelling can be classified into three main categories: 1- identification and development of experimental designs for effective approximation, 2- developing and applying function approximation techniques as surrogates, and 3- framework development utilizing surrogates. The research efforts in the water resources community tend to focus on the second and third categories. In the following, the literature on response surface surrogate modelling for water resources applications is reviewed in Section 3.2.1. Then Sections 3.2.2-3.2.6 further detail this review in relation with the above three categories and outline the advances, considerations, and limitations.

3.2.1 Response Surface Surrogates in Water Resources Literature

Response surface surrogate modelling has been widely applied in various water and environmental modelling problems for decades. Table 3-1 summarizes 32 studies utilizing response surface surrogate models in water resources problems published since 2000. Although this table does not cover all surrogate modelling studies over that period, we believe that it provides readers with an adequate coverage of the subject area. Note that not all these studies have been published in water resources related journals. According to Table 3-1, about 45% of the surrogate modelling studies focus on automatic model calibration. Most surrogate-enabled auto-calibration studies involve surrogates in conjunction with optimization algorithms; in four studies, surrogates have been used with uncertainty analysis algorithms for the purpose of model calibration (i.e., GLUE in *Khu and*

Werner [2003] and *Zhang et al.* [2009], ACUARS in *Mugunthan and Shoemaker*, [2006], and Markov Chain Monte Carlo in *Bliznyuk et al*.[2008]). *Schultz et al.* [2004; 2006] and *Borgonovo et al.* [2012] also use surrogates to speed up Monte Carlo sampling studies. Ten studies of the surrogate model applications listed in Table 3-1 are groundwater optimization problems, mostly groundwater remediation. Four surrogate modelling studies are in the context of water distribution system design and optimization. *Borgonovo et al.* [2012] is the only study in Table 3-1 using surrogate modelling for sensitivity analysis and interested readers are thus referred to *Blatman and Sudret* [2010], *Ratto et al.* [2007], and *Storlie et al.* [2009] for metamodel-enabled sensitivity analysis examples from the broader research community. Five studies, *Liong et al.* [2001], *Bau´ and Mayer* [2006], *Behzadian et al.* [2009], *di Pierro et al*. [2009], and *Castelletti et al.* [2010], use response surface surrogates in multi-objective optimization settings. In addition, most studies fit the response surface surrogates on continuous explanatory variables. Five studies [i.e., *Bau and Mayer*, 2006; *Behzadian et al.*, 2009; *Broad et al.*, 2005; *Broad et al.*, 2010; *Castelletti et al.*, 2010] apply surrogates to integer optimization problems (response surface surrogates are fitted on discrete variables), and *Yan and Minsker* [2006, 2011] and *Hemker et al*. [2008] apply surrogate modelling in mixed-integer optimization settings. Note that problems with discrete or mixed variables require special considerations and have been solved largely on an ad-hoc basis [*Simpson et al.*, 2004]. For mixed-integer optimization problems, *Hemker et al*. [2008] utilize response surface surrogates within a branch-and-bound optimization framework such that the surrogate model is employed when solving the relaxed optimization subproblems (integer variables allowed to assume non-integer values). *Shrestha et al.* [2009] use neural networks as a surrogate to completely replace computationally intensive Monte Carlo sampling experiments. In their approach, neural networks are used to emulate the predictive uncertainty (i.e., 90% prediction intervals) of a hydrologic model. The surrogate in this study does not follow the general strategy of response surface modelling where surrogates map model parameters or other problem variables to the system response, and instead their neural network model maps observed rainfall and runoff in the preceding time intervals to the model predictive uncertainty in the next time interval. In Sections 3.2.2-3.2.6, and also Section 3.4, we refer back to the studies listed in Table 3-1 to elaborate the details related to the subject of each section.

Table 3-1. Summary of closely reviewed metamodelling applications in the water resources literature.

Work by:	Type of problem	Type of metamodel	Type of Search or Sampling Algorithm	Type of framework	Number and Type of explanatory variables	Computational saving
Johnson and Rogers [2000]	Groundwater remediation	Linear Regression and Neural network (single hidden layer)	Simulated Annealing	Basic sequential framework	30 Continuous	Not reported
Liong et al. [2001]	Automatic calibration of the HydroWorks watershed model (Bi-objective)	Neural network (three hidden layers)	accelerated convergence GA (ACGA)	Basic sequential framework	8 Continuous	Not reported
Khu and Werner [2003]	Uncertainty based Automatic calibration of the SWMM rainfall-runoff model	Neural network (single hidden layer)	GLUE	Basic sequential framework (Instead of formal DoE, GA with niching was used to generate design sites)	8 Continuous	80% in number of full evaluations
Schultz et al. [2004; 2006]	Uncertainty analysis of a water quality model for regulatory decision support	Linear Regression, polynomial, and process-inspired simple functions	Monte Carlo Simulation	Basic sequential framework	9 Continuous	Not reported
Khu et al. [2004]	Automatic calibration of MIKE11/NAM	Radial basis functions	Genetic algorithms	Metamodel-embedded evolution framework, No formal DoE	9 Continuous	60% in number of full evaluations
Regis and Shoemaker [2004]	Groundwater bioremediation optimization and 17 test functions	Radial basis functions and second-order polynomial (both local)	(μ,λ) -evolution strategy	Metamodel-embedded evolution framework	12 Continuous	Not reported
Ostfeld and Salomons [2005]	Automatic calibration of CE-QUAL-W2 water quality model	*k*-nearest neighbors (kNN)	Genetic algorithms	Metamodel-embedded evolution framework, No formal DoE	3 Continuous	Not reported
Mugunthan et al. [2005]	Automatic calibration of a groundwater bioremediation model	Radial basis functions	Gaussian random sampler	Approximation uncertainty based framework	8 Continuous	Not reported
Broad el al. [2005]	Water distribution system design optimization	Neural network (single hidden layer)	Genetic algorithms	Basic sequential framework	22 Discrete	21% of CPU time
Mugunthan and Shoemaker [2006]	Automatic calibration and parameter uncertainty analysis of groundwater models	Radial basis functions	Gaussian random sampler	Approximation uncertainty based framework	3 and 7 Continuous	More than 87% in number of full evaluations *
Bau´ and Mayer [2006]	Pump-and-treat optimization (Bi-objective)	Kriging	Complete enumeration	Approximation uncertainty based framework	4 Discrete	Not reported
Yan and Minsker [2006]	Groundwater remediation design – MODFLOW and RT3D models used for flow field and contaminant concentration	Neural network (single hidden layer)	Genetic algorithms	Metamodel-embedded evolution framework, No formal DoE	9 and 28 Mixed integer	85-90% in number of full evaluations
Zou et al. [2007]	Automatic calibration of the WASP water quality model	Neural network (single hidden layer)	Genetic algorithms	Basic sequential and adaptive-recursive frameworks	19 Continuous	97% of CPU time
Regis and Shoemaker [2007a]	Groundwater bioremediation optimization and 17 test functions	Radial basis functions	Gaussian and uniform random samplers	Approximation uncertainty based framework	Test functions: 2-14 Groundwater problem: 12 Continuous	Not reported

***** Computational saving is not explicitly mentioned in the paper, and this value is interpreted based on the available information.

Table 3-1. Summary of closely reviewed metamodelling applications in the water resources literature (cont.).

Work by:	Type of problem	Type of metamodel	Type of Search or Sampling Algorithm	Type of framework	Number and Type of explanatory variables	Computational saving
Shoemaker et al. [2007]	Automatic calibration of two SWAT watershed models	Radial basis functions	(μ,λ) -evolution strategy	Metamodel-embedded evolution framework, No formal DoE	8 and 14 Continuous	Not reported
Hemker et al. [2008]	Well field design optimization and hydraulic capture problem	Kriging	Branch-and-bound on main problem - Sequential quadratic programming on sub-problems	Adaptive-recursive framework	5+ Integer with 15+ Continuous	Not reported
Bliznyuk et al. [2008]	Automatic calibration and Bayesian uncertainty analysis of an environmental model	Radial basis functions	Markov-Chain Monte Carlo Sampler	Basic sequential framework	4 Continuous	More than 90% in number of full evaluations
Kourakos and Mantoglou [2009]	Coastal aquifer pumping management	Neural network (modular) single hidden layer	Evolutionary Annealing Simplex Scheme optimization	Metamodel-embedded evolution framework, No formal DoE	34 Continuous (For metamodelling, disaggregated into smaller sub-sets having members with negligible correlations with members of other sub-sets)	95% of CPU time
Fen et al. [2009]	Cost and contaminant removal optimization for soil vapor extraction system design	Second-order polynomials and exponential functions	Genetic algorithms	Adaptive-recursive framework	6 and 9 Continuous	40-94% of total CPU time across 4 case studies *****
Behzadian et al. [2009]	Water distribution system monitoring locations optimization (Bi-objective)	Neural network (single hidden layer)	NSGA-II	Metamodel-embedded evolution framework, No DoE	15 and 50 Discrete	87% and 96% of CPU time (~34% of total saving is due to caching)
Zhang et al. [2009]	Uncertainty-based automatic calibration of SWAT models	Support vector machines (SVMs) and Neural network (single hidden layer)	GLUE (was used only with SVMs, ANN was used for bechmariking SVMs)	Basic sequential framework	6, 9, 12, and 16 Continuous	20-42% of total CPU across 4 case studies
Zou et al. [2009]	Automatic calibration of the WASP water quality model	Neural network (single hidden layer)	Genetic algorithms	Adaptive-recursive framework	19 Continuous	Not reported
Regis and Shoemaker [2009]	Groundwater bioremediation optimization problems, automatic calibration of a groundwater bioremediation model, and 20 test functions	Radial basis functions	Gaussian and uniform random samplers	Approximation uncertainty based framework	Test functions: 2-6 Groundwater optimization problems: 12 Automatic calibration: 6 Continuous	Roughly 50-97% in number of full evaluations *
Shrestha et al. [2009]	Predictive uncertainty estimation of a hydrologic model (HBV)	Neural network (single hidden layer)	N/A (see Section 2.1)	N/A (see Section 2.1)	N/A (see Section 2.1)	Not reported
di Pierro et al. (2009)	Water distribution system design optimization (bi-objective)	Kriging	Genetic algorithms	Approximation uncertainty based	34 and 632 (Continuous)	N/A Inferior performance
Broad et al. [2010]	Water distribution system design optimization	Neural network (single hidden layer)	Genetic algorithms	Basic sequential framework	49 Discrete	98% of wall clock time

***** Computational saving is not explicitly mentioned in the paper, and this value is interpreted based on the available information.

Table 3-1. Summary of closely reviewed metamodelling applications in the water resources literature (cont.).

Work by:	Type of problem	Type of metamodel	Type of Search or Sampling Algorithm	Type of framework	Number and Type of explanatory variables	Computational saving
Castelletti et. [2010]	Water quality rehabilitation in reservoirs (multi-objective optimization)	Radial basis functions, *n*-dimensional linear interpolator, and inverse distance weighted	Complete enumeration	Adaptive-recursive framework	3 Discrete	Not reported
Yan and Minsker [2011]	Reliability-based groundwater remediation design – MODFLOW and RT3D models used for flow field and contaminant concentration	Three Neural networks (single hidden layer)	Noisy Genetic algorithm	Metamodel-embedded evolution framework, No formal DoE	28 Mixed integer	86-90% in number of full evaluations (this figure includes saving due to caching)
Sreekanth and Datta [2011]	Coastal aquifer pumping management	Ensemble of genetic programming-based models	NSGA-II	Basic sequential framework	33 Continuous	Not reported
Razavi et al. [2012]	Automatic calibration of a SWAT model and a groundwater model and 4 test functions	Kriging, radial basis function, and Neural networks (single hidden layer)	Genetic algorithms and Gaussian random sampler	Adaptive-recursive and approximation uncertainty based frameworks	7, 10, 14, and 15 Continuous	Not reported
Borgonovo et al. [2012]	Sensitivity analysis of an environmental nuclear waste model and three test functions	Smoothing spline ANOVA and kriging	Monte Carlo Simulation	Basic sequential framework	Test functions: 2, 3, and 3 Environmental problem: 12 Continuous	96% of CPU time

* Computational saving is not explicitly mentioned in the paper, and this value is interpreted based on the available information.

3.2.2 Design of Experiments

Deterministic simulation systems like computer simulations can be very complex involving many variables with complicated interrelationships. Design of Experiments (also referred to as DoEs) employ different space filling strategies to empirically capture the behaviour of the underlying system over limited ranges of the variables. As a priori knowledge about the underlying (original) response surface is usually unavailable, DoEs tend to assume uniformity in distributing the commonly called "design sites", which are the points in the explanatory (input) variable space evaluated through the original simulation model. Most metamodel-enabled optimizers start with DoEs as in 23 out of 32 studies listed in Table 3-1. There are a wide variety of DoE methods available in the literature; however, full factorial design [e.g., used for metamodelling in *Gutmann*, 2001], fractional factorial design, central composite design [*Montgomery*, 2008], Latin hypercube sampling [LHS - *McKay et al.*, 1979], and symmetric Latin hypercube sampling [SLHS - *Ye et al.*, 2000] appear to be the most commonly used DoE methods. Full and fractional factorial designs and central composite design are deterministic and typically more applicable when the number of design variables is not large. For example, the size of the initial DoE sampled by the full factorial design in a 10 dimensional space with only two levels would be 1024 (=2^{10}), which may be deemed extremely large and beyond the computational budget. Latin hypercube sampling and symmetric Latin hypercube sampling both involve random procedures and can easily scale to different numbers of design variables. Research efforts on the metamodelling-inspired DoEs mostly focus on determining the optimal type [e.g., *Alam et al.*, 2004] and size of existing DoEs [i.e., number of initial design sites – e.g., *Sobester et al.*, 2005] for a specific problem or to develop new and effective DoEs [e.g., *Ye et al.*, 2000].

3.2.3 Function Approximation Models

Response surface surrogates approximate the response surface of computationally intensive models (i.e., original models) by fitting over a set of previously evaluated design sites. A variety of approximation techniques have been developed and applied as surrogates. Examples of such techniques include: polynomials [*Fen et al.*, 2009; *Hussain et al.*, 2002; *Myers and Montgomery*, 2002; *Wang*, 2003], kriging, which is also sometimes referred to as design and analysis of computer experiment (DACE) in the surrogate modelling context [*Sacks et al.*, 1989; *Sakata et al.*, 2003; *Simpson and Mistree*, 2001], *k*-nearest neighbours [kNN - *Ostfeld and Salomons*, 2005], artificial neural networks [ANNs - *Behzadian et al.*, 2009; *Khu and Werner*, 2003; *Papadrakakis et al.*, 1998], radial basis functions [RBFs - *Hussain et al.*, 2002; *Mugunthan et al.*, 2005; *Mullur and Messac*, 2006; *Nakayama et al.*, 2002; *Regis and Shoemaker*, 2007b], support vector machines [SVMs - *Zhang et al.*, 2009], multivariate adaptive regression splines ([MARS - *Friedman*, 1991; *Jin et al.*, 2001], high-dimensional model representation [*Rabitz et al.*, 1999; *Ratto et al.*, 2007; *Sobol*, 2003], treed Gaussian processes [*Gramacy and Lee*, 2008], Gaussian emulator machines [*Kennedy and O'Hagan*, 2001; *O'Hagan*, 2006], smoothing splines ANOVA models [*Gu*, 2002; *Ratto and Pagano*, 2010; *Curtis B. Storlie et al.*, 2011; *Wahba*, 1990], and proper orthogonal decomposition [*Audouze et al.*, 2009]. In the RBF references above, the RBFs are not associated with ANNs, whereas in some publications [e.g., *Jin*, 2005; *Khu et al.*, 2004], as well as in the MATLAB neural network toolbox [*Beale et al.*, 2010], RBFs are considered as a type of feed-forward artificial neural networks.

In the recent water resources related response surface surrogate modelling literature, ANNs and RBFs are the most commonly used function approximation techniques as, among all surrogate modelling studies listed in Table 3-1, 14 and 10 of the 32 studies have applied ANNs and RBFs, respectively. In addition, according to Table 3-1, polynomials have been used in five studies, and in two of these [i.e., *Johnson and Rogers*, 2000; *Regis and Shoemaker*, 2004], polynomials were used to highlight their weakness in comparison with other more promising alternatives. Five studies in Table 3-1 have employed kriging. Each of SVMs, k-NN, and smoothing spline ANOVA has been used in only one study. Moreover, the approximation models mostly act as global surrogates of underlying functions, which represent the original models over the entire input range. However, in some studies [e.g., *Regis and Shoemaker*, 2004; *Wang et al.*, 2004], the approximation model is fitted locally only over a specified (limited) number of design sites (a sub-set of all available design sites), which are in close vicinity of the point of interest in the explanatory variable space.

The information used to fit the response surface surrogates are typically the response values of the original function (i.e., original simulation model) at the design sites; however, there are studies aiming to include the sensitivity (gradient information) of the original function with respect to the explanatory variables (i.e., derivative values) to enhance the approximation accuracy and form the so-called "gradient-enhanced response surface surrogates". Examples include *Kim et al*. [2005] and *van Keulen and Vervenne* [2004] for gradient-enhanced polynomials and *Liu* [2003] for gradient-enhanced kriging and gradient-enhanced neural networks. In practice, such methods have serious limitations as in most of the problems that response surface surrogates are applied, the derivatives are not readily available and have to be calculated using numerical methods requiring the evaluation of the original function at extra points. The extra computational burden imposed can become prohibitively large when the number of dimensions in the explanatory variable space is more than a few, whereas this extra computation could be saved to evaluate the original function at new more intelligently sampled points.

Selection of an appropriate function approximation technique for a given surrogate-enabled analysis requires careful consideration. There are significant differences in the logic inspiring the development of different techniques and also in their level of practicality for a particular problem. In the following, some practical and technical details are presented on five common function approximation techniques used as response surface surrogates of computer simulation models (polynomials, RBFs, kriging, SVMs, and ANNs), all of which have been used for surrogate modelling in the water resources literature. SVMs were selected for detailed review in addition to the four most common techniques in Table 3-1 because *Vianna and Haftka* [2008] include them as one of the four main classes of techniques in their surrogate modelling review paper. The basic information and formulations of these techniques were not included as they are available elsewhere.

Polynomials and Simple Functions

Polynomials have the simplest type of parameters (i.e., coefficients in a polynomial regression), which are objectively determined usually through the least square regression method. Second-order polynomial functions, which are the most popular polynomials used as response surface surrogates, have (D+1)(D+2)/2 parameters where D is the number of explanatory variables (dimension of the input space). First- and second-order polynomials have had very successful applications in local non-linear programming optimization algorithms (e.g., different variations of gradient-descent and Newton/quasi-Newton methods) where, for example, a second-order polynomial is used to emulate a local mode in the original response landscape. However, the use of polynomials as global surrogates may be only plausible when the original response landscape is, or is reasonably close to, unimodal, which is not often the case in many water resources related problems.

Application of higher order polynomials (third-order or more, which are common in curve-fitting) is typically infeasible when D is greater than only a few variables. This is largely because specifying a proper polynomial form for a particular problem may become very challenging. Secondly, the number of polynomial parameters to be tuned (and therefore the minimum number of design sites required – see also Section 3.2.6) becomes excessively large. The inferior performance of polynomials compared to other function approximation techniques mostly due to having fairly inflexible pre-specified forms and being inexact emulators (see Section 3.2.6) has been acknowledged in several studies [e.g., *Hussain et al.*, 2002; *Regis and Shoemaker*, 2004; *Simpson and Mistree*, 2001]. Note that when the form of the underlying function is similar to a polynomial and when this form is known *a priori* [e.g., in *Fen et al.*, 2009] polynomials had been reportedly successful for the problem of interest) polynomials can be one of the best options to choose for surrogate modelling.

Other simple functional forms such as exponential functions [*Aly and Peralta*, 1999; *Fen et al.*, 2009] fitted by least square methods may also be applied for response surface modelling. *Schultz et al.* [2004] and *Schultz et al.* [2006] develop "reduced-form models" based on components of the actual process equations in an original model and fit them to design sites sampled from the original model. They point out that when the specified functional forms "are informed by the mechanics" of the original model, the reduced-form models demonstrate better predictive (generalization) ability.

Notably, what is called a reduced-form model in these publications is different from the lower-fidelity physically-based surrogates outlined in Section 3.3.

Radial Basis Functions

Radial basis function (RBF) models consist of a weighted summation of typically n (sometimes fewer) radial *basis functions* (also called correlation functions) and a polynomial (usually zero- or first- order) where n is the number of design sites. There are different forms of basis functions including Gaussian, thinplate spline, and multiquadric and some forms (e.g., Gaussian) have parameters specifying the sensitivity/spread of the basis function over the input domain, while some (e.g., thinplate spline) have fixed sensitivity (no parameter in the basis function) regardless of the scale (unit) and importance or sensitivity of each input variable. To address the scale problem, all the data are typically normalized to the unit interval. However, in RBF models, the sensitivity of the basis functions in all D directions is typically assumed identical (only one single parameter, if utilized, in all dimensions for all basis functions) treating all variables as equally important, although such an assumption is often not true. An RBF model is defined by the weights of the basis functions, the coefficients of the polynomial used, and the basis function parameter if it exists. Weights of the basis functions as well as the polynomial coefficients can be objectively determined by efficient least squares techniques; however, the basis function parameter is usually specified arbitrarily or by trial and error. The minimum number of design sites required to fit an RBF model is the number of coefficients of the polynomial used to augment the RBF approximation.

Kriging

Application of kriging in the context of design and analysis of computer experiments (DACE) was first formalized by *Sacks et al*. [1989] and since then has been frequently called DACE in some publications [*Hemker et al.*, 2008; *Ryu et al.*, 2002; *Simpson et al.*, 2001]. More recent literature uses DACE to refer to the suite of all metamodel/emulation techniques [e.g., *Ratto et al.*, 2012; *Simpson et al.*, 2008]. Similar to RBF models, the kriging model is also a combination of a polynomial model, which is a global function over the entire input space, and a localized deviation model (correlation model consisting of basis functions) based on spatial correlation of samples. The special feature of kriging (main difference from RBF models) is that kriging treats the deterministic response of a computer model as a realization of a stochastic process, thereby providing a statistical basis for fitting. This capability enables kriging to provide the user with an *approximation of uncertainty* associated with the expected value predicted by kriging at any given point. Approximation of uncertainty is the basis of the so-called “approximation uncertainty based framework” for surrogate-enabled analyses, described in Section 3.2.4. Note that such approximation of uncertainty may be available (heuristically or directly) for other function approximation models (see Section 3.2.4).

As opposed to RBF models, the correlation parameters (sensitivities) in kriging are typically different along different directions in the input space (*D* different values for correlation functions in a *D*-dimensional space), resulting in higher model flexibility. All the kriging parameters, including the correlation parameters, can be determined objectively using the maximum likelihood estimation methodology. Like RBF models, the minimum number of design sites needed to fit a kriging model is the number of coefficients in the polynomial augmenting the approximation. Kriging users only need to specify the lower and upper bounds on the correlation parameters, although the appropriate bounds are sometimes hard to specify [*Kleijnen*, 2009]. The kriging correlation parameters can be interpreted to some extent in that large values for a dimension indicate a highly non-linear function in that dimension, while small values indicate a smooth function with limited variation. Moreover, *Jones et al*. [1998] pointed out that the correlation matrix in kriging may become ill-conditioned (nearly singular) towards the end of an optimization run as then the optimization algorithm tends to sample points near the previously evaluated points (design sites); this ill-conditioning is usually manageable.

Support Vector Machines

Support vector machines (SVMs) are a relatively new set of learning methods designed for both regression and classification. Although SVMs to some extent rely on utilizing the concept of basis (correlation) functions as used in RBFs and kriging (especially when using Gaussian kernel function), unlike RBFs and kriging, they only involve a subset of design sites lying outside an ε-insensitive tube around the regression model response, referred to as support vectors, to form an approximation. When fitting the SVMs on data, the ε-insensitive tube is formed to ignore errors that are within a certain distance of the true values. This capability enables SVMs to directly control and reduce the sensitivities to noise (very suitable for inexact emulation, see Section 3.2.6). The other special feature of SVMs is that in the SVM formulation, there is a term directly emphasizing the regularization (smoothness) of the fitted model. There are two specific SVM parameters which are associated with the radius of the ε-insensitive tube and the weight of the regularization term. The Kernel function used within SVM (e.g., Gaussian function) may also have a parameter to be determined; this parameter acts like the correlation parameters in RBF and kriging models and adjusts the sensitivity/spread of the Kernel function with respect to explanatory variables. Users only require dealing with these two or three SVM parameters, and once their values are available, the dual form of the SVM formulation can be efficiently solved using quadratic programming to determine all the other SVM formulation parameters. SVM can better handle larger numbers of design sites as the operator associated with design site vectors in the SVM formulation is dot product [*Yu et al.*, 2006]. The two aforementioned specific parameters are mutually dependent (changing one may influence the effect of the other) and usually determined through a trial-and-error process or optimization. *Cherkassky and Ma* [2004] present some practical guidelines to determine these parameters. Optimal SVM parameter values are difficult to interpret and relate to characteristics of the response surface.

Neural Networks

Feedforward artificial neural networks (ANNs) are highly flexible tools commonly used for function approximation. ANNs in this paper refer to multilayer perceptrons (MLPs), which are by far the most popular type of neural networks [*Maier et al.*, 2010]. Development and application of ANNs involve multiple subjective decisions to be made by the user. Determination of the optimal structure of ANNs for a particular problem is probably the most important step in the design of ANN-based surrogates. ANN structural parameters/decisions include number of hidden layers, number of neurons in each hidden layer, and the type of transfer functions. Various methodologies have been developed to determine appropriate ANN structures for a given problem, including the methods based on the growing or pruning strategies [*Reed*, 1993] such as the methods presented in *Teoh et al*. [2006] and *Xu et al.* [2006], methods based on the network geometrical interpretation such as *Xiang et al*.'s method [2005], and methods based on the Bayesian approaches such as the methods in *Kingston et al.* [2008] and *Vila et al.* [2000]. However, these methods, each of which may result in an appropriate but different structure for a given problem, typically require extensive numerical analyses on the training data as they generally attempt to test different network structures in systematic ways. Despite these methodologies, trial-and-error is the approach to determine the number of hidden neurons in all ANN-based surrogate modelling studies listed in Table 3-1. Considering alternative types of neural network architectures beyond MLP (such as generalised regression neural network, GRNN [*Maier et al.*, 2010]) may provide another way to reduce or eliminate subjective ANN building decisions.

ANNs with one sigmoidal hidden layer and the linear output layer have been proven capable of approximating any function with any desired accuracy provided that associated conditions are satisfied [*Hornik et al.*, 1989; *Leshno et al.*, 1993]. Although one hidden layer is adequate to enable neural networks to approximate any given function, some researchers argue that neural networks with more than one hidden layer may require fewer hidden neurons to approximate the same function. It is theoretically shown in *Tamura and Tateishi* [1997] that to be an exact emulator (interpolating approximator – see also Section 3.2.6), neural networks with two hidden layers require considerably fewer hidden neurons compared to neural networks with one hidden layer. However, developing an exact emulator is not usually the objective of neural network practitioners (except when used as response surface surrogates of deterministic computer simulation models) as the data used are usually noise-prone and the number of input-target sets is typically large. The need for exact emulation is still a case study specific determination. In addition, to be an exact emulator, excessively large ANN structures are required, whereas such networks would more likely fail in terms of generalization (perform poorly at unsampled regions of input space). Section 3.2.6 deals with exact emulation versus inexact emulation. From a more practical perspective, it is shown in *de Villiers and Barnard* [1993] through extensive numerical experiments that single-hidden-layer neural networks are superior to networks with more than one hidden layer with the same level of complexity mainly due to the fact that the latter are more prone to fall into poor local minima in training.

Neural network practitioners tend to use single-hidden-layer neural networks as, for example, ten of eleven neural network applications in response surface surrogate modelling listed in Table 3-1 have used feed-forward neural networks with only one hidden layer; *Liong et al.* [2001] is the only study in Table 3-1 using more than one hidden layer. Single-hidden-layer neural networks form the approximation by combining m sigmoidal units (i.e., sigmoidal lines, planes, or hyperplanes in the 1-, 2-, or 3-and-more- dimensional problem space) where m is the number of hidden neurons. The number of parameters (weights and biases) of the single-hidden-layer neural networks is $m\times(2+D)+1$ where D is the dimension of input space (i.e., number of input variables of the response surface surrogate). The optimal number of hidden neurons, m, is a function of shape and complexity of the underlying function [*Xiang et al.*, 2005] as well as the training data availability [*Razavi et al.*, 2012a]. In the response surface surrogate modelling context, the form of the original function is often unclear, therefore, the number of data points available (i.e., design sites) for training, p, is the main factor involved in determining m. It is usually preferred that the number of ANN parameters be less (or much less) than *p* as discussed in *Maier and Dandy* [2000], although mathematically, there is no limitation when the number of parameters is higher than *p*. A possible strategy is to enlarge *m* dynamically as more design sites become available. Generally, for a specific problem, there are multiple appropriate ANN structures, and for each structure, there are many appropriate sets of network weights and biases. The term “appropriate” here refers to the network structures and weight and bias values that can satisfactorily represent the training data. Neural network training (adjusting the network parameters) can be a time-consuming optimization process depending on the ANN structure and training method selected. Second-order variations of backpropagation algorithms (i.e., quasi-Newton algorithm such as Levenberg-Marquardt) are the most computationally efficient ANN training methods [*Hamm et al.*, 2007] the role of each weight and bias in forming the network response is typically unclear

3.2.4 Metamodel-Enabled Analysis Frameworks

Research efforts on metamodelling arising from water resources modelling are mainly focused on framework development for utilizing metamodels. Metamodel-enabled analysis frameworks in the literature can be categorized under four main general frameworks outlined in the following. The basic sequential framework and adaptive-recursive framework are multi-purpose frameworks (conceivably applicable in all sampling or search analyses), while the metamodel-embedded evolution framework is clearly limited to search analyses dependent on evolutionary optimization algorithms. The approximation uncertainty based framework is primarily used for search analyses but may also be applicable in some sampling studies.

Basic Sequential Framework

Basic sequential framework (also called off-line) is the simplest metamodel-enabled analysis framework consisting of three main steps. It starts (Step 1) with design of experiment (DoE) through which a pre-specified number of design sites over the feasible space are sampled and their corresponding objective function values are evaluated through the original function. In Step 2, a metamodel is globally fitted on the design set. Then in Step 3, the metamodel is fully substituted for the original model in performing the analysis of interest. In this step, a search or sampling algorithm is typically conducted on the metamodel. The result obtained from the metamodel is assumed to be the result of the same analysis with the original model; for example, in metamodel-enabled optimizers with the basic sequential framework, the optimal point found on the metamodel is typically evaluated by the original function and is considered as the optimal (or near optimal) solution to the original function. In some studies on optimization with metamodelling such as *Broad et al.* [2005] and *Broad et al.* [2010], at Step 3, extra promising points on the metamodel found on the convergence trajectory of the optimization algorithm may also be evaluated by the original function. The size of DoE in Step 1 of this "off-line" framework is large compared to the size of initial DoEs in more advanced "on-line" metamodelling frameworks since almost the entire computational budget allocated to solve the problem is spent on Step 1 in the basic sequential framework.. In this regard, the other frameworks (explained below) may be called "on-line" as they frequently update the metamodel when new data become available.

According to Table 3-1, eleven out of 32 recent metamodelling studies use the basic sequential framework. As the sampler in Step 3, *Khu and Werner* [2003] and *Zhang et al.* [2009] use the GLUE uncertainty-based calibration algorithm, *Schultz et al.* [2004; 2006] use traditional Monte Carlo simulation for uncertainty analysis, *Borgonovo et al.* [2012] use Monte Carlo simulation for sensitivity analysis, and *Bliznyuk et al.* [2008] use a Markov Chain Monte Carlo (MCMC) sampler for uncertainty-based calibration. In all the other studies with the basic sequential framework, different optimization algorithms are used in Step 3. In *Liong et al*. [2001] and *Khu and Werner* [2003], instead of fitting the metamodel over the design sites obtained through a formal DoE, which tends to assume uniformity, an optimization trial is conducted on the original function and the set of points evaluated over the convergence trajectory is used for metamodel fitting (see Section 3.2.5 for discussion on whether an initial DoE is required). Instead of having a conventional DoE at Step 1, *Bliznyuk et al.* [2008] first locate a high posterior density region of the explanatory variable space by direct optimization on the original function and then fit a metamodel on the approximate high posterior region (local) rather than the entire space (global). *Zou et al.* [2007] contrast the basic sequential framework with the adaptive-recursive framework.

Although widely used, the basic sequential framework has potential failure modes arising from the fact that the metamodel, especially when developed off-line, is not necessarily a reasonably accurate representation of the original model in the regions of interest in the explanatory variable space. For example, in the optimization context, the global optimum of the metamodel found in Step 3 (i.e., the final solution returned by the framework) is very unlikely to be a local optimum of the original function. In other words, there is no guarantee that the returned solution is even located close to a stationary point of the original function. Figure 3-2 demonstrates a simple case in which the basic sequential framework fails and returns a solution close to a local maximum of the original function in a minimization problem. The metamodel in Figure 3-2 is a tuned kriging model rather than a conceptual example. The original function used in Figure 3-2 (and also used in Figures 3-3 to 3-7) was generated in this review to demonstrate the behaviour of different surrogate modelling strategies. When the original function is simple (e.g., unimodal functions), the probability of such failures is minimal.

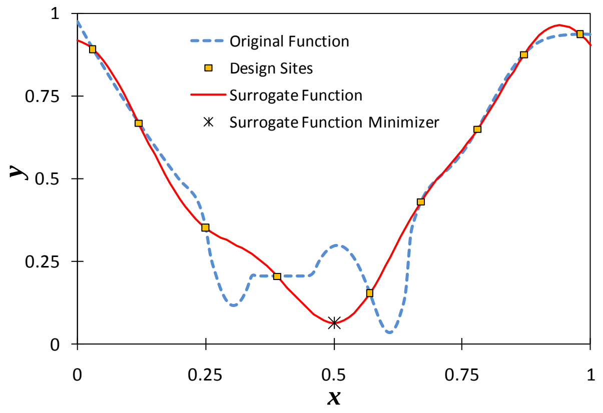


Figure 3-2. A failure mode of basic sequential framework for response surface surrogate modeling where the minimizer of surrogate function very poorly approximates the local or global minimum of the original function.

Adaptive-Recursive Framework

Like the basic sequential framework, in Step 1, the adaptive-recursive framework starts with a DoE to design the initial set of design sites. In Step 2, a global/local metamodel is fitted on the set of design sites. In Step 3, a search or sampling algorithm is employed on the metamodel, identify the regions of interest in the explanatory variable space, and screen out one or multiple points. When used for optimization, an optimization algorithm is typically used to find the near-optimal point (or multiple high-quality points) on the metamodel. The point(s) in the explanatory variable space obtained in Step 3, are evaluated by the original function and added to the set of design sites to update the metamodel. Step 2 and Step 3 are subsequently repeated many times to adaptively evolve the metamodel until convergence or stopping criteria are met. When the adaptive-recursive framework is used for optimization, the best point the framework finds (available in the final set of design sites) is considered as the optimal (or near-optimal) solution to the original function.

Six out of the 32 studies listed in Table 3-1 apply procedures lying under the adaptive-recursive framework. Five studies utilize formal optimization algorithms in Step 3, but *Castelletti et al.* [2010] conduct a complete enumeration at this step since their combinatorial optimization problem has only three decision variables. *Johnson and Rogers* [2000] point out that the quality of solutions obtained through a metamodel-enabled optimizer is mostly controlled by metamodelling performance, not the search technique applied on the metamodel.

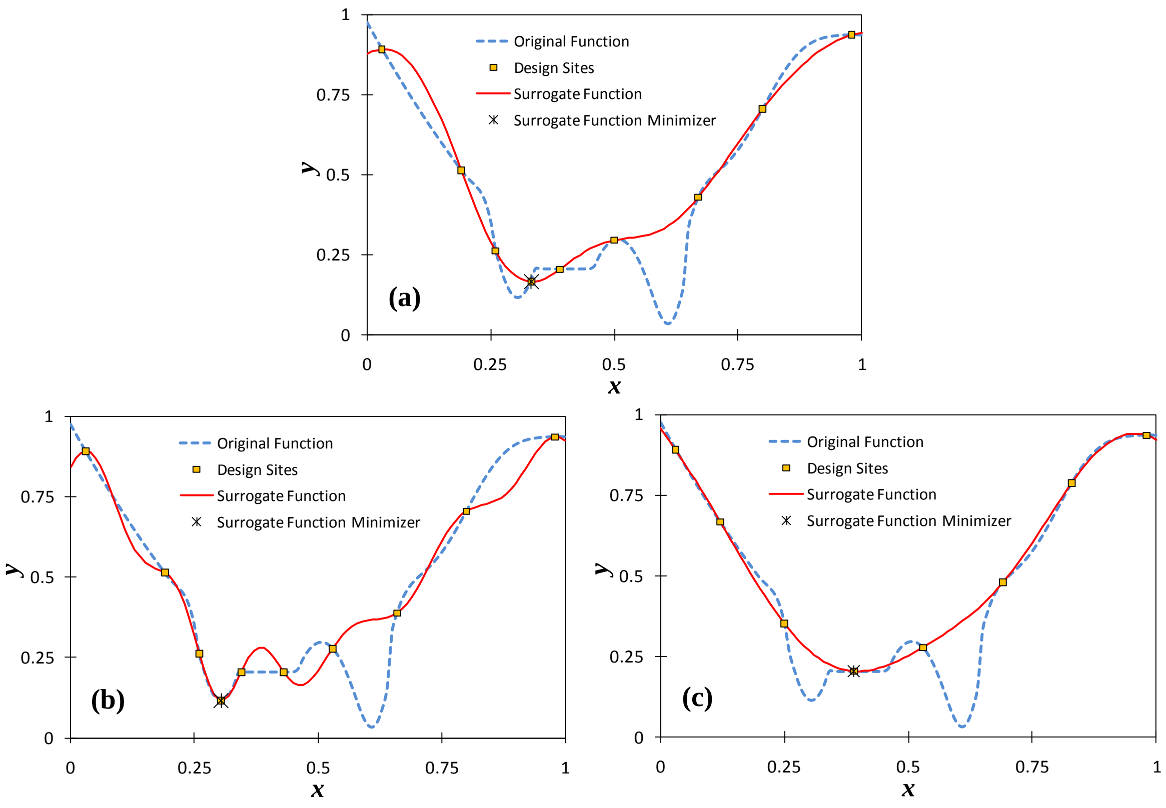


Figure 3-3. Failure modes of adaptive-recursive framework for response surface surrogate modeling: (a) minimizer of surrogate function is located at a previously evaluated point, (b) minimizer of surrogate function is a local optimum of original function far from the global optimum as surrogate function is misleadingly inaccurate in the vicinity of global region of attraction, and (c) minimizer of surrogate function is located at a plateau of original function

The adaptive-recursive framework attempts to address the drawbacks of the basic sequential framework [*Zou et al.*, 2007]. When used for optimization purposes, however, the adaptive-recursive framework is helpful at best for local optimization as reported by *Jones* [2001] who identified some possible cases where the adaptive-recursive framework may even fail to return a stationary point on the original function. Figure 3-3 which depicts possible behaviours of the adaptive-recursive framework, was partially inspired by the discussion in *Jones* [2001]. As shown in Figure 3-3a, if the optimal point on the metamodel (found in Step 3) is located at a previously evaluated point on the original function (i.e., a point already existing in the set of design sites used in Step 2), the algorithm would stall as re-evaluating and adding this point to the set of design sites would not change the metamodel (may also cause mathematical problems in some function approximation methods such as kriging). Another similar failure mode that is less likely to occur is when the optimal point on the metamodel (found in Step 3) has the same objective function value in both the metamodel and the original function (see Figure 3-3a but assume there was no design site at the surrogate minimizer, which is on a non-stationary point of the original function); evaluating and adding this new point to the set of design sites would not have any effect on the updated metamodel. Although it seems very unlikely that Step 3 returns a point exactly the same as the points described in Figure 3-3a, returning new points lying in their close vicinity may also result in very little change in the updated metamodel and lead to the same problems. Some procedure is required in this framework to address these algorithm stalls and revive the search (one example procedure is in *Hemker et al*. [2008]). *Jones* [2001] notes that one way to mitigate these algorithm stalls is to match the gradient of the response surface surrogate with the gradient of the original function at the sampled points (at least on the sample point at which the search stalls). However, as explained in Section 3.2.3, the application of the "gradient-enhanced response surface surrogates" is non-trivial with practical limitations.

The adaptive-recursive framework when used for optimization can easily miss the global optimum of the original function [*Gutmann*, 2001; *Jones*, 2001; *Schonlau*, 1997b; *Sobester et al.*, 2005]. Although significance of missing the global optimum depends on a number of factors, Figure 3-3b depicts a case where the framework has found a local optimum, but as the metamodel is misleadingly inaccurate in the neighbourhood of the global optimum, Step 3 would never return a point in the global region of attraction to be evaluated by the original model. The adaptive-recursive framework with gradient-enhanced response surface surrogates (see Section 3.2.3) may only guarantee convergence to a stationary point on the original function, which may be a global/local optimum, plateau, or saddle point [*Jones*, 2001]. Figure 3-3c shows a case where the framework has converged to a point on a plateau on the original function at which the gradient of the metamodel matches the gradient of the original function. Notably, the surrogate functions in Figure 3-3a-3c represent the actual response of a kriging model developed for the purpose of this review and tuned for the given sets of design sites.

Metamodel-embedded Evolution Framework

The metamodel-embedded evolution framework is a popular framework for optimization. Among the 32 studies on metamodelling listed in Table 3-1, eight studies utilize algorithms lying under this framework. The metamodel-embedded evolution framework shares some characteristics with the adaptive-recursive framework but with significant differences including: the metamodel-embedded evolution framework is inherently designed to be used with evolutionary (i.e., population-based) optimization algorithms, no formal DoE is typically involved at the beginning (except very few studies [e.g., see *Regis and Shoemaker*, 2004]), and decision criteria through which candidate solutions are selected for evaluation by the original function are different. In the metamodel-embedded evolution framework, a population-based optimization algorithm is employed and run initially on the original function for a few generations. All the individuals evaluated by the original function in the course of the first generations are then used as design sites for metamodel fitting. In the following generations, individuals are selectively evaluated by either the metamodel or the original model. The metamodel is usually updated (re-fitted) a couple of times in the course of optimization as more design sites become available. *Jin et al*. [2002] adopt the metamodel-embedded evolution framework and formalize the concept of *evolution control* with two approaches: *controlled individuals* through which a subset of the individuals in the population at each generation (i.e., the best η individuals or η randomly selected individuals) are evaluated by the original function and *controlled generations* through which all the individuals in the population, but at selected generations, are evaluated by the original function. The parameters of metamodel-embedded evolution framework, such as η and the frequency of metamodel updating, can be adaptively changed in the course of optimization depending on the accuracy of the metamodel.

When an evolutionary algorithm is enabled with metamodels through the metamodel-embedded evolution framework, it typically exhibits less consistent behaviour compared to when the same evolutionary algorithm is used without metamodelling [*Jin et al.*, 2002]. This degradation in stability is expected as such an algorithm switches between two different response landscapes (the original function and the metamodel) while searching. A metamodel-enabled optimizer with this framework should satisfy at least two necessary conditions to become a global optimizer: 1- the evolutionary algorithm used should be a global optimizer, and 2- any solution (i.e., any individual in any generation) regardless of the approximate function value obtained by the metamodel should have the chance to be selected for evaluation through the original function. Thus, like the optimizers with the adaptive-recursive framework, metamodel-enabled optimizers with evolution control strategies that only evaluate best individuals (best in terms of approximate function values) through the original function are at best local optimizers. In such cases, the failure modes explained in Figure 3-3 for the adaptive-recursive framework may also apply to the metamodel-embedded evolution framework. Convergence properties of metamodel-embedded evolution framework with other evolution control strategies could likely overcome these failure modes.

Some variations of the Learnable Evolution Model (LEM) [*Michalski*, 2000] might also fall under the metamodel-embedded evolution framework. LEM is an attempt to improve upon the basic and inefficient Darwinian evolution operators by using machine learning tools. Unlike the conventional metamodelling practice which typically produces a continuous approximate response surface, LEM employs classifying techniques (e.g., decision tree learners) to discriminate promising/non-promising solutions based on the search history. The classifier in LEM can involve domain-specific knowledge for rule induction. *Jourdan et al.* [2006] develop a multi-objective version of LEM, called LEMMO (LEM for multi-objective), for water distribution network design problems. LEMMO embeds a decision tree classifier within the NSGAII multi-objective optimization algorithm; the solutions generated by the evolution operators in NSGAII are first evaluated by the classifier and then modified if needed. *di Pierro et al.* [2009] compare LEMMO with ParEGO which is a multi-objective metamodel-enabled optimization algorithm on water distribution network design problems.

Approximation Uncertainty based Framework

The approximation uncertainty based framework may be deemed as an extension to the adaptive-recursive framework designed for optimization. The adaptive-recursive framework relies solely on the approximate values from the response surface surrogate in the course of optimization. As the adaptive-recursive framework assumes these approximate values as true, it may easily miss the main region of attraction where the global optimum lies. This behaviour is illustrated above and also reported in *Gutmann* [2001], *Jones* [2001], *Schonlau* [1997a], and *Sobester et al*. [2005]. Addressing this shortcoming, the approximation uncertainty based framework considers the uncertainties associated with the approximation. In this framework, the approximation value resulting from the response surface surrogate is deemed as *approximation expected value* and then a measure is utilized to quantify the associated *approximation uncertainty*. Such a measure is explicitly available in some function approximation techniques, e.g., in polynomials, kriging , Gaussian Radial Basis Function models [*Sobester et al.*, 2005], and smoothing spline ANOVA [*Gu*, 2002]. To provide the approximation uncertainty, these techniques assume that the deterministic response of a simulation model is a realization of a *stochastic process*. Unlike these techniques, which provide some statistical basis for approximation uncertainty, *Regis and Shoemaker* [2007b] propose a distance-based metric (i.e., the minimum distance from previously evaluated design sites) as a heuristic measure of the approximation uncertainty applicable to any given response surface surrogate. Other deterministic function approximation techniques may also provide measures of uncertainty when trained with Bayesian approaches; for example, Bayesian neural networks [*Kingston et al.*, 2005] provide the variance of prediction. Bayesian learning typically involves a Markov Chain Monte-Carlo procedure, which is more computationally demanding than conventional learning methods.

The approximation uncertainty based framework consists of the three steps outlined in the adaptive-recursive framework with a major difference in Step 3 being, instead of optimizing the approximate response surface (i.e., the surface formed by approximation expected values) as in the adaptive-recursive framework, it optimizes a new surface function that is defined to emphasize the existence of approximation uncertainty. Different ways have been proposed in the literature to define such a surface function, each of which emphasizes the approximation uncertainty to a different extent. In this regard, the adaptive-recursive framework, in which the surface function totally ignores the approximation uncertainty, can be deemed an extreme case that typically yields a local optimum. The other extreme is to build and maximize a surface function solely representing a measure of approximation uncertainty (e.g., approximation variance available in kriging) over the explanatory variable space. In the approximation uncertainty based framework with such a surface function, Step 3 returns the point where the approximation uncertainty is the highest, and as such, subsequently repeating Step 2 and Step 3 in the framework would evolve a globally more accurate metamodel on the basis of a well-distributed set of design sites. Although globally convergent under some mild assumptions [*Sobester et al.*, 2005], the framework solely using the approximation uncertainty would require impractically large number of original function evaluations, especially when the number of decision variables is more than a few.

An effective uncertainty based surface function to be optimized in Step 3 combines the approximation expected value and the associated approximation uncertainty in a way that balances exploration (i.e., searching unexplored areas of high approximation uncertainty) and exploitation (i.e., fine-tuning a good quality solution by reducing the attention to approximation uncertainty) during the search. Most common methods to define the uncertainty based surface functions assume that the hypothetical stochastic process is normal with the expected value $\hat{y}$ and standard deviation σ generated by the surrogate model for any given point in the explanatory variable space, *x*. Figure 3-4a depicts the information typically available from a response surface surrogate that is capable of producing approximation uncertainty – this plot is the outcome of an actual experiment with kriging. There are two popular approaches to make use of such information in Step 3 of the approximation uncertainty based framework: 1- maximizing a new surface function representing the *probability of improvement* and 2- maximizing the so-called *"expected improvement"* surface function [*Schonlau*, 1997b].

Figure 3-4(a-c) illustrates the concept of probability of improvement during optimization based on a real experiment with a tuned kriging model. As can be seen, f_{min} is the current best solution found so far (the design site with the minimum original function value), and as such, any function value which lies below f_{min} is an improvement. Thus, at any given point $\boldsymbol{x}$ in the explanatory variable space, a possible improvement, *I*, and the probability of improvement, *PI*, over the current best are given by:

$$I = \begin{cases} f_{min} - Y(x) & if \quad Y < f_{min} \\ 0 & otherwise \end{cases} \tag{3-1}$$

$$PI = \phi\left(\frac{f_{min} - \hat{y}(x)}{\sigma(x)}\right) \tag{3-2}$$

where $Y(\boldsymbol{x})$ is a possible function value at the point $\boldsymbol{x}$ being a random number following $N(\hat{y}(\boldsymbol{x}), \sigma(\boldsymbol{x})^2)$, and ϕ is the normal standard cumulative distribution function. Notably, if the probability of improvement over the current best is used as the surface function in the framework, the search will be highly local around the current best solution unless there is a point on the response surface surrogate having an estimated expected value, $\hat{y}$, less than f_{min}. To address this drawback, a desired target improvement, *T*, which is smaller than f_{min}, is assumed and then the probability that the original function value is equal or smaller than T would be:

$$PI = \phi\left(\frac{T - \hat{y}(x)}{\sigma(x)}\right) \tag{3-3}$$

The higher the desired improvement is (the smaller the value of *T* is), the more global the search is. As such, when desired improvement is assumed very small, the search would be very local typically around the current best until the standard error of the surrogate in that local area becomes very small. Very large desired improvements (*T* values much smaller than f_{min}) may force the algorithm to search excessively global resulting in very slow convergence. As *Jones* [2001] also points out, the performance of the algorithm based on the probability of improvement is highly sensitive to the choice of desired target, *T*, and determining the appropriate value for a given problem is not trivial. To diminish the effect of this sensitivity, *Jones* [2001] presents a heuristic way to implement the probability of improvement approach based on multiple desired target values. Further details are available in *Jones* [2001], *Sasena et al*. [2002], and *Watson and Barnes* [1995].

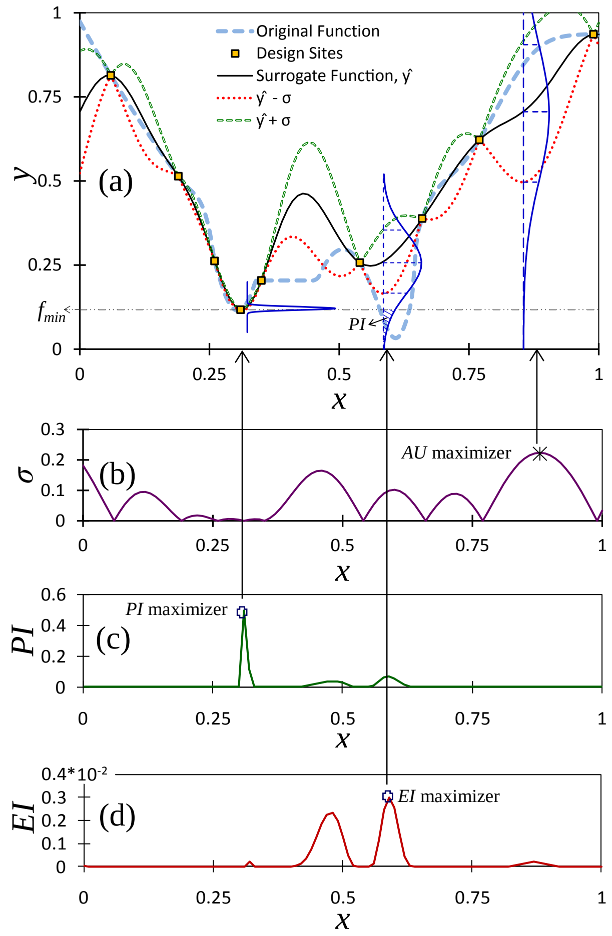


Figure 3-4. (a) Probabilistic output of a response surface surrogate capable of producing approximation uncertainty assuming the deterministic response of simulation model follows a normal stochastic process, (b) approximation uncertainty function (*AU* or *σ*), (c) function representing probability of improvement (*PI*), and (d) function representing expected improvement (*EI*)

The so-called expected improvement approach might be considered as a more advanced extension to the probability of improvement approach. Expected improvement is a measure that statistically quantifies how much improvement is expected if a given point is sampled to evaluate through the original function. Expected improvement at a given point $\boldsymbol{x}$, $EI(\boldsymbol{x})$, is the expectation of improvement, $I(\boldsymbol{x})$, (defined in Equation 3-1) for all possible $Y(\boldsymbol{x})$ function values, which follow $N(\hat{y}(\boldsymbol{x}), \sigma(\boldsymbol{x})^2)$, and calculated by:

$$EI = \left(f_{min} - \hat{y}(x)\right)\Phi\left(\frac{f_{min}-\hat{y}(x)}{\sigma(x)}\right) + \sigma(x)\phi\left(\frac{f_{min}-\hat{y}(x)}{\sigma(x)}\right) \tag{3-4}$$

where Φ is standard normal cumulative distribution function, and ϕ is standard normal probability density function. Interested readers are referred to *Schonlau* [1997b] for derivation of Equation 3-4. Figure 3-4d is a real example of the new surface function formed by the expected improvement approach. The EGO (Efficient Global Optimization) algorithm developed by *Jones et al*. [1998] is the most commonly used metamodel-enabled optimizer with the approximation uncertainty based framework that utilizes the expected improvement surface function.

The approximation uncertainty based framework may utilize a different statistical approach than the approaches explained above to build a new surface function to use in Step 3. This approach, introduced by *Gutmann* [2001] and implemented first using Gaussian radial basis functions, hypothesizes about the location and the objective function value of the global optimum, and then evaluates the "credibility" of the hypothesis by calculating the likelihood of the response surface surrogate passing through the design sites conditioned to also passing through the hypothetical global optimum. As such, for any given hypothetical objective function value, a surface function can be formed over the entire variable space representing the credibility of having the hypothetical objective function value at different points. If kriging is used as the response surface surrogate, its parameters can also be optimized for any given point in the variable space to maximize the conditional likelihood [*Jones*, 2001] – different kriging parameter values are generated for different points in the variable space. *Jones* [2001] points out that the key benefit of this approach over the expected improvement approach is that the next candidate solution in this approach is not selected solely based on the parameters of the surrogate, which may be in substantial error when the initial design sites are sparse and/or deceptive. However, as even an estimate of the optimal objective function value is not known *a priori* in many real-world problems, hypothesizing about the optimal objective function value to be used in the framework may be non-trivial. Therefore in practice, the hypothetical optimal function value(s) should be heuristically defined and changed as the framework proceeds [*Gutmann*, 2001; *Jones*, 2001]. In addition, this approach is typically more computationally demanding than the expected improvement approach, especially when the surrogate parameters (e.g., kriging parameters) are also to be optimized to calculate the conditional likelihood for any given point. *Regis and Shoemaker* [2007a] propose a strategy to improve the method by *Gutmann* [2001] by controlling and increasing its local search ability.

Among the studies listed in Table 3-1, three studies utilize the statistics-based approaches for the approximation uncertainty based framework: *di Pierro et al*. [2009] use the expected improvement approach through the ParEGO algorithm, *Mugunthan et al.* [2005] use the conditional likelihood approach developed by *Gutmann* [2001], and *Bau and Mayer* [2006] make use of the kriging error estimates to make termination decisions in their surrogate-enabled optimization framework. *Regis and Shoemaker* [2007b] also emphasize the need of considering the approximation uncertainty from a non-statistical point of view and develop a metamodel-enabled optimizer involving a heuristic distance-based measure indirectly representing the uncertainty in approximation. Among the studies listed in Table 3-1, *Mugunthan et al*. [2005], *Razavi et al*. [2012a], and *Regis and Shoemaker* [2007b; 2009] use the *Regis and Shoemaker* [2007b] metamodel-enabled optimizer.

The metamodel-enabled optimizers with the approximation uncertainty based framework such as EGO are essentially for global optimization. They work very well when the general shape of the original function and its degree of smoothness are properly approximated typically as a result of a good initial DoE [*Schonlau*, 1997b]. However, there are two possible drawbacks associated with this framework. Any new surface function based on the approximation uncertainty is highly multi-modal, and as such finding its global optimum in Step 3 is not easy especially when the number of decision variables is large. To search for the global optimum in Step 3 of the framework, the original EGO algorithm [*Jones et al.*, 1998] uses the branch-and-bound algorithm, the EGO-based metamodel-enabled optimizer developed in *Sobester et al*. [2005] uses a multi-start BFGS algorithm, and the algorithm by *Regis and Shoemaker* [2007b] uses a brute-force random sampler.

The other possible drawback of this framework arises from the fact that the approximation uncertainty based framework relies extensively on a measure of approximation uncertainty to select the next candidate solution as if it is *correct*; whereas such a measure is only an estimation of the approximation uncertainty suggesting that if poorly estimated, the approximation uncertainty would be very deceptive in guiding the search. The approximation uncertainty at a given point is mainly a function of the degree of non-smoothness of the underlying (original) function being approximated through the design sites and then the distance from surrounding previously evaluated points (i.e., design sites) (note that the measure of uncertainty proposed in *Regis and Shoemaker* [*Regis and Shoemaker*, 2007b] is solely based on distance). As such, if the degree of non-smoothness of the original function is poorly captured by poorly distributed initial design sites, the framework would result in a very slow convergence or even, in extreme cases, premature stalls.

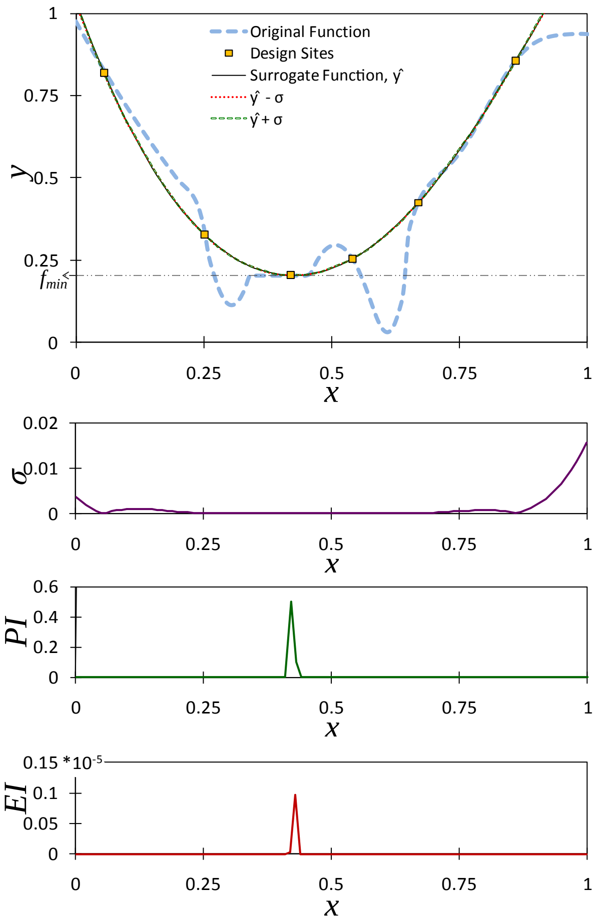


Figure 3-5. A possible failure mode of approximation uncertainty based framework when the degree of non-smoothness of the original function is largely underestimated and as such the standard deviation of surrogate function, σ, is misleadingly very small resulting in an exhaustive search around the minimizer of surrogate function; $\hat{y} + \sigma$ and $\hat{y} - \sigma$ series are almost on $\hat{y}$ and not visually distinguishable.

Figure 3-5 demonstrates one of our experiments with kriging where the degree of non-smoothness of the original function is underestimated by the response surface surrogate due to a poor distribution of the initial design sites. In such a case, the framework tends to search locally around the minimum of the surrogate (similar to the behaviour of the adaptive-recursive framework) until a point better representing the degree of non-smoothness of the original function is evaluated or until an exhaustive search is completed in this local region of attraction, which results in very small approximation uncertainty values at this region guiding the search to other regions. *Jones* [2001] identifies an extreme case where the framework stalls. There are different modifications in the literature to improve upon the approximation uncertainty based framework when enabled with the expected improvement concept. These modifications, include the generalized expected improvement [*Schonlau*, 1997b] and the weighted expected improvement [*Sobester et al.*, 2005], dynamically changing the emphasis on the global search capability of the approximation uncertainty based framework.

3.2.5 Design Considerations of Metamodel-enabled Search (Optimization) Frameworks

Local Optimization vs. Global Optimization

As illustrated in Section 3.2.4, the metamodel-enabled optimizers under the approximation uncertainty based framework aim to improve upon other frameworks by recognizing the importance of approximation uncertainty for global optimization. In this framework, the chance of missing unexplored promising regions in the feasible space is reduced, and the evolved metamodels have more uniform global accuracies. Nonetheless, these gains might be at the expense of increasing the number of required original function evaluations and thus lowering the speed of convergence to a promising solution [*Sobester et al.*, 2005]. In some cases when the original function is highly computationally expensive to evaluate, the maximum possible number of original function evaluations will be very limited (sometimes as small as 100-200). As such, practitioners have to be satisfied with adequate solutions that might not be very close to the global optimum. There is typically a trade-off between the global search (exploration) capability and the efficiency of a metamodel-enabled optimizer when computational budget is limited, especially in higher dimensional problems where the size of the feasible space is very large. In such cases, the adaptive-recursive framework might be more favourable as it may find an adequate local optimum within fewer original function evaluations than the approximation uncertainty based framework. This statement is consistent with *Regis and Shoemaker*'s conclusion [2007b], after evaluating different metamodel-enabled optimizers involving approximation uncertainty, that "… more emphasis on local search is important when dealing with a very limited number of function evaluations on higher dimensional problems." In this regard, *Regis and Shoemaker* [2007a] propose strategies to control the local search ability of two metamodel-enabled optimizers with the approximation uncertainty based framework.

Is Initial DoE required?

As demonstrated in Section 3.2.2, most metamodel-enabled optimizers developed in the literature start with formal DoEs to generate an initial set of design sites that is uniformly distributed in the explanatory variable space. A well-distributed initial set helps the metamodel better represent the underlying (original) function. Nonetheless, there are metamodelling studies not involving formal DoEs. There are studies that only use previously evaluated points typically from previous optimization attempts to initially develop the metamodel. Furthermore, as pointed out in Section 3.2.4, the studies that follow the metamodel-embedded evolution framework typically use the points evaluated in the early generations of the evolutionary algorithm to develop the first metamodels.

We believe that an initial DoE is in fact required and a sufficiently large, well-distributed initial set of design sites to develop the metamodel is a key factor to success of a metamodelling practice. As demonstrated in Section 3.2.4, metamodel-enabled optimizers can be easily deceived by metamodels fitted to poorly distributed design sites. Typically, the metamodel fitted on a set of points collected in a previous optimization attempt would be biased towards the already explored regions of the feasible space (probably containing local optima) and could be quite misleading; therefore, the unexplored regions would likely remain unexplored when optimizing on such metamodels [*Jin et al.*, 2002; *Regis and Shoemaker*, 2007b; *Yan and Minsker*, 2006].

In an evolutionary algorithm the initial population is usually uniformly distributed, but the individuals in the following generations are conditioned to the individuals in the previous generations. As such, in the metamodel-embedded evolution framework, the initial set of design sites to develop the metamodel, which is a collection of points evaluated in the initial and first few generations, may not be adequately distributed, and therefore the resulting metamodel may not have adequate global accuracy. In such a case, the metamodel that is only accurate in small regions might be completely misleading in the remaining parts that may contain the global optimum [*Broad et al.*, 2005].

However, if a sufficiently large subset of well-distributed points exists in the initial set of design sites (formed by the first few generations in an evolutionary algorithm), the subset may act as if it is from a formal DoE.

Size of Initial DoE

The optimal size of an initial set of design sites is highly dependent on the shape and complexity of the original response surface as well as the computational budget available. The term *'optimal'* here reflects the fact that, for a given original response function, increasing the number of initial design sites would enhance the accuracy of fit (a positive effect), however, after some point (which is the optimum) this enhancement would be at the expense of unnecessarily increasing the computational budget having to be initially spent on DoEs while it could have been spent more effectively in the next steps (a negative effect). The framework through which the metamodel is used (see Section 3.2.4 for different frameworks) is also a factor affecting the optimal size of initial DoEs; for example, smaller initial DoEs may suffice when the metamodel-enabled optimizer puts more emphasis on approximation error (global accuracy). The optimal size also varies for different function approximation techniques based on their level of flexibility and conformability.

In practice, determination of the optimal size for a particular problem may only be possible through extensive numerical experiments. The only prior knowledge is usually the minimum limit on the number of design sites, which is mathematically required to use a particular function approximation technique. There are some suggestions in the metamodelling literature on the size of initial DoEs when the approximation techniques are kriging and RBFs as summarized in the following. *Jones et al.* [1998] find that the approximate number of initial design sites required, p, is:

$$p = 10D \tag{3-5}$$

where D is the number of dimensions of the explanatory variable space. *Gutmann* [2001] uses a 2-level full factorial design, which samples the corners of the variable space, and as such the number of initial design sites used is:

$$p = 2^D \tag{3-6}$$

which becomes very large when the number of dimensions is more than a few. *Regis and Shoemaker* [2007b], based on the fact that kriging and RBFs with linear polynomials need at least D+1 design sites to fit, suggest that:

$p = 2(D+1)$ (3-7)

Razavi et al. [2012a] (also in Chapter 4) relate the proper size of the initial DoE to the available computational budget for optimization and suggest the following equation for kriging and RBFs with linear polynomials:

$p = \max\,[2(D+1), 0.1n]$ (3-8)

where n is total number of original function evaluations (which typically accounts for almost all available computational budget) to be evaluated during the optimization. For relatively small n values, Equation 3-8 is equivalent to Equation 3-7, but when n becomes larger, in order to design a more detailed metamodel with a better global accuracy, $0.1n$ is used as the size of the initial DoE.

Sobester et al. [2005] conduct extensive numerical experiments with their proposed metamodel-enabled optimizer based on EGO on multiple test functions to study the effect of the size of initial DoEs on the algorithm performance. They suggest that:

$p = 0.35n$ (3-9)

Sobester et al. [2005] also note that the size of initial DoEs from Equation 3-9 is an upper bound (safe choice) suitable for very deceptive, highly multi-modal functions, and for simpler functions, smaller initial DoEs may be more appropriate. They also demonstrate that if the size of the initial DoE exceeds $0.60n$, the metamodel-enabled algorithm becomes inefficient. Overall, as the size of initial DoEs (and the total number of function evaluations) cannot typically be large when the original function is computationally expensive, there is no guarantee that the initial design sites are adequately well distributed to effectively represent the shape of the underlying function (e.g., estimate locations of the regions of attraction), particularly when it is very deceptive.

Metamodel Refitting Frequency

All algorithms utilizing metamodels except those using the basic sequential framework, aim to evolve the metamodels over time by refitting them over the newly evaluated points (the growing set of design sites). The ideal strategy is to refit the metamodel after each new original function evaluation; this strategy is the basis of the adaptive-recursive and approximation uncertainty based frameworks (see Section 3.2.4). Metamodel-enabled optimizers with the metamodel-embedded evolution framework do not fundamentally need to refit the metamodel frequently (e.g., after each original function evaluation), although the higher the frequency, the more accurate the metamodel, and therefore, the better algorithm performance. Generally, the computational time required for refitting a metamodel (mostly non-linearly) increases with an increase in the size of the set of design sites. The type of the function approximation technique used to build the metamodel is also a main factor in determining the appropriate refitting frequency. As such, the computational time required for the metamodel refitting substantially varies for different types of function approximation techniques and different data sizes and may become computationally demanding and even sometimes prohibitively long. Neural networks may suffer the most in this regard, as the neural network training process is typically computationally demanding relative to other alternatives even for small sets of design sites. Kriging refitting may also become computationally demanding for large numbers (more than a few hundreds) of design sites [*Gano et al.*, 2006; *Razavi et al.*, 2012a]. The maximum likelihood estimation methodology for correlation parameter tuning is the main computational effort in the kriging (re)fitting procedure. Chapter 4 (also in *Razavi et al.* [2012a]) proposes a two-level strategy for refitting ANNs and kriging in which the first level (fast but not very accurate) is performed after each original function evaluation, but the frequency of performing the second level (complete refitting, computationally demanding but more accurate) is reduced through a function representing the complexity of the fitting problem as the number of design sites becomes larger. Refitting polynomials and RBFs with no correlation parameters is very fast even for moderately large sets (i.e., about 1000) of design sites. SVMs, as explained in Section 3.2.3, also have two or more parameters that are determined by trial-and-error or direct optimization and as such their refitting might be time consuming. The appropriate metamodel refitting frequency for a given problem is also a function of the computational demand of the original computationally expensive model as the computational budget required for metamodel refitting may sometimes be negligible and easily justified when compared to the computational demands of the original computationally expensive models.

Optimization Constraint Function Surrogates

In the water resources modelling literature, as well as the scientific literature in general, surrogate modelling has been used the most in an optimization context where the surrogate of a computationally intensive simulation model is used to approximate either the objective function or the constraints or both. The discussions in this paper mainly apply to surrogate modelling when emulating the objective functions and also when constraints are included in the objective function through penalty function approaches. When a binding constraint is approximated using a surrogate model, the approximation accuracy is highly important as it determines the feasibility/infeasibility of a solution.

A number of the optimization studies in Table 3-1 [e.g., *Broad et al.*, 2005; *Broad et al.*, 2010; *Kourakos and Mantoglou*, 2009; *Yan and Minsker*, 2006; *Yan and Minsker*, 2011] apply surrogate models to approximate constraint functions and these constraint functions are built into the overall objective function via penalty functions. Overlooking the importance of constraint satisfaction and thus failing to take special precautions to ensure the metamodel-enabled optimizer yields a feasible solution could compromise the entire metamodelling procedure [*Broad et al.*, 2005]. As a result, *Broad et al.* [2005] demonstrate an insightful three stage approach to deal with constraint function inaccuracies and part of this approach simply involves archiving the good quality solutions found in the course of optimization with the surrogate and then evaluating a set of these with the original model after optimization if it turns out that the final solution is infeasible. They also note the importance of training the surrogate model on both feasible and infeasible design sites. *Yan and Minsker* [2011] report for their ANN surrogate model of constraints that their penalty function parameters were determined by trial and error experiment. Although such a trial and error approach to penalty function parameters can be difficult to avoid, such experimentation with a metamodel-enabled optimizer will present incredible computational challenges. There are also different approaches in the broader research community to more accurately handle constraints with surrogates [e.g., *Kim and Lee*, 2010; *Lee et al.*, 2007; *Picheny et al.*, 2008; *Viana et al.*, 2010]. The paper by *Viana et al.* [2010] nicely overviews these general approaches (designing conservative surrogates and adaptively improving surrogate accuracy near the boundary between feasible and infeasible solutions).

Multi-objective Optimization

Surrogate models have been used in combination with a variety of multi-objective optimization algorithms to approximate the true Pareto-front within limited original model evaluations. Example metamodel-enabled multi-objective optimization algorithms are formed by 1- fitting response surface surrogate models to only one computationally expensive objective (or constraint) when other objectives are fast to run [e.g., *Behzadian et al.*, 2009], 2- aggregating multiple objectives into one response function (e.g., by a weighting scheme) to be approximated by a single response surface surrogate [e.g., *di Pierro et al.*, 2009; *Knowles*, 2006; *Zhang et al.*, 2010], and 3- using multiple surrogate models for multiple objectives [e.g., *Bau and Mayer,* 2006; *Keane,* 2006; *Li et al.*, 2008; *Ponweiser et al.*, 2008]. The design considerations/limitations of surrogate modelling in single-objective optimization also typically apply to all metamodel-enabled multi-objective optimization algorithms; however, the algorithms following form 3 have additional considerations or limitations discussed below.

The use of multiple surrogate models for multiple objectives would have the potential to increase the problems with inaccuracy and would definitely increase metamodelling time. The multi-objective optimization algorithms that utilize multiple surrogates commonly assume that the approximation errors (uncertainties) of these multiple surrogates are independent (no correlation) despite the fact that the objectives are typically conflicting [*Wagner et al.*, 2010]. These considerations practically limit the metamodel-enabled optimization algorithms in applicability to problems with only a small number of objectives. The issue of multiple correlated outputs being approximated by surrogate models that is discussed in Section 3.2.6 becomes relevant for metamodel-enabled multi-objective optimizers that approximate two or more objectives. Recent research by *Bautista* [2009] and *Svenson* [2011] address the dependencies between multiple objective functions when these functions are emulated with multivariate Gaussian processes.

The metamodel-enabled frameworks outlined in Section 3.2.4 are all applicable to multi-objective optimization (see Table 3-1 for example applications). When using the basic sequential framework, at the end of Step 3, all approximate tradeoff solutions should be evaluated with the original computationally expensive objective functions to determine which of these solutions are actually non-dominated (i.e., still tradeoff solutions). The EGO (Efficient Global Optimization) single-objective optimization algorithm [*Jones et al*., 1998] under the approximation uncertainty based framework, explained in Section 3.2.4, has stimulated a great deal of research to extend the expected improvement concept to multi-objective optimization [*Bautista*, 2009; *Ginsbourger et al.*, 2010; *Jeong et al.*, 2005; *Keane*, 2006; *Knowles*, 2006; *Ponweiser et al.*, 2008; *Svenson*, 2011; *Wagner et al.*, 2010; *Zhang et al.*, 2010]. ParEGO [*Knowles*, 2006], which utilizes EGO to optimize a single aggregated function (weighted sum with varying weights) of all the objective functions, and SMS-EGO [*Ponweiser et al.*, 2008], which develops multiple surrogates for multiple objectives, are among the most popular algorithms extending EGO to multi-objective optimization. *Li et al.* [2008] propose an approximation uncertainty based approach independent of EGO to account for uncertainties of multiple surrogate models for multi-objective optimization.

3.2.6 Limitations and Considerations of Response Surface Surrogates

In addition to the considerations listed in Section 3.2.5 that are specific to the design of response surface surrogate-enabled search frameworks, there are more general limitations and considerations that are relevant in any modelling analysis utilizing response surface surrogates. These limitations and considerations are discussed in the following.

High-dimensional Problems

Dimensionality is a major factor affecting the suitability of response surface surrogate modelling. Response surface surrogate modelling becomes less attractive or even infeasible when the number of explanatory variables is large. In such problems, the primary issue is that the minimum number of design sites required to develop some function approximation models can be excessively large. For example, to determine the coefficients of a second-order polynomial in a D-dimensional input space, at least $p=(D+1)(D+2)/2$ design sites are required in a 25-variable input space, at least 351 design sites are required. *Koch et al.* [1999] demonstrate that in order to obtain a reasonably accurate second-order polynomial, the minimum number of design sites may not be sufficient and suggest that $4.5p$ design sites (1580 when D=25) are necessary, which might be well beyond the computational budget available when dealing with computationally expensive models. Note that this curse of dimensionality problem exists in all other function approximation models that are augmented by second-order polynomials (e.g., RBFs and kriging used in conjunction with second-order polynomials).

Most importantly, high-dimensional problems have an extremely large search space. As such, the number of design sites required to reasonably cover the space becomes extremely large for higher number of variables. As an example, *O'Hagan* [2006] notes that 200 design sites in a 25-D space yield a very sparse coverage, but the same figure can result in a quite dense, adequate coverage for metamodelling in a 5-D space. As a result, the number of explanatory variables (decision variables, DVs, in optimization problems) in metamodel-enabled frameworks is typically not large. Among the metamodelling studies listed in Table 3-1, more than 65% of the metamodel applications are on functions having less than ten decision variables, and more than 85% have less than 20. *Behzadian et al.* [2009] and *Broad et al.* [2010] are the only studies reporting successful applications of metamodelling in relatively large-sized problems (with 50 and 49 decision variables, respectively). Notably in both studies, the number of original model evaluations are very large (>>10,000), and ANNs are used as metamodels to fit the very large sets of design sites. However, *di Pierro et al.*

[2009] report an unsuccessful application of a metamodel-enabled optimizer [ParEGO by *Knowles*, 2006] on problems having 34 and 632 DVs – the other optimizer they used (LEMMO by *Jourdan et al.*, 2006) considerably outperformed ParEGO on both problems. They point out that they could not improve the final solution quality of ParEGO even with increasing the total number of original function evaluations.

Shan and Wang [2010a] survey existing strategies to tackling the problems associated with high-dimensional problems in optimization. These strategies, which are also typically applicable to metamodel-enabled optimization, include: 1- screening aiming at identifying and removing less important decision variables [e.g., *Ratto et al.*, 2007; *Young and Ratto*, 2009], 2- decomposition aiming to decompose the original problem into a set of smaller scale sub-problems [*Shan and Wang*, 2010a], and 3- space reduction being concerned with shrinking the feasible decision variable space by reducing the variable ranges to only focus on more attractive regions in optimization [*Shan and Wang*, 2004; *Shin and Grandhi*, 2001]. Notably, none of the above strategies is a complete remedy to the issues arising in metamodelling in the context of high-dimensional problems, and as outlined by *Shan and Wang* [2010a], each has its own limitations in applicability and usefulness. *Shan and Wang* [2010b] develop a new function approximation model that is computationally efficient for larger number of decision variables.

Exact Emulators versus Inexact Emulators

A question that metamodel users need to address in any metamodelling practice is whether an exact fit (i.e., exact emulator) to the set of design sites or an approximate fit (i.e., inexact emulator), possibly with smoothing capabilities, is required. Exact emulation, also referred to as *interpolation* in numerical analysis, aims to construct a response surface surrogate representing the underlying function that goes through all design sites (i.e., exactly predicts all design sites). Kriging for computer experiments, RBFs, and Gaussian emulator machines [*O'Hagan*, 2006] are examples of exact emulators. Unlike exact emulators, there are emulation techniques that are inexact in that they produce a varying bias (deviations from the true values that are sometimes unpredictable) at different design sites. Polynomials, SVMs, MARS (multivariate adaptive regression splines), and ANNs are example inexact emulators generating *non-interpolating* emulator of the underlying function.

Under certain (usually impractical) circumstances, inexact emulators may turn to exact emulators. For example, a polynomial can exactly reproduce all design sites when the degree of freedom of the polynomial regression is zero – in case where there are as many coefficients in the polynomial as there are design sites. SVMs can also behave as if they are (almost) exact emulators when the radius of the ε-insensitive tube is set very close to zero and the weight of the regularization term is set very small so that it becomes non-dominant (see Section 3.2.3 for the details of SVM parameters). It has been proven that single-hidden-layer ANNs are also able to exactly fit *n* design sites provided that there are *n*-1 neurons in the hidden layer [*Tamura and Tateishi*, 1997]. ANNs with two hidden layers having (*n*/2)+3 hidden neurons are also capable of acting as almost exact emulators [*Tamura and Tateishi*, 1997]. Nevertheless, neither SVMs nor ANNs have been developed to apply as exact emulators and such applications would be impractical.

SVMs have been fundamentally developed for inexact emulation with strong and direct smoothing capabilities. Although ANNs are inexact emulators, their smoothing properties are usually unclear to the user and very hard to manipulate [*Razavi and Tolson*, 2011]. Kriging with the so-called “Nugget effect” [*Cressie*, 1993] is also an inexact emulation technique with smoothing capabilities producing a statistics-based bias at design sites. Any smoothing capability usually has an associated tuning parameter that controls the extent of smoothing.

There are two general types of problems involving function approximation models: physical experiments and computer experiments. There may exist substantial random errors in physical experiments due to different error sources, whereas computer simulation models are usually deterministic (noise-free), which means observations generated by a computer model experiment with the same set of inputs are identical. The inexact emulators are more suitable for physical experiments than computer experiments as the usual objective is to have an approximation that is insensitive (or less sensitive) to noise. An example application where inexact emulation is recommended is data-driven hydrologic and rainfall-runoff modelling as e.g., neural networks have been extensively used in this context. Conversely, exact emulators are usually more advisable when approximating the deterministic response of a computer model.

Figure 3-6 presents two real example experiments with inexact emulators to show how the non-interpolating behaviour can be quite misleading in optimization especially in the vicinity of regions of attraction. Figure 3-6a shows a case where the set of design sites is relatively well distributed and includes a point very close to a local optimum, however, the quadratic polynomial fitted on this set is quite misleading and returns a point (surrogate function minimizer) on the plateau while ignoring the already found local region of attraction; evaluating the surrogate function minimizer and re-fitting would not noticeably change the polynomial. Figure 3-6b demonstrates a similar case where a single-hidden-layer neural network with 5 hidden neurons is trained to be used as the surrogate. In this case, the set of design sites is intentionally very well distributed such that there are design sites located at both regions of attraction (one global and one local). As can be seen, in our experiment with this neural network, the local region of attraction (local mode on the left) is easily ignored despite the fact that there is a design site very close to the local minimum, and secondly the location of the global region is misinterpreted. We believe that such a misleading behaviour is not unlikely as we easily observed it in this simple experiment after a few trial-and-errors in network initialization and training. Notably, evaluating and adding the surrogate function minimizer to the set of design sites and re-fitting might not properly change the shape of the surrogate.

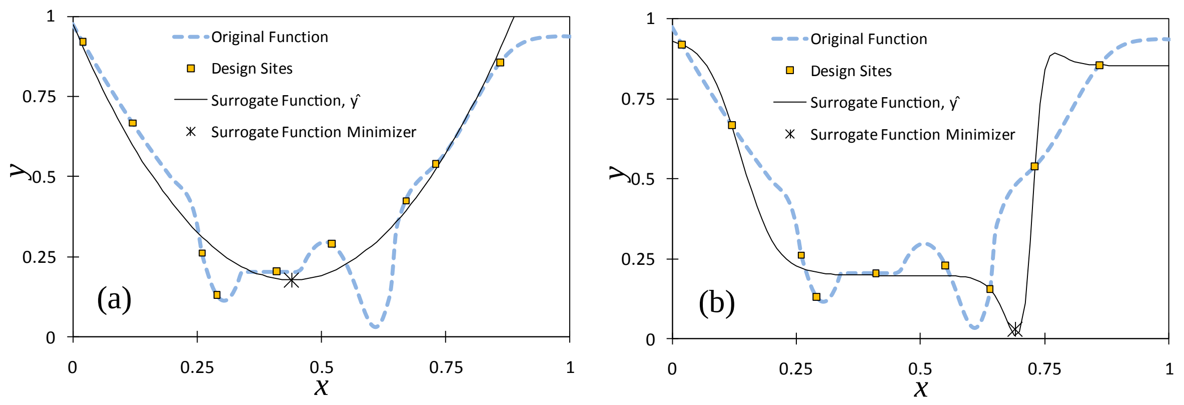


Figure 3-6. Example misleading responses of inexact emulators: (a) a quadratic polynomial, and (b) a neural network with 5 hidden neurons

In search-based metamodel-enabled analyses, it might be beneficial to have a smooth inexact emulator generating a surface passing smoothly across the design sites in the regions of the explanatory variable space with inferior quality as it may lead the search smoothly to the main regions of attraction. In contrast, inexact emulators can be very misleading in the regions of attraction (i.e., regions containing good quality local/global optima) where even a marginal superiority of candidate solutions over each other is very important and the key to continue the search. Combining the two behaviours (i.e., exact emulation and inexact emulation) and adaptively switching from one to the other may appear promising, although how to implement this adaptive switching using the common function approximation techniques is not trivial.

Exact emulation would seem to be the most appropriate way of approximating the deterministic response of computer simulation models. To our knowledge, the issues and shortcomings of inexact emulation for response surface modelling as described above have not been fully addressed in the literature, although the problems associated have been acknowledged to some extent in some publications, e.g., in *Sacks et al.* [1989], *Jones* [2001], and *Razavi et al.* [2012a]. For example, *Jones* [2001] points out that inexact emulators are unreliable because they might not sufficiently capture the shape of the deterministic underlying function. In particular, it is not clear to us from the water resources literature on surrogates for constraints, why the inexact emulation of a penalty function (which for large and sometimes continuous regions of decision space can be zero) is preferred or selected over an exact emulator. In contrast, for reliability-based optimization studies where a metamodel is fit to predict solution reliability [e.g., *Bau and Mayer,* 2006; *Yan and Minsker,* 2011], the metamodel is usually trained to design sites with a non-deterministic estimate of reliability that was generated by an approximate Monte Carlo sampling type of experiment. In this case, the choice between exact versus inexact emulation is not so clear.

Limits on Number of Design Sites

The number of design sites used for metamodel fitting can be a limiting factor affecting the suitability of a function approximation technique for a specific problem. The appropriate range (lower and upper bounds) for this number varies from one function approximation technique to another. Generally, the more design sites used for metamodel fitting, the higher the computational expense incurred in the fitting process. The computational expense associated with metamodel development and fitting should be taken into account in any metamodelling application. This expense may be limiting and directly affect the suitability of a function approximation technique for a specific problem especially when the total number of original model evaluations (and accordingly the number of design sites) in a metamodel-enabled application is relatively large.

The function approximation techniques utilizing basis (correlation) functions, such as kriging, RBFs, SVMs, and Gaussian Emulator Machine (GEM), are the most prone to the limitations arising from the large numbers of design sites. In these techniques, except for SVMs, the number of correlation functions is typically as many as the number of design sites, and as such, their structures and the computations associated for large sets of design sites become excessively large. GEM may suffer the most in this regard as the maximum number of design sites utilized in GEM applications in the literature is only 400 [*Ratto et al.*, 2007]. Kriging has also a limited applicability when the number of design sites is large, mostly because determining the kriging correlation parameters through the maximum likelihood estimation methodology can become computationally demanding for large sets. Practical numbers of design sites in kriging applications are typically less than a few thousands.

RBFs and SVMs can handle larger numbers of design sites. Least squares methods can efficiently fit RBFs even on large sets of design sites. SVMs are also capable of more efficiently handling larger numbers of design sites as the operator associated with the design site vectors in the SVM formulation is dot product [*Yu et al.*, 2006]. However, both RBFs and SVMs may involve a relatively computationally demanding parameter tuning process for the correlation parameters and the other two specific parameters of SVMs.

Unlike the correlation functions existing in GEM, kriging, RBFs and SVMs, each of which only responds to a small region in the input space close to the corresponding design site, ANNs consist of sigmoidal units each of which is associated with a hidden neuron having an active part over a large domain of the input space. As such, even for large sets of design sites, ANNs may have relatively limited numbers of hidden neurons forming reasonably sized ANN structures. There are ANN applications for very large sets of design sites; for example, *Broad et al.* [2005] use 10,000 design sites in ANN fitting and *Behzadian et al.* [2009] utilize ANNs in the adaptive-recursive framework with 590,000 and 2,098,400 original function evaluations.

As opposed to the above function approximation techniques, which consist of a number of locally active building blocks, polynomials have a single global form covering the entire input space. As such, polynomial structure does not expand as the number of design sites increases. Polynomials can be fitted very fast even over very large sets of design sites. Similar to polynomials, the structure and complexity of multivariate adaptive regression splines (MARS) is not a function of the number of design sites, and instead, it is a function of the shape and complexity of the underlying function represented by the design sites. MARS builds multiple piece-wise linear and non-linear regression models (basis functions) to emulate the underlying function in the design sites [*Friedman*, 1991], and its main computational effort is to search over a variety of combinations by first adding the basis functions to the model (forward pass) and then extensively prunes the model (backward pass) to find a parsimonious model with a satisfactory generalization ability.

The minimum number of design sites required to develop a metamodel also varies for different function approximation techniques. For a polynomial, the minimum number equals the number of coefficients existing in the polynomial. Thus, for zero-, first- and second-order polynomials the minimum numbers are 1, D+1 and (D+1)(D+2)/2, respectively, where D is the dimension of the input space. When using these minimum numbers, the polynomials would act as exact emulator (i.e., zero degree of freedom). The minimum number of design sites in kriging and RBFs depends on the polynomials by which they are augmented; as zero- and first- order polynomials are commonly used in conjunction with kriging and RBFs, the minimum number of design sites in these techniques can be very small (e.g., 11 for D=10 when augmented with first-order polynomials). GEM, SVM, and MARS also require reasonably small sets of design sites. For ANNs, although mathematically there is not any minimum limit for the number of design sites, it is commonly accepted that neural networks require relatively larger sets of design sites to be properly trained. The studies listed in Table 3-1 are consistent with this fact as the minimum number of initial design sites for neural network training in these studies is 150 [in *Zou et al.,* 2007] and the second smallest number is 300 [in *Yan and Minsker*, 2006], while this number can be as small as 20-25 when the RBFs are acting as metamodels [in *Regis and Shoemaker*, 2007b].

Validation and Over-fitting

Validation may be an important step in developing a response surface surrogate and reflects how the model performs in terms of generalizability. When a function approximation model exhibits a good fit to the design sites (i.e., zero error for exact emulators and satisfactorily small errors for inexact emulators), a validation measure is also required to ensure that the model performs consistently for unseen areas in the model input space. Cross validation strategies, such as k-fold cross validation and leave-one-out cross validation, are the commonly used means of validation of response surface surrogates [*Wang and Shan*, 2007], particularly when the set of design sites is not large. The importance of validation differs for different approximation techniques. For example, the process of developing the polynomials, RBFs, or kriging approximation models is less dependent on validation as there are studies utilizing them without conducting a validation step; whereas, validation is an inseparable step in developing SVMs and ANNs. *Bastos and O'Hagan* [2009] claim that there has been little research on validating emulators before using them (i.e., in a surrogate-enabled analysis framework). While this statement is accurate for some emulators (in particular Gaussian process emulators studied in *Bastos and O'Hagan* [2009]), it is much less accurate when emulators such as SVMs and ANNs are considered. The studies in Table 3-1 that do not utilize SVMs or ANNs as response surface surrogates, also show little in the way of metamodel validation (although most do attempt to demonstrate the utility of the overall surrogate-enabled analysis). *Bastos and O'Hagan* [2009] propose some diagnostics to validate Gaussian process emulators.

Over-fitting (over-training) degrades the generalization ability of approximation models. All approximation models are prone to over-fitting. In statistics, over-fitting is usually described as when the model fits the noise existing in the data dominantly rather than the underlying function. However, as discussed in Chapter 4 (also in *Razavi et al*. [2012a]), surrogate models fitted on noise-free data (e.g., data generated from deterministic computer models) are also prone to over-fitting, because there is another factor affecting the risk of over-fitting, which is the *conformability* of the model structure with the shape of the available data. Over-fitting due to conformability is more likely when the approximation model has a large degree of freedom (is over-parameterized) compared to the amount of available data. Curve-fitting (regression analysis) practices are typically less prone to the negative effects of this factor, especially for low orders, because they have a global pre-specified model form covering the entire input variable space. But in highly flexible approximation models, including ANNs, the problem associated with the conformability factor can be substantial.

Neural networks are highly prone to the risk of over-fitting. However, not all of the studies involving neural networks listed in Table 3-1 pay attention to over-fitting/generalizability. Seven studies listed in Table 3-1 [*Broad et al.*, 2005; *Broad et al.*, 2010; *Johnson and Rogers*, 2000; *Khu and Werner*, 2003; *Razavi et al.*, 2012a; *Zou et al.*, 2007; 2009] apply the early stopping approach [*Beale et al.*, 2010] to avoid over-training, and one study [*Kourakos and Mantoglou*, 2009] applies the Bayesian regularization procedure [*Foresee and Hagan*, 1997]. The main problem with early stopping is that the available design sites have to be split into a training set, a testing sets, and sometimes a validation set resulting in fewer data available to train ANNs. In contrast, the Bayesian regularization procedure does not require this splitting of design sites – all can be used for training. *Razavi and Tolson* [2011] recently proposed a neural network regularization measure that also does not require a testing/validation set.

Over-fitting due to the conformability factor may also occur in the function approximation techniques that are based on basis functions such as kriging and RBFs. However, the risk and extent of over-fitting in kriging and RBFs is typically less compared to ANNs. The risk of over-fitting is higher when there are very few design sites relative to the number of kriging and RBF parameters to be tuned. Note that, for example, a kriging model with Gaussian correlation functions has *D* correlation function parameters each of which is associated with one dimension in the *D*-dimensional input space and an RBF model with thin-plate splines has no correlation parameter to tune; as such, the number of parameters in these approximation models is typically small compared to the number of available design sites. As over-fitting in kriging is not a major challenge, it has not been directly addressed in most kriging studies. To mitigate the possible over-fitting problem in kriging, *Welch et al*. [1992] propose to initially keep all the correlation function parameters the same in all input space dimensions in the maximum likelihood estimation process, and then relax them one-by-one to identify the ones resulting in higher increase in the likelihood function and only let them be different.

Emulating Multiple Outputs or Multiple Functions

The literature reviewed in Table 3-1 shows that even though the vast majority of response surface surrogate studies involve a simulation model with temporally and spatially varying outputs, the required number of model outputs or number of output functions (e.g., calibration error metrics) to approximate with surrogates is typically limited to only handful of outputs/functions (often just one). Recently, Bayesian emulation techniques have appeared that are tailored to approximate a time series of output (e.g., emulating a dynamic simulator in *Conti and O'Hagan* [2010]). According to *Conti and O'Hagan* [2010], there are three approaches to emulating dynamic simulators: 1- multi-output emulators that are unique because it accounts for correlations among outputs 2- multiple single-output emulators and 3- time input emulators that uses time as an auxiliary input. They conclude that the multiple single-output emulator approach is inappropriate because of its failure to account for temporal correlations and believe that multi-output emulators should eventually lead to the successful emulation of time series outputs. *Fricker et al.* [2010] also propose multi-output emulators considering correlation between multiple outputs based on multivariate Gaussian processes.

In terms of the common function approximation techniques in Table 3-1, since ANNs have the ability to predict multiple outputs simultaneously, ANNs are a type of multi-output emulators. None of the other function approximation techniques reviewed in detail in Section 2.3.1.-2.3.4 can directly act as multi-output emulators. Thus, a single ANN model of multiple correlated outputs should conceptually be able to account for these correlations among outputs. Based on the work by *Conti* *and O'Hagan* [2010], we believe that the ability to account for correlations among outputs that are significantly correlated (even multiple output functions such as two model calibration objective functions) in the response surface surrogate should conceptually lead to increased surrogate accuracy. However, there are multiple studies demonstrating the need for multiple ANN surrogates to model multiple outputs (e.g., *Kourakos and Mantangoglou* [2009], *Yan and Minsker* [2011], and *Broad et al.* [2005]). *Yan and Minsker* [2011] approximate six outputs with three independent ANN surrogate models while *Broad et al*. [2005] model each output of interest with independent ANNs. *Kourakos and Mantangoglou* [2009] utilize ANNs to approximate 34 outputs and they explain how their single ANN to model all outputs would lead to a practically infeasible ANN training procedure as nearly 2400 ANN parameters were to be specified. Instead they built and trained 34 modular sub-networks to circumvent this computational bottleneck in ANN training, assuming that the correlations between the outputs are negligible (justified based on the physics of their case study). Further studies could investigate scenarios where multiple-output ANNs are beneficial and to determine all the reasons they can fail relative to multiple single output ANNs.

3.3 Lower-Fidelity Physically-based Surrogates

In contrast to response surface surrogates, which are *data-driven* techniques for approximating the response surface of high-fidelity (original) models based on a limited number of original model evaluations, lower-fidelity physically-based surrogates are essentially cheaper-to-run alternative simulation models that are less faithful to the system of interest. For any real-world system, there may exist several simulation models with different levels of fidelity (accuracy). A *high-fidelity* model refers to the most accurate and as such the most desirable model available to users. As the high-fidelity models may typically be computationally intensive, there are frameworks concerned with efficiently utilizing the high-fidelity models in conjunction with lower-fidelity models (as surrogates of high-fidelity models) to enhance the overall computational efficiency; these surrogate modelling frameworks when applied in the field of optimization are also referred to as "multi-fidelity" or "variable-fidelity" optimization [*Forrester et al.*, 2007; *Gano et al.*, 2006; *Leary et al.*, 2003; *Madsen and Langthjem*, 2001; *Sun et al.*, 2010]. In some publications, low-fidelity models are called "coarse" models, and high-fidelity models are called "fine" models [e.g., in *Bandler et al.*, 1994]. As a simple example, a numerical model with very small numerical time steps may be deemed a high-fidelity model and its corresponding low-fidelity model may be one with larger numerical time steps. In this paper, we often refer to “lower-fidelity physically-based surrogates” as “lower-fidelity surrogates” for simplicity.

Lower-fidelity surrogates have two immediate advantages over the response surface surrogates: 1- they are expected to better emulate the unexplored regions in the explanatory variable (input) space (i.e., regions far from the previously evaluated points with the high-fidelity model) and as such perform more reliably in extrapolation, and 2- they avoid or minimize the problems associated with high-dimensional problems (see section 3.2.6), as they use domain-specific knowledge. There is a main assumption behind any lower-fidelity surrogate modelling practice: high-fidelity and low-fidelity models share the basic features and are correlated in some way [*Kennedy and O'Hagan*, 2000]. As such, the response of the low-fidelity model for a given input vector is expected to be reasonably close to the response of the high-fidelity model for the corresponding input vector in the high-fidelity model input space. This closeness enables the lower-fidelity model to relatively reliably predict the performance of the high-fidelity model in unexplored regions in the variable space. If this assumption is violated, the surrogate modelling framework would not work or the gains would be minimal.

There are multiple strategies to reduce the number of expensive original model evaluations when a lower-fidelity model is available. The immediate strategies include using the lower-fidelity model first to reduce the variables’ feasible space and/or to identify unimportant variables to reduce the problem dimensionality [*Madsen and Langthjem*, 2001]. Most studies utilizing lower-fidelity models are concerned with developing optimization strategies where low- and high-fidelity models are adaptively chosen to be evaluated in the course of optimization [*Forrester et al.*, 2007; *Gano et al.*, 2006; *Huang et al.*, 2006; *Leary et al.*, 2003; *Viana et al.*, 2009]. There are also lower-fidelity surrogate applications in uncertainty analysis [*Allaire*, 2009; *Kennedy and O'Hagan*, 2000].

Lower-fidelity surrogate modelling has only very recently started to gain popularity in the water resources literature. Although it is a well-established area of research in the broader research community, formal terminologies and common methods available for lower-fidelity surrogate modelling in other disciplines seem to be largely unused in water resources literature. This section first reviews and categorizes the research efforts for lower-fidelity surrogate modelling in the broader research community and then reports the research efforts accomplished in water resources literature.

3.3.1 Types of Lower-Fidelity Physically-based Models

Depending on the original model type, at least three different general classes of strategies may be used to yield lower-fidelity models. In the first class of strategies, the lower-fidelity models are fundamentally the same as the original models but with reduced numerical accuracy. For example, in numerical simulation models of partial differential equations, a lower-fidelity model can be a variation of the original model but with larger (coarser) spatial/temporal grid size [*Leary et al.*, 2003; *Madsen and Langthjem*, 2001; *Thokala and Martins*, 2007]. Finite element models with simpler basis functions can also be a low-fidelity model of an original model involving more complex basis functions. Whenever applicable, lower-fidelity models can be essentially the same as the original model but with less strict numerical convergence tolerances. *Forrester et al.* [2006] employ a partially converged CFD model as a lower-fidelity surrogate of a fully converged (original) model.

A second class of strategies to derive lower-fidelity models involves model-driven approximations of the original models using *model order reduction* (MOR) techniques [*Gugercin et al.*, 2000; *Rewienski and White*, 2006; *Willcox and Megretski*, 2005]. MOR aims to reduce the complexity of models by deriving substitute approximations of the original complex equations involved in the original model. These substitute approximations are systematically obtained by rigorous mathematical techniques without the need of knowing the underlying system.

In the third class of strategies, lower-fidelity models can be simpler models of the real-world system of interest in which some physics modeled by the high-fidelity model is ignored or approximated. Strategies such as considering simpler geometry and/or boundary conditions in the model, utilizing a lumped-parameter model in lieu of a distributed model, and utilizing a two-dimensional model instead of a three-dimensional model lie under this class. For example in fluid dynamics, numerical models solving Navier-Stokes equations are the highest-fidelity and the most expensive models, models based on Euler equations are the lower-fidelity and less expensive models, and analytical or empirical formulations are the lowest-fidelity and cheapest models [*Alexandrov and Lewis*, 2001; *Simpson et al.*, 2008; *Thokala and Martins*, 2007]. Note that for any real-world system, there may be a hierarchy of models with different levels of fidelity.

Thokala and Martins [2007] conduct multiple experiments utilizing different types of lower-fidelity models and conclude that the lower-fidelity models that share the same physical components with the original models (i.e., lower-fidelity models lying under the first and second classes) are more successful than those having different/simplified physical bases (i.e., lower-fidelity models lying under the third class), as no correction (see Section 3.3.2 for definition of correction functions) can compensate for ignoring a component of the physical characteristics of a system.

3.3.2 Lower-Fidelity Model Enabled Analysis Frameworks

Variable-Fidelity Models with Identical Variable Space

Models with different levels of fidelity may be defined over the same variable/parameter space. There are multiple frameworks selectively utilizing lower-fidelity models as substitutes of the original models to reduce the number of expensive evaluations of the original model. These frameworks have mostly arisen from the optimization context but have also applied for other purposes including uncertainty analysis. The main challenge to be addressed in these frameworks is that the response landscapes of the original and lower-fidelity models are somewhat different. Figure 3-7 presents an illustrative hypothetical example of high- and low-fidelity response landscapes in a one-dimensional space. As can be seen, there are discrepancies between the two response landscapes; the low-fidelity function under-estimates the response on the left part of the plot and over-estimates in most of the right part. Moreover, both functions have two regions of attractions (modes), but the global minimizer of the low-fidelity function coincides with the local minimizer of the high-fidelity function which is far from the global optimum.

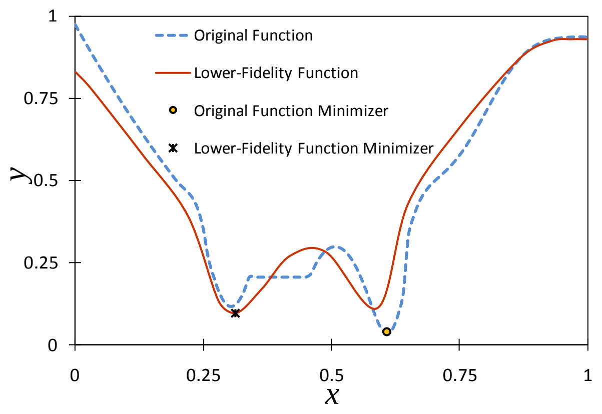


Figure 3-7. A hypothetical example of lower-fidelity functions along with the high-fidelity (original) function

There are three main independent (but sometimes complimentary) approaches to formally address the discrepancies between low- and high-fidelity models: correction functions, space-mapping, and hybrid strategies. The most common approach is to build a correction function to *correct* the response landscape of the lower-fidelity model and align it with the response landscape of the original model. This process in the multi-fidelity modelling literature is referred to as correction, tuning, scaling, or alignment. Suppose that $f_h(\boldsymbol{x})$ and $f_l(\boldsymbol{x})$ represent the response surfaces of the high- and low-fidelity models of the real-world system of interest, respectively; the two general strategies for defining a correction function are the additive approach [*Gano et al.*, 2006; *Leary et al.*, 2003; *Viana et al.*, 2009] in Equation 3-10 and the multiplicative approach [*Alexandrov and Lewis*, 2001; *Madsen and Langthjem*, 2001; *Thokala and Martins*, 2007] in Equation 3-11:

$$g_a(x) = f_h(x) - f_l(x) \tag{3-10}$$

$$g_m(x) = f_h(x)/f_l(x) \tag{3-11}$$

where $g_a(x)$ is the additive correction function directly emulating the discrepancies between the high- and low-fidelity response surfaces and $g_m(\boldsymbol{x})$ is the multiplicative correction function. As the exact form of the correction function is typically unknown for any given problem, an approximate correction function is to be built by a limited number of high- and low-fidelity model evaluations. Then, the surrogate model can be:

$$f_s(x) = f_l(x) + \hat{g}_a(x) \tag{3-12}$$

$$f_s(x) = f_l(x)\,\hat{g}_m(x) \tag{3-13}$$

where $\hat{g}_a(\boldsymbol{x})$ and $\hat{g}_m(\boldsymbol{x})$ are the approximate correction functions that are designed to correct the low-fidelity model response by offsetting (additive) and scaling (multiplicative), respectively. Notably, the multiplicative form is prone to ill-conditioning when the model response approaches zero; to avoid this ill-conditioning, a constant can be added to both the numerator and denominator of Equation (3-11). *Eldred et al.* [2004] demonstrate the superiority of the additive form over the multiplicative form across multiple test problems.

The process of developing the approximate correction function is analogous to response surface surrogate modelling; however, the correction function is supposedly less complex than typical response surface surrogates, as the response surface of a lower-fidelity model is supposed to be reasonably close to the response surface of the original model. Different approaches or tools have been proposed to develop the approximate correction function. Linear regression [*Alexandrov and Lewis*, 2001; *Madsen and Langthjem*, 2001; *Vitali et al.*, 2002] and quadratic polynomials [*Eldred et al.*, 2004; *Sun et al.*, 2010; *Viana et al.*, 2009] are two common, simple approaches to build the correction functions. More flexible function approximation models have also been used for this purpose, including kriging [*Gano et al.*, 2006] and neural networks [*Leary et al.*, 2003].

Note that limitations and considerations raised in Section 3.2.6 for response surface surrogates may also hold when using complex function approximation models to build the approximate correction functions. However, the limitations of response surface surrogates used in this context for high dimensional problems (see section 3.2.6) are not as important. This is because the correction function for a good quality lower-fidelity surrogate is only of secondary importance (it adjusts the lower-fidelity model output). As a result, less complex approximate correction functions may be more desirable in practice. Building correction functions that are correlated for multiple outputs is similarly not as important.

The general correction-function-based framework utilizing lower-fidelity surrogates is as follows: the framework, in Step 1, starts with an initial DoE to generate sample points and then evaluates them by both the original and lower-fidelity models. In Step 2, a global correction function is developed to emulate the discrepancies (errors) between the responses of the original and lower-fidelity models at the sampled points. In Step 3, a search or sampling is applied on the corrected response surface of the lower-fidelity model, identify the regions of interest in the explanatory variable space, and screen out one or multiple points. In cases where the search algorithm is for optimization, this step returns the optimal/near-optimal point (or multiple high-quality points) of the corrected response surface of the lower-fidelity model. In Step 4, the candidate points from Step 3 are evaluated by the original function. If needed, the framework goes back to Step 2 to modify the correction function and repeat the analyses in Step 3.

Trust-region approaches for optimization have also been applied in correction-function-based framework [*Alexandrov and Lewis*, 2001; *Eldred et al.*, 2004; *Robinson et al.*, 2006]. In such a framework, an initial DoE is not required and the framework may start with any (but desirably a good quality) initial solution (Step 1). The initial trust region size is also specified in this step. In Step 2, the current solution is evaluated by both original and lower-fidelity models. In Step 3, the correction function is locally fitted around the current (best) solution. In Step 4, the corrected lower-fidelity response surface is optimized within the trust region (centered at the current best solution) and the best solution found is evaluated by the original model. In Step 5, depending on how close this high-fidelity response is to the low-fidelity response, the trust region is expanded, remains the same, or is reduced. Steps 3 through 5 are repeated until convergence or stopping criteria are met. The trust-region based framework is provably convergent to a local optimum of the original model response surface if the corrected lower-fidelity response surface is at least first-order accurate at the center of the trust region [*Robinson et al.*, 2006]. *Eldred et al.* [2004] demonstrate that second-order accurate corrections can lead to more desirable convergence characteristics.

The second main approach to tackle the discrepancies between low- and high-fidelity response surfaces is the so-called *space mapping* approach. Initially introduced by *Bandler et al.* [1994] for optimization purposes, space mapping aims to locally establish a relationship between the original model variables and the lower-fidelity model variables. By definition, space mapping can be used to make use of *any* sufficiently faithful lower-fidelity model even if it is defined on a different variable space. To establish a space mapping relationship, multiple points on the original response surface and their corresponding points on the lower-fidelity response surface are required. For any given point $\boldsymbol{x}$ in the original variable space, a corresponding $\hat{\boldsymbol{x}}$ point in the lower-fidelity variable space is defined as the point where $f_l(\hat{\boldsymbol{x}})$ is equal (or reasonably close) to $f_h(\boldsymbol{x})$. To find each point required in the lower-fidelity variable space, one optimization sub-problem is to be solved with the objective of minimizing $|f_l(\hat{\boldsymbol{x}}) - f_h(\boldsymbol{x})|$ by varying $\hat{\boldsymbol{x}}$. Notably, there may exist multiple points $\hat{\boldsymbol{x}}$ having the same lower-fidelity response value equal to $f_h(\boldsymbol{x})$ leading to the failure of space mapping. Many approaches have been proposed to address this problem of non-uniqueness [*Bakr et al.*, 1999; *Bandler et al.*, 1996; *Bandler et al.*, 2004]. Once the corresponding points in the two spaces are available, different linear or non-linear functions may be used to relate the two spaces by fitting over these points [*Bandler et al.*, 2004]. Then any solution in the lower-fidelity variable space obtained in analyses with the lower-fidelity model can be mapped to the original variable space. The space mapping relationships can be updated as the algorithm progresses. As many optimization sub-problems are to be solved on the lower-fidelity model to adaptively establish/update the mapping relationships, the total number of lower-fidelity model evaluations is relatively high. As such, if the low-fidelity model is not much cheaper than the original model, space mapping would not be computationally feasible.

There are hybrid strategies following a third general approach to make use of lower-fidelity models jointly with original models to build response surface surrogates. These strategies may be used with any of the frameworks utilizing response surface surrogates (detailed in Section 3.2.4) with the main difference that there are (at least) two sets of design sites with different levels of fidelity. These two sets of design sites are used to either build a single response surface surrogate formed by the sets or two response surface surrogates representing the two sets independently. *Forrester et al.* [2007] develop a single response surface surrogate using co-kriging, which is an exact emulator on the high-fidelity design sites and an inexact emulator on the lower-fidelity design sites – such a response surface surrogate can capture the exact behaviour of the underlying function where high-fidelity design sites are available in the variable space and only extract the trends and curvatures in the unexplored regions from the cheaply available lower-fidelity design sites that are far from the high-fidelity design sites. *Leary et al.* [2003] propose a heuristic way to incorporate the lower-fidelity model response into ANN- and kriging-based response surface surrogates. *Huang et al.* [2006] develop a methodology to incorporate the data obtained by a lower-fidelity model to enhance the EGO algorithm. *Vitali et al.* [2002] and *Sun et al*. [2010] use a fourth-order polynomial and an RBF model, respectively, to develop response surface surrogates of their developed lower-fidelity models and then utilize a correction function approach to correct these response surface surrogates to be used in optimization problems with their original computationally expensive models.

Variable-Fidelity Models with Different Variable Spaces

The variable space associated with a lower-fidelity model may differ from the original model variable space. In such cases, the number of variables associated with the lower-fidelity model can be unequal to (typically less than) the number of original variables. Since their application is not trivial, multi-fidelity models when defined on different variable spaces have been less appealing in surrogate modelling literature [*Simpson et al.*, 2008]. When the spaces are different, a mapping must be established such that $\boldsymbol{x} = P(\hat{\boldsymbol{x}})$ or $\hat{\boldsymbol{x}} = Q(\boldsymbol{x})$ where $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$ are corresponding points in the original and lower-fidelity variable spaces, and P and Q are mapping functions that transform points from one space to the other. In some cases, the space mapping relationship is clear based on the physics of the problem, and the lower-fidelity variable vector is a sub-set or interpolation of the original variable vector. However, when such knowledge is not available, empirical relationships are to be derived. The space mapping approach, explained above, is a means to derive these empirical relationships

[*Bandler et al.*, 1994; *Bandler et al.*, 2004]. The main procedure in space mapping between different spaces is essentially the same as the space mapping procedure when the spaces are identical. Another mapping approach has been proposed based on proper orthogonal decomposition (also called principal component analysis) and compared against space mapping in a trust-region optimization framework [*Robinson*, 2007; *Robinson et al.*, 2006]. *Robinson et al.* [2006; 2008] incorporate the idea of correction function into space mapping to match the gradients of the lower-fidelity and original response surfaces at points of interest.

3.3.3 Related Research in Water Resources Literature

The basic idea of replacing a high-fidelity simulation model with a lower-fidelity model of the system for the purposes of optimization and uncertainty or sensitivity analysis is an intuitive concept that exists in the water resources literature. However, the vast majority of such studies have applied this idea independent of the lower-fidelity surrogate modelling studies, methods and terminology from other disciplines described in Section 3.3.2. This section reviews the research efforts arising from water resources modelling and relates them to the general methodologies for lower-fidelity surrogate modelling. The terms in quotations below represent the terminology used in the associated publications.

There are research efforts to reduce the complexity level of various water resources models to be typically used in optimization or calibration. *Ulanicki et al.* [1996] and *Maschler and Savic* [1999] develop “model reduction methods” to simplify water distribution network models by eliminating less important pipes and nodes and allocating their demands to the neighbouring nodes. *Shamir and Salomons* [2008] and *Preis et al.* [2011] utilized the model reduction method developed by *Ulanicki et al.* [1996] to create “reduced models” of water distribution networks to be used in optimization frameworks. *McPhee and Yeh* [2008] develop a “reduced model” of a groundwater model and linked it with optimization in lieu of the original model for the purpose of groundwater management – this “reduced model” was defined over a different parameter space based on empirical orthogonal functions (ordinary differential equation instead of partial differential equation). *Vermeulen et al.* [2004] and *Siade et al.* [2010] propose methods to develop “reduced models” of high-dimensional groundwater models based on proper orthogonal decomposition. *Vermeulen et al.* [2005] and *Vermeulen et al.* [2006] utilize such “reduced models” as substitutes of the original groundwater models for model inversion (calibration).

The well-established idea of using response matrices in place of full groundwater models for groundwater management [e.g., in *Reichard*, 1995], where the same methods for developing response surface surrogates are typically used, can also be classified as lower-fidelity surrogates. *Cheng et al.* [2011] also refer to the response matrix as a “reduced model” replacing a groundwater model in their optimization problem formulated for groundwater management. *Pianosi and Soncini-Sessa* [2009] use a “reduced model” of a reservoir system to improve the computational efficiency of stochastic dynamic programming for designing optimal reservoir regulation plans. *Crout et al.* [2009] develop a methodology to “reduce” water resources models by iteratively replacing model variables with constants.

Notably, in all the above studies that are for optimization purposes, reduced models (i.e., lower-fidelity models) after being developed are treated as if they are high-fidelity representations of the underlying real-world systems and fully replace the original models (i.e., high-fidelity models) in their analyses. In other words, the discrepancies between the high- and low-fidelity models are ignored.

Simplified surrogate models have been also used with different Markov Chain Monte Carlo (MCMC) frameworks for uncertainty analysis of water resources models. *Keating et al.* [2010] develop a simplified groundwater model as a “surrogate” of a computationally intensive groundwater model, defined over a different parameter space. They conduct auto-calibration and uncertainty assessment experiments on the surrogate instead of the original model and tune the algorithm parameters of an MCMC uncertainty analysis method; the tuned MCMC is then used on the original computationally expensive model. *Efendiev et al.* [2005] develop a two-stage strategy employing a groundwater model with a coarse grid, referred to as “coarse-scale model”, as a lower-fidelity surrogate of the original “fine-grid model” to speed up an MCMC experiment. In their methodology, both the surrogate and original models are defined over the same parameter space, and the surrogate is first evaluated to determine whether the original model is worth evaluating for a given solution.

There are very few surrogate modelling studies in water resources addressing the discrepancies between the response surfaces of the lower-fidelity surrogate and the original model. *Mondal et al.* [2010] develop “coarse-grid models” (also called “upscaled models”) of a (high-fidelity) groundwater model to speed up MCMC computations for uncertainty quantification; in their study, the discrepancies are recognized and quantified by a linear correction function built off-line before

MCMC experiments to avoid biased approximated posterior distribution. *Cui et al*. [2011] also develop a “reduced order model” of a “fine model” by coarsening the gird structure and employ it within a correction function framework to enhance the efficiency of an MCMC algorithm for groundwater model inversion; an adaptive local correction function is used in their study in the course of MCMC sampling to improve the accuracy.

3.4 Efficiency Gains of Surrogate-enabled Analyses

The most important question in assessing a surrogate-enabled analysis is how efficient or effective it is in comparison with other efficient alternative tools without surrogate modelling, especially because the computational efficiency achieved is the main factor motivating the research and application of surrogate modelling. Surrogate-enabled analyses typically sacrifice accuracy for efficiency as they utilize approximate models (less accurate than the original models) to more efficiently achieve the analysis objectives. As such, there is always a risk that surrogate models yield misleading results; this risk is higher when the original response landscape is complex and deceptive and is minimal for simple original response landscapes (e.g., almost negligible for smooth unimodal functions being optimized). A thorough discussion of this matter in the context of optimization is available in Chapter 4 (also in *Razavi et al.* [2012a]) where a comparative assessment framework for metamodel-enabled optimizers is developed presenting a computational budget dependent definition for the success/failure of the metamodelling strategies. The careful selection of a benchmark alternative analysis or decision-making procedure without surrogate modelling is a vital step for fair assessment of a given surrogate-enabled analysis. To be clear, a benchmark alternative analysis has available at least the same number of original model simulations as utilized in the surrogate-enabled analysis. Although *Broad et al.* [2005] note that “Metamodels should only be used where time constraints prohibit the possibility of optimizing a problem with a simulation model.”, in our view, the determination of such prohibition is not always clear cut. In a sampling context, one may take fewer samples than they would prefer, while in a search context, the algorithm can be terminated before it converges.

In an optimization or search context, the most tangible measure of efficiency gains over a benchmark alternative is *computational saving*. For a single metamodel-enabled optimization analysis (e.g., optimal design) this can be calculated as:

$$Saving = 100 \times \left[\frac{t - t_s}{t} \right] \tag{3-14}$$

where t is the computational budget or time required to reach a desired solution quality through an algorithm without surrogate modelling, and t_s is the computational budget or time a surrogate-enabled algorithm requires to reach a solution with the same quality. *Regis and Shoemaker* [2007a] present such comparative assessments in terms of efficiency by quantifying t and t_s as the numbers of original function evaluations the algorithms require to reach within 1% of the optimal value of the original function. Computational budgets may be quantified as the total CPU clock time [as in *Behzadian et al.*, 2009; *Broad et al.*, 2005; *Kourakos and Mantoglou*, 2009] or the number of original function evaluations [as in *Mugunthan and Shoemaker*, 2006; *Regis and Shoemaker*, 2007b; *Zou et al.*, 2007; 2009]. As stated in Table 3-1, 15 (out of 32) studies present quantitative information demonstrating the efficiency of the surrogate modelling strategies used. Some of these studies report the associated computational savings explicitly; in the other ones, savings are not clearly reported and we interpreted them based on the published results. According to Table 3-1, computational savings achieved through the use of surrogate models can vary significantly, ranging from 20% of CPU time in *Zhang et al.* [2009] to 97% in *Zou et al*. [2007].

Chapter 4 (*Razavi et al.* [2012a]) demonstrates that the failure of metamodel-enabled optimizers is a function of not only the degree of complexity and deceptiveness of the original landscape but also the available computational budget. In very limited computational budgets surrogate modelling is expected to be very helpful, whereas when the computational budget is not severely limited, surrogate modelling might not be as helpful, as equivalent or better solutions can be achieved by the benchmark optimizer. We believe similar findings are probable for all other types of metamodel-enabled analyses. However, the details of the comparative efficiency assessment framework for each of these other types of analysis (i.e., sensitivity analysis or reliability assessment), along with developing meaningful variants for Equation 3-14, would need to be determined. An example variation to the efficiency assessment procedure for metamodel-enabled GLUE is demonstrated in *Khu and Werner* [2003].

In any surrogate-enabled analysis, the available computational budget or time is divided between three main parts: 1- budget or time required to run the original model, 2- budget or time required to develop, run, and update the surrogate model, and 3- budget or time the analyst needs to identify and create an appropriate surrogate-enabled analysis framework. Parts 2 and 3, which are referred to as "metamodelling time" and "analyst time", respectively, should consume a small portion of the available computational budget leaving the majority for part 1. Nonetheless, metamodelling time and analyst time should ideally be taken into account when assessing the computational efficiency of a surrogate-enabled analysis. The computational and perhaps the analyst's efforts are typically higher for lower-fidelity surrogates than response surface surrogates. As such, it is difficult to imagine any comparison involving lower-fidelity surrogates on the basis of the number of original function evaluations required as is commonly done with response surface surrogate comparisons. When a developed surrogate is to be used in repeat applications the importance of the analyst time is reduced.

Any conclusion on the efficiency of a developed algorithm with surrogate models *must* be based on performing multiple replicates as any single application of such an algorithm (as with any other stochastic algorithm) is a single performance level observation from a statistical population of possible performance levels. Despite the obvious computational burden of performing multiple replicates, it is the only way to conduct valid numerical assessments and comparisons.

3.5 Summary and Final Remarks

There is a large body of literature, from different contexts and disciplines, developing and applying a wide variety of surrogate modelling strategies typically to improve the computational efficiency of sampling or search-based modelling analyses. The surrogate modelling literature was reviewed with an emphasis on research efforts in the field of water resources modelling. A set of publications including 48 references on surrogate modelling in water resources problems were analyzed and summarized in this paper and 100 other references from other disciplines were also reviewed. We overview the components involved in a surrogate-enabled modelling analysis framework and detail different framework designs.

The most important observations and available guidance on the alternative methods and surrogate modelling frameworks that have been applied to the water resources studies reviewed here are as follows:

- It is not trivial to suggest the best function approximation technique for the purpose of response surface modelling, and metamodel developers typically pick a technique based on their preference and level of familiarity as well as software availability. Function approximation techniques that are able to 1- act as exact emulators, 2- provide a measure of approximation uncertainty, and 3- efficiently and effectively handle the size of the data set (design sites) of interest, conceptually seem to be the most appealing for modelling the deterministic response of computer simulation models.

- The metamodel-enabled optimization frameworks that utilize metamodel approximation uncertainty estimates are conceptually the most robust strategies in comparison with the other three frameworks, especially for problems with highly multi-modal and deceptive original response surfaces. In our view, this framework, particularly the statistics based approach [e.g., *Jones,* 2001] is underutilized in the water resources literature as the metamodel uncertainty characterization should prove useful in Bayesian model calibration studies and traditional Monte Carlo-based reliability or uncertainty analysis.
- When evidence is available suggesting the original function is a relatively simple/unimodal function, using the basic sequential framework or adaptive-recursive framework would be the most appropriate as they would be successful and more efficient.
- Difficulties are introduced moving from unconstrained (or just box-constrained) surrogate-enabled single optimization to surrogate-enabled constrained or multi-objective optimization.
- Probably the most important limitation of surrogate modelling in applicability, especially response surface surrogate modelling, is when the number of dimensions in the problem variable space is large (successful surrogate-enabled analyses reviewed here were limited to 50 at most and typically less than 20 explanatory variables). Lower-fidelity surrogates are much less vulnerable to this limitation.
- Lower-fidelity models are conceptually more reliable in exploring the unseen regions in the explanatory variable space compared to response surface surrogates. This reliability directly relates to the level of fidelity of a surrogate model and diminishes for the surrogates with very low fidelity. As there is typically a trade-off between the level of fidelity of a model and its computational demand, the lower-fidelity model developers should create a level of fidelity that is sufficiently faithful to the original model while being efficient enough to permit the case study specific analysis required.

- Lower fidelity models have advantages when there is an interest in emulating multiple model outputs or in multi-objective optimization with two or more emulated objectives as the lower fidelity models would inherently account for output/objective function correlations.

The following remarks summarize some suggestions on future research directions, many of which are inspired from ideas not commonly used in water resources literature:

- Since the level of complexity of the original response function (which is typically unknown a priori) plays a key role in the determination of an appropriate function approximation technique, future research may be directed at developing methods to pre-analyse the original response landscapes with a very limited number of samples to measure their level of complexity. The study by *Gibbs et al*. [2011] is an example of research efforts for the response landscape analysis in the optimization context (not involving surrogate modelling).
- The strategies utilizing lower-fidelity surrogate models are relatively new and seem very promising as they circumvent many of the limitations accompanying response surface surrogates and although the lower-fidelity surrogate concept is slowly making its way into the water resources literature, there are multiple advanced strategies in the broader research community that are unseen or underutilized in the water resources literature we reviewed here (e.g., correction functions and space mapping).
- A recent review by *Vianna and Haftka* [2008] argues against applying a single surrogate model and instead suggests that multiple surrogate models should be fit and even used in conjunction with one another. The body of multi-surrogate model methods they review should prove useful in water resources applications.
- Building new/revised model analysis specific computational efficiency assessment frameworks, similar in concept to the one proposed in Chapter 4 (also in *Razavi et al*. [2012a]), for Bayesian model calibration studies, sensitivity analysis, multi-objective optimization and traditional Monte-Carlo based uncertainty analysis would help formalize metamodel-enabled methodological comparisons. In particular, such frameworks should account for the challenges noted in Section 3.4 associated with the comparison of response surface surrogates with lower-fidelity modelling.

Our final direction for future research is the most important one. Studies demonstrating a methodology for validation, or perhaps a case study specific proof of concept, of the *entire* metamodel-enabled analysis would be invaluable. In practical situations, the metamodel-enabled modelling analysis would not be repeated without a metamodel and the analyst either hopes the metamodel-enabled analysis yields helpful results (e.g., the list of most sensitive model parameters is mostly correct) or provides a better answer than they started with (e.g., the optimization solution is better than the initial solution before any metamodelling). Such complete uncertainty about the quality or accuracy of the final analysis result after such a time consuming procedure does not breed confidence – in particular given that the success or failure of the entire metamodel-enabled analysis depends on many subjective decisions, the computational budget, the case study original model characteristics, the random number generator, etc.! Imagine suggesting and defending a worldwide policy decision to combat climate change on the basis of a single metamodel-enabled analysis without rigorous validation showing the entire procedure could reliably accomplish what it was intended to do. The real question is how to build relevant case study specific examples for proof of concept - clearly, this would involve developing lower-fidelity models of the system.

Chapter 4
Development and Evaluation of Response Surface Surrogate Models

This chapter is based on the published article below with changes outlined in the following. Sections 4.1 and 4.2 were shortened (relative to the article) by removing the contents already presented in Section 3. In addition, a new case study (i.e., FEFLOW case study) was added to this Chapter (not available in the article), and minor organizational changes were made to be consistent to the body of the thesis. References are unified at the end of the thesis.

Razavi, S., B. A. Tolson, and D. H. Burn (2012), Numerical assessment of metamodelling strategies in computationally intensive optimization, *Environ. Modell. Software*, 34, 67–86.

Summary

As reported in Chapter 3, there is a sizeable body of literature developing and applying a variety of response surface surrogate modelling (also called metamodelling) strategies to various environmental and water resources related problems including environmental model calibration. Overall, this literature generally implies *metamodelling yields enhanced solution efficiency and (almost always) effectiveness of computationally intensive optimization problems*. This chapter initially develops a comparative assessment framework which presents a clear computational budget dependent definition for the success/failure of the metamodelling strategies, and then critically evaluates metamodelling strategies, through numerical experiments, against other common optimization strategies not involving metamodels. Three different metamodel-enabled optimizers involving radial basis functions, kriging, and neural networks are employed. A robust numerical assessment within different computational budget availability scenarios is conducted over four test functions commonly used in optimization as well as three real-world computationally intensive model calibration problems. Numerical results show that *metamodelling is not always an efficient and reliable approach to optimizing computationally intensive problems*. For simpler response surfaces, metamodelling can be very efficient and effective. However, in some cases, and in particular for complex response surfaces when computational budget is not very limited, metamodelling can be misleading and a hindrance, and better solutions are achieved with optimizers not involving metamodels. Results also demonstrate that neural networks are not appropriate metamodelling tools for limited computational budgets while

metamodels employing kriging and radial basis functions show comparable overall performance when the available computational budget is very limited.

4.1 Introduction and Objective

One of the most commonly used approaches to dealing with computational burdens of optimization efforts involving computationally intensive simulation models is response surface surrogate modelling (also called metamodelling) or the use of function approximation. There is a sizeable body of literature developing and applying a variety of metamodelling strategies in various environmental and water resources related problems including surface water model and groundwater model calibration. The wide and extensive application of metamodelling strategies over more than four decades in a wide variety of optimization problems from different disciplines [*Simpson et al.*, 2008] suggests that this research field is sufficiently mature so that the pros and cons of metamodelling can be fairly evaluated. These publications typically give the impression that metamodelling increases the overall computational efficiency (i.e., attaining good quality solutions while consuming less computational budget compared to when metamodelling is not applied) and almost always effectiveness (i.e., attaining better quality solutions than the solutions achieved without metamodelling) in computationally intensive optimization problems.

The main objective of this chapter is to test this common belief through a comparative assessment of metamodel-enabled optimizers against optimizers without metamodelling. This comparison is focused on serial optimization algorithms only and does not extend the comparison to parallel optimization algorithms. In general, since metamodel-enabled optimizers and optimizers without metamodelling can both be implemented to take advantage of a parallel computing network, it is expected that these findings would be similar if parallel optimization algorithms were considered. Two major factors affecting the performance of metamodel-enabled optimizers namely the shape/complexity of the original computationally expensive function being optimized and computational budget availability are directly addressed. The experiments and the detailed descriptions of the metamodels we implemented were designed in a way that they give metamodel users a clear view of metamodel characteristics, benefits and shortcomings and demonstrate the complexities and subjective decisions required by analysts building a metamodel-enabled optimizer.

The organization of this chapter is as follows. Section 4.2 raises some fundamental metamodelling considerations that metamodel-enabled optimizer users should consider before any metamodel development/application and any subsequent comparison with optimizers that do not rely on metamodelling. Section 4.3 describes three different metamodel-enabled optimizers representing different metamodelling strategies from the current literature that are adopted and implemented in this study. This section contains detailed practical information regarding the metamodel-enabled optimizers implemented in this thesis as such details seem largely unreported in the current literature. Conversely, there is only limited information in Section 4.3 on theories and metamodel equations (e.g., kriging and neural networks theories and equations), as they are available elsewhere. Section 4.4 benchmarks the optimization algorithms (without metamodelling) used in this study to develop the baseline for metamodelling assessment. Section 4.5 presents the test functions and the real-world case studies used as well as the details of experiments settings. Experimental results are reported in Section 4.6 and followed by a discussion in Section 4.7 and conclusions and final remarks in Section 4.8.

4.2 Fundamental Considerations

4.2.1 Evaluating the Effectiveness and Efficiency of Metamodelling

Based on our analysis of the metamodelling in environmental and water resources optimization literature, the lowest CPU time saving observed due to metamodelling is 21% in *Broad et al*. [2005] and the highest saving is 97% reported in *Zou et al*. [2007]. Typically, efficiency or effectiveness of a metamodel-enabled algorithm is comparatively quantified with respect to a benchmark algorithm without metamodelling. Thus, for any given metamodel-enabled algorithm, it is important to carefully choose an appropriate benchmark algorithm to make appropriate and logical conclusions.

Metamodels become attractive when the maximum possible number of original simulation model evaluations is limited. However, there are other computationally efficient optimization algorithms independent of metamodelling [e.g. *Kuzmin et al.*, 2008; *Tan et al.*, 2008; *Tolson and Shoemaker*, 2007b; *Tolson et al.*, 2009] that have been designed to work in a limited computational budget (i.e., limited number of simulation model evaluations or, equivalently, objective function evaluations, or simply function evaluations), and there also exist strategies to enhance the efficiency of some inefficient metamodel-independent optimization algorithms [*Ostfeld and Salomons*, 2005; *Razavi et al.*, 2010]. Unlike these fast algorithms, there are algorithms like standard genetic algorithms (GAs) which are robust and effective given a large number of function evaluations but are not typically designed for cases where the number of function evaluations is quite limited. As such, assessing the efficiency/effectiveness of a metamodel-enabled algorithm with respect to a single *inefficient* algorithm without metamodelling is not appropriate when more efficient alternative algorithms without metamodelling are available.

Moreover, there is substantial amount of randomness inherently involved in metamodel-enabled optimization algorithms. Any single application of such an algorithm is a single performance level observation from a population (statistical) of possible performance levels. Thus, to ensure that the findings are representative of the population of possible performance levels, empirical assessment and comparison of these algorithms *must* be based on performing multiple replicates (despite the obvious computational burden). Regrettably, there are many example studies in water resources related metamodelling literature which make conclusions based on only a single run of their developed stochastic metamodel-enabled optimization algorithm(s).

We believe that there are three possible outcomes (cases) in the evaluation of a metamodel-enabled optimizer relative to an optimizer without metamodelling for a given optimization problem (see Figure 4-1). As stated in Section 4.1, the current metamodelling literature as a whole presents results in a way that implies relative performance of metamodel-enabled optimizers compared to optimizers without metamodelling most often looks like Case A ("idealized relative metamodel performance") shown in Figure 4-1. In Case A, which is too optimistic in our view, the metamodel-enabled optimizer always outperforms (or equals) the optimizer without metamodelling for any computational budget. Conversely, from a pessimistic point of view, there may exist metamodel-enabled optimizers that result in inferior relative performance for any computational budget (Case C in Figure 4-1). If lying in Case C, the metamodel-enabled optimizer fails and its application is not justifiable. Clearly, another case must exist (i.e., Case B in Figure 4-1) in between Case A and Case C. We refer to Case B as "computational budget dependent relative metamodel performance". In Case B, as its name reflects, the relative performance of a metamodel-enabled optimizer is a function of the available computational budget (or equivalently total number of original function evaluations). The most likely scenario for Case B is that in smaller computational budgets, the metamodel-enabled optimizer outperforms the optimizer without metamodelling; while in higher computational budgets, the performance of the metamodel-enabled optimizer is inferior. Notably, "equivalence time", t^*, in Case B (see Figure 4-1) is not known *a priori*, and for any specific problem lying under Case B, determination of t^* requires multiple comparative numerical experiments. Practically, we do not consider relative performance given an infinite computational budget and as such, the computational budgets in Figure 4-1 should be viewed as a range representing relatively limited or practical computational budgets. This study aims at demonstrating the three possible relative performance cases through extensive numerical experiments with different metamodel-enabled optimizers and different optimizers without metamodelling.

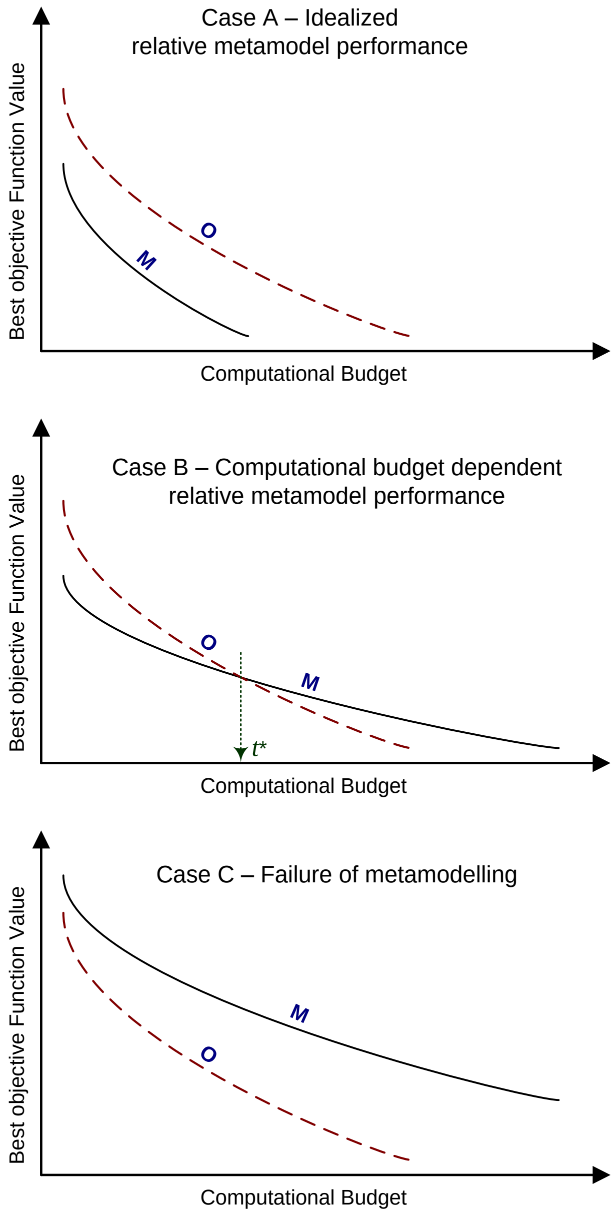


Figure 4-1. Three possible cases for relative performance of metamodel-enabled optimizers (M) compared to optimizers without metamodelling (O)

4.2.2 Metamodel-Enabled Optimizer Framework Development

Development of a successful metamodel-enabled optimizer for a given problem involves several subjective, non-trivial decisions. These decisions are important as they affect the behaviour and performance of the algorithm. The major decisions include the selection of the appropriate function approximation technique capable of acting as a metamodel and the selection of the framework through which the original model, metamodel, and the optimizer can be effectively linked. There are also a number of subjective decisions to be made for each selected metamodel-enabled optimizer as discussed in Section 4.3 for our adopted algorithms. Obviously, any metamodelling effort would be burdened by computational plus analyst time, as highlighted in the following. Design of experiment (also referred to as DoE) is another subjective component in many metamodel-enabled optimizers. DoE is focused on identifying a logical initial set of fully evaluated solutions on which to train or fit the first metamodel. DoE considerations for the metamodelling frameworks we implement are discussed later where these frameworks are introduced (Section 4.3).

Selecting a Function Approximation Technique

Metamodels approximate the response surface of a computationally intensive simulation model (i.e., original model) by fitting a simplified function over a set of previously evaluated points, commonly referred to as *design sites*, in the decision variable space. A variety of function approximation techniques have been developed and applied as metamodels (see Chapter 3: Section 3.2.3). Function approximation techniques can be classified as interpolating (exact emulator) or non-interpolating (inexact emulator). DACE and RBF models, which are interpolants, exactly predict all design sites to represent the underlying function. Polynomials and ANNs, which are non-interpolants, produce a varying bias (unpredictable in ANNs) at different design sites (note that if there are as many coefficients in a polynomial as there are design sites, it becomes an interpolant). Note that kriging with the so-called “Nugget effect” [*Cressie*, 1993] can also be a non-interpolating approximator that produces a statistics-based bias at design sites and serves as a smoother. As most environmental and water resources simulation models perform deterministic simulation (i.e., the outputs of running a model multiple times with the same input are identical), exact emulators seem more appealing. Section 3.2.6 of Chapter 3 and Section 4.3.3 further deal with the metamodelling issues related to interpolation versus smoothing.

How to Link the Metamodel, Original Model, and Optimizer

The most important part of developing metamodel-enabled optimizers is the framework through which the optimizer, metamodel, and original computationally expensive model interact. There is a large body of literature developing frameworks that combine metamodels with optimization concepts. The main focus of water resources related metamodelling literature is also on this framework development [e.g. as in *Behzadian et al.*, 2009; *Broad et al.*, 2005; *Mugunthan and Shoemaker*, 2006; *Zou et al.*, 2007; 2009]. The existing metamodelling frameworks can be classified into four main frameworks as outlined in Section 3.2.4 of Chapter 3. The adaptive-recursive and approximation uncertainty-based frameworks are probably the most commonly used metamodelling approaches in environmental and water resources context [*Bau and Mayer*, 2006; *Bliznyuk et al.*, 2008; *di Pierro et al.*, 2009; *Fen et al.*, 2009; *Mugunthan and Shoemaker*, 2006; *Mugunthan et al.*, 2005; *Regis and Shoemaker*, 2007a; b; 2009; *Zou et al.*, 2007; 2009]. Therefore, we selected two different metamodel-enabled optimization frameworks (DACE-GA and ANN-GA, see Sections 4.3.2 and 4.3.3) based on the adaptive-recursive framework and one (LMSRBF, see Section 4.3.1) based on the approximation uncertainty-based framework in our numerical comparisons.

Assessing Metamodelling Time and Analyst Time

To evaluate the true efficiency of a metamodel-enabled optimizer, besides the computational burden associated with the original or expensive simulation model evaluations, other time consuming efforts should be taken into account, namely “metamodelling time” and “analyst time”. Metamodelling time is the time needed for determining the location of initial DoE sites, metamodel fitting and refitting [which can be prohibitively long especially when the metamodel is a neural network as in *Broad et al.*, 2005], metamodel evaluations, and the search procedure on the metamodel. Metamodelling time is considered in some water resources metamodelling studies such as *Behzadian et al.* [2009], *Broad et al*. [2005], and *Kourakos and Mantoglou* [2009], and in these studies, metamodelling time is based on tracking total optimization run time as opposed to simply counting the number of original model runs required. Unfortunately, tracking (and then comparing) total optimization run times requires more care than tracking the number of original model runs as it is dependent on the computer programming language, the skill of the person writing the code and the processor on which the code is executed.

Analyst time is the human time required to develop and/or apply a metamodel-enabled optimization algorithm. In most of the water resources related metamodelling studies we are aware of, although metamodel-enabled algorithms were developed or implemented, none of these studies considered the analyst time in their comparative assessment against existing metamodel independent optimizers. Obviously, if a practitioner uses readily available software implementing a metamodel-enabled algorithm, the analyst time for such an experiment is zero (or equivalent to analyst time using an optimizer without metamodelling). Unfortunately, analyst time is not something that can be ignored currently since most water resources related metamodelling studies introducing new frameworks or example metamodel-enabled optimizers do not make corresponding software available for future researchers/users.

Figure 4-2 shows the computational budget dependent relative metamodel performance when metamodelling time and analyst time are taken into account. As can be seen, when metamodelling time and/or analyst time are considered, the equivalence time, *t**, decreases and in situations with substantial analyst time, relative metamodel performance can transfer from Case B to Case C.

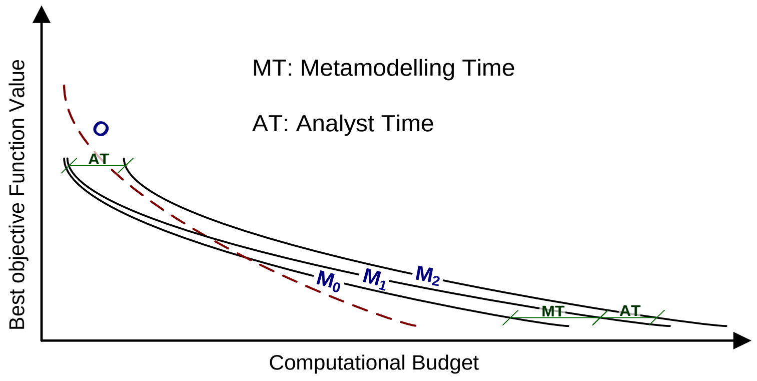


Figure 4-2. Case B relative performance of metamodel-enabled optimizers compared to optimizers without metamodelling (O) when metamodelling time and analyst time are taken into account. M_0: computational budget considered for comparison is only based on the number of original function evaluations, M_1: computational budget also includes metamodelling time, and M_2: both metamodelling time and analyst time are considered.

4.2.3 Difficulties in High-dimensional Problems

As also outlined in Chapter 3 (Section 3.2.6), metamodelling becomes less attractive or even infeasible due to the curse of dimensionality when the number of decision variables is large [*Wang* *and Shan*, 2007]. In such a problem, the primary issue is that the minimum number of design sites required to fit some metamodels can be excessively large. For example, to determine the coefficients of a second-order polynomial in a D-dimensional input space, at least $p=(D+1)(D+2)/2$ design sites are required. Note that this curse of dimensionality problem exists in all other metamodels being augmented by second-order polynomials (e.g., RBF models and kriging if used in conjunction with second-order polynomials). For example, the kriging software we utilized in this paper [*Lophaven et al.*, 2002] allows users an option to use second-order polynomials. The minimum number of design sites in RBF models and kriging augmented by first-order polynomials is $D+1$ which is not very limiting. In ANNs, there is no clear mathematical minimum number for design sites, but practically, it is commonly accepted that this number should be greater than the number of network weights.

Most importantly, high-dimensional problems have an extremely large search space. As such, the number of design sites required to reasonably cover the space becomes extremely large for a higher number of decision variables (DVs). As a result, the number of DVs reportedly tackled by metamodel-enabled optimizers is typically not large and most metamodel applications in the environmental and water resources context are on functions having less than 15-20 DVs [as in *Bau and Mayer*, 2006; *Bliznyuk et al.*, 2008; *Fen et al.*, 2009; *Khu and Werner*, 2003; *Khu et al.*, 2004; *Liong et al.*, 2001; *Mugunthan and Shoemaker,* 2006; *Mugunthan et al.*, 2005; *Ostfeld and Salomons*, 2005; *Regis and Shoemaker*, 2004; 2007a; b; 2009; *Shoemaker et al.*, 2007; *Zhang et al.*, 2009; *Zou et al.*, 2007; 2009]. Therefore, the main body of numerical experiments conducted in this study utilized case studies with only 7 to 15 DVs; some numerical experiments were also conducted with an automatic calibration case study with 56 DVs.

Screening is the most commonly used compensating solution for difficulties associated with metamodelling in high-dimensional problems. Based on the fact that models never respond strongly to all inputs, DV space is typically screened to identify and remove DVs that are less important. Various approaches to screening, especially for high-dimensional model representation, are available in the literature [e.g. *Ratto et al.*, 2007; *Young and Ratto*, 2009]. However, it can be difficult to obtain substantial reductions of dimensionality for large-scale problems [*Koch et al.*, 1999], and also any reduction in dimensionality is accompanied by a decrease in the overall accuracy of approximation unless only the absolutely irrelevant parameters (if they exist) are fixed.

4.3 Metamodel-Enabled Optimizers

We adopted three metamodel-enabled optimizers to be evaluated against the benchmark optimizers without metamodelling. Although various metamodel-enabled optimizers have been developed, software packages implementing them are not as commonly available as optimizers without metamodelling. One available metamodel-enabled optimization software in the environmental and water resources metamodelling literature (available at http://www.sju.edu/~rregis/pages/software.html) implements multistart local metric stochastic RBF (MLMSRBF) developed by Regis and Shoemaker [2007b]. MLMSRBF, described in Section 4.3.1, was employed in this study as a well-established, readily available metamodel-enabled optimizer. Two other metamodel-enabled optimizers, which are called DACE-GA and ANN-GA hereafter, were also tested in this study. Both ANN-GA and DACE-GA were implemented in almost the same GA-based metamodel-enabled optimization framework with the only difference being the function approximation techniques (ANN or DACE) and their associated design/fitting procedures. Sections 4.3.2 and 4.3.3 deal with developing/implementing DACE-GA and ANN-GA, respectively.

4.3.1 Multistart Local Metric Stochastic RBF

Multistart Local Metric Stochastic RBF (MLMSRBF) is an efficient RBF embedded optimizer that has shown superior performance over multiple existing metamodel-enabled optimizers, specifically for the problems with 8 to 15 decision variables [*Regis and Shoemaker*, 2007b]. MLMSRBF has been successfully applied in multiple studies [e.g. *Mugunthan and Shoemaker*, 2006; *Mugunthan et al.*, 2005; *Regis and Shoemaker*, 2009]. MLMSRBF implicitly considers the metamodel approximation uncertainty. The algorithm starts with a DoE and iteratively generates candidate points by perturbing the current best solution through a normal distribution with zero mean and a specified covariance matrix. MLMSRBF implements a local search algorithm in the close vicinity of the current best solution since the spread of this normal distribution is relatively small compared to the size of feasible space. To obtain a global search capability, MLMSRBF includes multiple independent restarts each initialized using a new DoE whenever it appears to have converged to a local minimum. See *Regis and Shoemaker* [2007b] for full algorithm details.

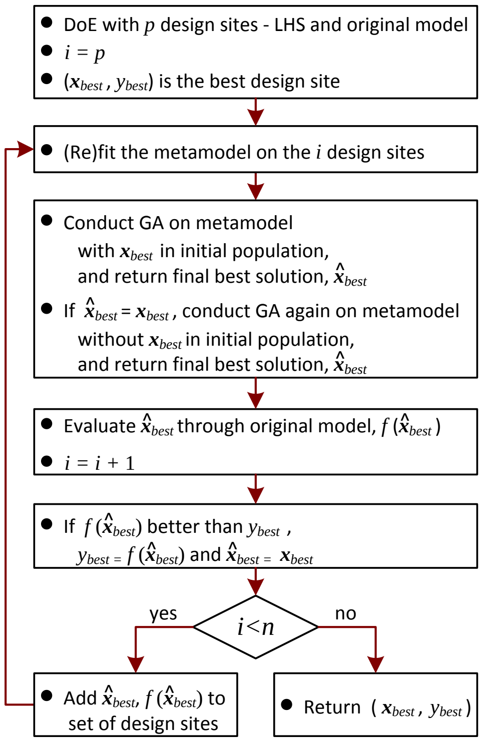
4.3.2 Design and Analysis of Computer Experiment – Genetic Algorithm (DACE-GA)

DACE-GA employs the DACE function approximation technique in conjunction with a GA. DACE has been used as a metamodel in *Bau and Mayer* [2006] for pump-and-treat optimization and in *di*

Pierro et al. [2009] for water distribution system design optimization. Although not as common in the environmental and water resources optimization literature, DACE has been widely used in other fields for approximating computer simulation models, and hence, we selected it as one of our metamodelling techniques. We selected a GA as the optimization engine to search over the approximated response surface (metamodel) because GAs are so commonly used in conjunction with metamodelling [*Broad et al.*, 2005; *di Pierro et al.*, 2009; *Fen et al.*, 2009; *Khu and Werner*, 2003; *Khu et al.*, 2004; *Ostfeld and Salomons*, 2005; *Yan and Minsker*, 2006; *Zou et al.*, 2007; 2009]. In addition, *Johnson and Rogers* [2000] report that the quality of solutions obtained through an adaptive-recursive approach based metamodel enabled optimizer is mostly controlled by metamodelling performance, not the search (optimization) algorithm (which is a GA in DACE-GA). This is also consistent with the results of our preliminary experiments. In other words, different global optimization algorithms require different computational budgets to converge to the near-global optimum, but when applied on fast-to-run metamodels, the difference, which is a small part of the total computational budget needed for metamodel enabled optimization, is negligible. The GA used in developing our metamodel-enabled optimizers is the same as the GA used as a benchmark optimization algorithm without metamodelling that is detailed in Section 4.4.1.

It is worth noting that, initially, our goal was to exactly replicate previously published implementations of the DACE-GA (and ANN-GA, explained later in Section 4.3.3) frameworks. Unfortunately, since software was unavailable and there is a shortage of required framework implementation details in the publications from various sources, exact replication was not possible. Instead, we implemented our own interpretation of the framework based on adaptive-recursive approach that shares features with other similar metamodel-enabled optimizers [e.g. *Regis and Shoemaker*, 2007a; *Zou et al.*, 2007; 2009]. Figure 4-3 shows the flowchart of our DACE-GA metamodel-enabled optimizer. Details of the metamodel, the size of the initial set of design sites, *p*, and the procedure and frequency of metamodel (re)fitting are presented in the following paragraphs.

To design the metamodelling framework and initial DoE, there is a basic question which needs to be answered first: how much extra computational budget or equivalent time should be allocated to metamodel associated efforts (i.e., metamodelling time)? Clearly, the metamodelling time allocated can be in a direct relationship with the computational time required for the original computationally expensive model evaluations. For instance, it is quite logical to allocate 5 seconds on average to metamodelling time in order to avoid an original model evaluation when the original model takes 5 minutes to run; while it seems illogical to allocate the same 5 seconds when the running time of the original model is only 10 seconds. In this study, for the two real-world computationally intensive case studies (see Section 4.5.2), we limited ourselves to use at most 5% of our available computational budget for metamodelling time. Note that since the four test functions used in this study (see Section 4.5.1) were assumed to represent the response surface of computationally intensive simulation models, we ignored the issue of metamodelling time in these problems as it was assumed negligible.



DoE with p design sites - LHS and original model
$i = p$
$(\mathbf{x}_{best}, y_{best})$ is the best design site
(Re)fit the metamodel on the i design sites
Conduct GA on metamodel with $\mathbf{x}_{best}$ in initial population, and return final best solution, $\hat{\mathbf{x}}_{best}$
If $\hat{\mathbf{x}}_{best} = \mathbf{x}_{best}$, conduct GA again on metamodel without $\mathbf{x}_{best}$ in initial population, and return final best solution, $\hat{\mathbf{x}}_{best}$
Evaluate $\hat{\mathbf{x}}_{best}$ through original model, $f(\hat{\mathbf{x}}_{best})$
$i = i + 1$
If $f(\hat{\mathbf{x}}_{best})$ better than y_{best}, $y_{best} = f(\hat{\mathbf{x}}_{best})$ and $\hat{\mathbf{x}}_{best} = \mathbf{x}_{best}$
$i<n$
yes
no
Add $\hat{\mathbf{x}}_{best}$, $f(\hat{\mathbf{x}}_{best})$ to set of design sites
Return $(\mathbf{x}_{best}, y_{best})$

n : total number of original function evaluations

Figure 4-3. Flowchart of the developed DACE-GA and ANN-GA metamodel-enabled optimizers – the term ‘metamodel’ in this flowchart interchangeably represents ANNs and DACE

In this study, a well-established, well documented implementation of DACE written by *Lophaven et al.* [2002] was utilized. It is a MATLAB add-on toolbox (available at http://www2.imm.dtu.dk/~hbn/dace/) which supports a variety of user selected correlation functions. The DACE model applied in this study utilizes a Gaussian correlation function augmented with a first order polynomial. The minimum number of design sites required to initially fit our selected DACE approximation in a *D*-dimensional space is *D*+1; while the optimal number is highly function-and-computational-budget dependent and very difficult to determine. The term “optimal” here reflects the fact that increasing the number of initial design sites would enhance the accuracy of fit (a positive effect), however, after some point (which is the optimum) this enhancement would be at the expense of unnecessarily increasing the computational budget having to be initially spent on DoE while it could have been spent more intelligently in the next steps (a negative effect). In our DACE-GA implementation, the size of the initial set of design sites was calculated as follows based on some preliminary experiments:

$$p = \max\,[2(D+1),\, 0.1n] \qquad (4\text{-}1)$$

where n is total number of original function evaluations (which typically accounts for almost all available computational budget) to be evaluated during the optimization. When n is relatively small, $2(D+1)$ initial design sites are used, which is twice as many design sites as the minimum requirement [*Regis and Shoemaker*, 2007b, recommend 2(D+1) initial design sites]. When n becomes larger, in order to design a more detailed metamodel, $0.1n$ is used as the size of the initial DoE.

Fitting/refitting the metamodel can become computationally demanding such that the frequency of refitting the metamodel must not be too high. Inversion of an $i\times i$ matrix (i=number of design sites, $p\leq i\leq n$) is the main computational effort of DACE output calculation for a given input. Therefore, assuming Gauss–Jordan elimination is applied for matrix inversion (actual inversion method is unclear in DACE software), the complexity of the (re)fitting problem can be assumed to be proportional to $O(i^3)$. Refitting of DACE in our DACE-GA is performed in two levels: the first level is to simply add the new points to the set of design sites in DACE, but the second level is to re-optimize (re-tune) the DACE hyper-parameters (correlation function parameters), over the entire set of design sites through maximum likelihood estimation. As the first level is relatively fast, even for large i values (in this study $i\leq n\leq 1000$), this level is performed whenever the algorithm goes back to the refitting step (second box in the flowchart shown in Figure 4-3). However, since the second level can be computationally demanding for large i values, in order to limit metamodelling time, the frequency of performing the second level should be decreased as i becomes larger. This decreasing trend can exactly follow the polynomial form of the complexity equation mentioned above. In our DACE-GA implementation, the second level is always performed at the DACE refitting step while $i\leq 100$, but afterwards, this level is performed less often following the complexity equation so that, for example, for the i values around 200, the second level is performed after receiving every 8 new design sites ($8=200^3/100^3$), and for the *i* values around 500, after receiving every 125 new design sites ($125=500^3/100^3$). Our two level refitting approach seems unique in comparison with previous DACE-based studies although exact details of the refitting strategies employed in previous papers are not always clearly reported.

4.3.3 Artificial Neural Network – Genetic Algorithm (ANN-GA)

ANN-GA utilizes the same framework presented in Section 4.3.2 for DACE-GA. As explained later in this section, we believe that there are multiple problems and shortcomings associated with the application of ANNs in metamodel-enabled optimizers for computationally intensive problems. Despite this view, it is hard to ignore the empirical evidence showing that ANNs are the most commonly employed metamodel in environmental and water resources optimization problems [as in *Behzadian et al.*, 2009; *Broad et al.*, 2005; *Johnson and Rogers*, 2000; *Khu and Werner*, 2003; *Kourakos and Mantoglou*, 2009; *Liong et al.*, 2001; *Yan and Minsker*, 2006; *Zhang et al.*, 2009; *Zou et al.*, 2007; 2009] and are mostly used in conjunction with GAs [as in *Broad et al.*, 2005; *Khu and Werner*, 2003; *Liong et al.*, 2001; *Yan and Minsker*, 2006; *Zou et al.*, 2007; 2009]. Therefore, we adopt such a combination (ANN-GA) in this study. Similar to the DACE-GA discussion in Section 4.3.2, we could not exactly replicate previously published ANN-GA frameworks due to ANN-GA software unavailability and a lack of framework implementation details in various ANN-GA publications. Figure 4-3 shows the flowchart of ANN-GA when ANN is used as the metamodel. For the ANN implementation, the MATLAB Neural Network Toolbox [*Beale et al.*, 2010] was used.

ANN Structure and Training

The ANN design and fitting/refitting procedures along with the difficulties associated with the application of ANNs in metamodelling context (and how we tried to address them) are presented in this section. Determination of optimal or proper structure of ANNs is a main step in ANN-based metamodel design. In a multilayer perceptron neural network (MLP), number of hidden layers, number of neurons in each hidden layer, and transfer functions are the subjective decisions (structure parameters) the user must make. ANNs with one sigmoidal hidden layer and linear output layer have been proven capable of approximating any function with any desired accuracy provided that associated conditions are satisfied [*Hornik et al.*, 1989; *Leshno et al.*, 1993]. Almost all metamodel-enabled optimization frameworks using ANNs have utilized single hidden layer neural networks. For example, we are only aware of one ANN-based metamodelling study (*Liong et al.* [2001]) that used an MLP with more than one hidden layer. Accordingly, in our study, a single hidden layer neural network with a tangent sigmoid function in hidden layer and a linear function in output layer was used. The number of parameters (weights and biases) to be adjusted in such an ANN is $m\times(2+D)+1$ where D is the number of inputs (DVs of the optimization problem) and m is the number of hidden neurons. The optimal number of hidden neurons, m, is a function of form and complexity of the underlying function [*Xiang et al.*, 2005] as well as the training data availability. In the optimization context, the form of a function to be optimized is often unclear, therefore, the number of data points available for training, p, is the main factor involved in determining m. It is usually preferred that the number of ANN parameters be less (or much less) than p as discussed in *Maier and Dandy* [2000], although mathematically there is no limitation when the number of parameters is higher than p. A possibly good idea is to enlarge m dynamically as more design sites become available. In this study, we followed the trial-and-error approach used in all similar studies [*Behzadian et al.*, 2009; *Broad et al.*, 2005; *Johnson and Rogers*, 2000; *Khu and Werner*, 2003; *Kourakos and Mantoglou*, 2009; *Liong et al.*, 2001; *Yan and Minsker*, 2006; *Zhang et al.*, 2009; *Zou et al.*, 2007; 2009]. Clearly, this trial-and-error step considerably adds to both analyst time and metamodelling time. Generally, for a specific problem, there are multiple good structures, and for each structure, there are many good/acceptable parameter sets. However, the error surface (ANN error function with respect to network weights and biases) of the more parsimonious networks are more complex and harder to train, while the more flexible ones may become over-parameterized and degrade in generalization ability [*Razavi and Tolson*, 2011]. In this study, significant effort was also devoted to determine a proper number of initial design sites, p (these numbers are presented in Section 4.6).

As there is no guarantee to reach to a good solution when whatever ANN training algorithm applied has converged, at the first ANN training step, we conduct 50 independent training trials starting from different solutions initialized through Nguyen-Widrow method [*Nguyen and Widrow*, 1990] and take the best one (the one with lowest error function value – see Equation 4-2). Like DACE-GA, refitting the ANN in our ANN-GA is also performed in two levels: the first level is to add the new point to the set of design sites and re-train the ANN starting from the current state (current weights and biases), but the second level is to re-train an ANN by exactly the same procedure as the one used in the first ANN training step (50 independent training trials). Like DACE-GA, the first level is performed whenever the algorithm goes back to the refitting step (second box in the flowchart shown in Figure 4-3). But the second level is performed after every 50 new design sites are collected.

The logic behind frequently re-training the ANN from scratch is that it helps ANN-GA escape from false regions of attraction, which sometimes may be captured by an inappropriately formed ANN, and thus explore the search space more effectively and globally. The second-order variations of backpropagation algorithms (i.e., quasi-Newton algorithm) are the most computationally efficient training algorithms [*Hamm et al.*, 2007]. In this study, the highly efficient Levenberg-Marquardt algorithm available in MATLAB neural network toolbox [*Hagan and Menhaj*, 1994] was used. ANN training even through the Levenberg-Marquardt algorithm is computationally demanding. The analyst and metamodelling times spent for designing and (re)training an ANN metamodel (even through the Levenberg-Marquardt algorithm) can be prohibitively long.

Non-interpolation Issue in Emulation

In addition to the ANN challenges associated with its subjective design process and computationally demanding training, there is a shortcoming in applying ANNs to approximate the deterministic response of computer simulation models. Neural networks are non-interpolating approximators, also called inexact emulators, (as opposed to interpolating approximators such as DACE) producing a varying bias (usually unpredictable) at different design sites. ANNs may suffer the most from this varying bias when emulating deterministic response of a system as, according to *Villa-Vialaneix et al.* [2011], other non-interpolating emulation techniques may perform better in such cases. The interpolative/non-interpolative distinction is important since there are two general types of problems involving function approximation: physical experiment and computer experiment. There may exist substantial random errors in physical experiments due to different error sources; whereas, computer simulation models are usually deterministic (noise-free) which means observations generated by a computer model experiment with the same set of inputs are identical. This distinction has also been acknowledged in some publications, e.g., *Sacks et al.* [1989] and *Jones* [2001]. Accordingly, the non-interpolating approximation techniques such as ANNs are more suitable for physical experiments than computer experiments as the usual objective is to have an approximation that is insensitive (or less sensitive) to noise. As a result, ANNs have been widely and successfully applied in relating or approximating different error-prone variables existing in the water resources problems including: meteorological variable forecasting [e.g. *Karamouz et al.*, 2008; *Luk et al.*, 2000], rainfall-runoff modelling [e.g. *Hsu et al.*, 1995; *Khan and Coulibaly*, 2006; *Shamseldin*, 1997], and streamflow modelling and prediction [e.g. *Razavi and Karamouz*, 2007; *Razavi and Araghinejad*, 2009].

Jones [2001] pointed out that non-interpolating surfaces are unreliable because they might not sufficiently capture the shape of the deterministic underlying function. However, Ratto and Pagano [2010] showed that the most advanced smoothing (non-interpolating) methods compare favourably with kriging which is an interpolant. Overall, in metamodel-enabled optimization, it is sometimes beneficial to have a smooth approximate response surface passing between design sites in regions of the feasible space with inferior quality, and it may lead the search smoothly to the regions of attraction. However, on the other hand, it can be very misleading in regions of attraction where the superiority of candidate solutions over each other is very important and the key to continue the search. To our knowledge, this issue has not been addressed in the metamodelling literature. In this study, to diminish this inherent negative effect, we devised a weighted error function to be used in ANN training instead of usual error functions which put equal emphasis on all design sites (and in our preliminary ANN-GA experiments yielded inferior results compared to when our weighted version was used). An ANN error function quantifies the discrepancies between the ANN outputs and design sites which are to be minimized in the training process. The new error function is a weighted sum of squared errors as follows:

$$wsse = \sum_{j=1}^{i} (w_j e_j)^2 \tag{4-2}$$

where i is the number of design sites, e_j is the approximation error for the jth design site, and w_j is its corresponding weight. The w_j values are linearly proportional to the quality (objective function value) of a design site in a way that the best design site gets a weight value of 1 and the worst one gets 0.1. This strategy leads the neural network to learn the better design sites (which are more likely in the main region of attraction) more accurately and only capture the general trend (less accuracy) of the underlying response surface in the poorer quality regions. We implemented our *wsse* through customization capabilities of the MATLAB neural network toolbox.

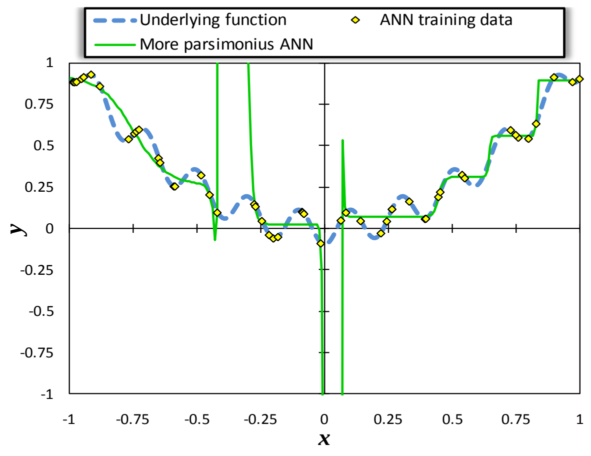
Over-fitting Issue

The other issue in ANN-based metamodels is over-training. Although the likely occurrence of this phenomenon is globally accepted when ANNs are applied to approximate physical processes (fitting on physical experiments), surprisingly, some researchers believe that over-training never occurs when ANNs are fitted over noise-free data (i.e., data obtained from deterministic computer experiments). For example, in water resources independent publications, this belief has been explicitly stated in

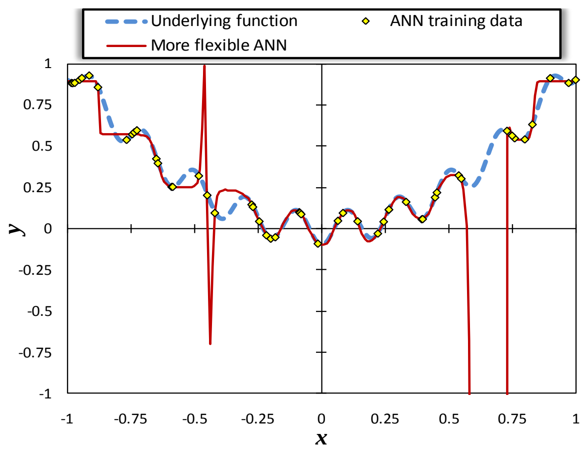
Sexton et al. [1998] expressing “when there is no error in the data, as with the examples so far, a NN cannot be over-trained”. Similarly, in the paper by *Jin et al*. [2002] which proposes the concept of evolution control in metamodel-enabled optimizers based on the metamodel embedded evolution approach, no attention/effort was devoted to avoid ANN over-training. Similarly, in ANN enabled optimization papers in water resources literature, there are papers involving ANNs as metamodels which were not concerned about over-training at all.

In statistics, *over-fitting* is usually described as when the model fits the noise existing in the data dominantly rather than the underlying function. This phenomenon is more likely when the model has a large degree of freedom (is over-parameterized) compared to the amount of available data. However, there is another factor affecting the potential for overfitting which is the *conformability* of the model structure with the shape of the available data. Curve-fitting (regression analysis) practices are less prone to the negative effects of this factor because they have a global pre-specified model structure (form) covering the input variable space. While in the ANN context, as the response of neural networks is formed by a union of a number of local flexible nonlinear units, the problem associated with the conformability factor can be substantial (especially when the highly efficient quasi-newton training algorithms are used). This is not only an issue with noisy data, and it may also occur when the data are noise-free, especially when the underlying function in the noise-free data is non-smooth or complex (i.e., highly multi-modal). Therefore, the so-called over-training issue in ANN context can be caused by both aforementioned factors.

As in the metamodelling context (noise-free data) the second factor may cause over-training (and it is sometimes ignored), Figure 4-4 demonstrates how it may happen and how significant its effect can be through a simple illustrative example. In this figure, the underlying function is $y = x^2$-0.1 cos (10 πx), and the objective is to approximate it with 50 random design sites. Two single-hidden-layer ANNs differing in the number of hidden neurons, 15 and 8, are applied. The response of both ANNs after 1000 epochs with the Levenberg-Marquardt training algorithm is shown in Figure 4-4. The first ANN which has 46 parameters (weights and biases) is a relatively flexible ANN when compared to the number of design sites (Figure 4-4a); while the second one with 25 parameters is relatively parsimonious (Figure 4-4b). As demonstrated, although in a majority of the problem domain ANNs have properly captured the underlying function, both ANNs have presented wild and unpredictable fluctuations in some other parts of the domain. The occurrence probability of such behaviour is higher in the areas where fewer design sites are found.



(a)



(b)

Figure 4-4. Examples of over-trained ANNs responses over noise-free data: (a) a relatively flexible ANN, and (b) a relatively parsimonious ANN

In environmental and water resources modelling literature, there are two well-known approaches to avoid over-training: early stopping and Bayesian regularization, and both have been efficiently implemented in MATLAB neural network toolbox [*Beale et al.*, 2010]. Most ANN-based metamodelling studies considering over-training have applied the early stopping approach [*Broad et al.*, 2005; *Johnson and Rogers*, 2000; *Khu and Werner*, 2003; *Zou et al.*, 2007; 2009]. The main problem with early stopping is that the available design sites have to be split into training and testing sets (and also sometimes validation set) resulting in fewer data available to train ANNs which decreases the approximation accuracy, especially in cases where the original model is computationally expensive and, therefore, the available data are scarce.

The Bayesian regularization procedure is less common in ANN metamodelling literature. It has been used in *Kourakos and Mantoglou* [2009] to avoid over-training of an ANN metamodel for coastal aquifer management. Bayesian regularization has been proposed by *MacKay* [1992] and generalized to neural network applications by *Foresee and Hagan* [1997]. Although the logic behind this technique, which tries to keep a balance between accuracy and smoothness, and the steps involved are very sophisticated, it can be easily used through its efficient MATLAB implementation. Our understanding is that Bayesian regularization may substantially sacrifice neural network accuracy for smoothness, particularly for noise-free data. We have verified this statement through some experiments which are not presented in this paper. In this study, the early stopping approach was used to avoid over-training. As higher quality design sites are of more importance in metamodel fitting (higher accuracy in more promising regions is desired), the testing sites were not selected from the first one-third best sites.

Over-fitting may also occur in other function approximation techniques including DACE, even when being fitted over noise-free data due to the aforementioned *conformability* issue. However, the risk and extent of over-fitting in DACE is typically less compared to ANNs. The risk of over-fitting in DACE is higher when there are very few design sites relative to the number of DACE hyper-parameters (i.e., correlation function parameters) to be tuned [*Welch et al.*, 1992]. Note that, for example, the DACE model we used in this study with Gaussian correlation function has *D* correlation function parameters (hyper-parameters) each of which associated with each dimension in the design sites space (*D* equals the number of decision variables in the original optimization problem) and *i* parameters (*i* is the number of design sites) determined through BLUP (best linear unbiased predictor) [*Sacks et al.*, 1989]. As such, typically, the number of hyper-parameters in DACE is considerably less than the number of parameters in ANNs. As an example for the first test problem in this study, as demonstrated in Section 4.6.1, the number of ANN parameters was determined as 85, 121, and 181 for different computational budgets, while the number of DACE hyper-parameters was 10 for the same test problem (for all computational budget scenarios). As over-fitting in DACE is not a major challenge, it has not been directly addressed in some DACE studies including this work. Over-fitting has been also addressed in other function approximation techniques including smoothing spline ANOVA models [*Gu*, 2002; *Curtis B. Storlie et al.*, 2011].

4.4 Benchmark Optimization Algorithms

Two benchmark optimization algorithms (without metamodelling), GA and dynamically dimensioned search (DDS), were used in this study to develop a baseline for assessing the applied metamodel-enabled optimizers. Whenever applicable, DDS was also enabled with the so-called "model preemption" strategy to enhance the efficiency of the optimization process.

4.4.1 Genetic Algorithm (GA)

The GA was used as a benchmark optimization algorithm despite its potential ineffectiveness for limited computational budgets, because GAs are one of the most commonly used family of optimization algorithms in environmental and water resources [*Nicklow et al.*, 2010] and are one of the most common benchmark algorithms applied in metamodelling studies [*Broad et al.*, 2005; *Fen et* *al.*, 2009; *Khu et al.*, 2004; *Ostfeld and Salomons*, 2005; *Zou et al.*, 2007]. In this study, the GA in the MATLAB global optimization toolbox [*MathWorks*, 2010] was used. The specific GA reproduction steps utilized here were tournament selection, scattered crossover, and adaptive feasible mutation, chosen based on our previous experience with this GA. Note that scattered crossover and adaptive feasible mutation are default operators for bounded optimization problems. Details of these operators have been well-documented in *MathWorks* [2010]. The GA parameters that should be tuned for any specific problem are presented in Section 4.5.3.

4.4.2 Dynamically Dimensioned Search (DDS)

DDS [*Tolson and Shoemaker*, 2007b] was selected here as the second benchmark optimization algorithm because it was designed and has been demonstrated to work very well when the computational budget is very limited. DDS, which is a single-solution based algorithm, is unique compared to other optimization algorithms with respect to the way that the neighbourhood is dynamically defined by changing the dimension of the search as a function of current iteration number and the user-specified maximum number of function evaluations. One valuable feature of DDS is that it requires no algorithm parameter adjustment, unlike other commonly used stochastic global search algorithms such as GAs. This value becomes more important in computationally intensive optimization problems, as tuning algorithm parameters in such problems can be prohibitively long, forcing practitioners to use default algorithm parameter settings which may be far from optimal.

4.4.3 DDS with Preemption

This study also uses a more efficient extension of DDS called “DDS with preemption” hereafter. The *deterministic* model preemption concept developed and formalized in Chapter 2 (also in *Razavi et al*. [2010]) can be used in conjunction with a variety of optimization algorithms to enhance their computational efficiency. Model preemption is an approach to opportunistically evade unnecessary evaluations of computationally expensive simulation models. Deterministic preemption is applicable when the objective function value monotonically increases (decreases) in minimization (maximization) problems as simulation proceeds. For example, in hydrologic model automatic calibration, preemption is applicable when the objective function is a summation of model prediction error terms accumulating throughout the simulation time period. As such, model preemption monitors the intermediate results of a model simulation and terminates the simulation early (i.e., prior to simulating the entire time period) once it recognizes that this solution is so poor that it will not contribute to guiding the calibration algorithm. The attractive feature of the deterministic preemption strategy is that its application leads to exactly the same result as when it is not applied. As reported in Chapter 2 (also in *Razavi et al.* [2010]), preemption may lead up to 60% computational saving in an optimization problem. In this study, DDS was used for test functions (see Section 4.5.1) and DDS with preemption was used for real-world computationally intensive automatic calibration case studies (see Section 4.5.2).

4.5 Design of Numerical Assessments

This section aims to design a fair and comprehensive numerical assessment of metamodel-enabled optimizers versus the optimizers without metamodelling. As outlined in Section 4.1, we believe that the shape and the complexity of the original function and the computational budget availability are the most important factors affecting the performance of metamodelling. Therefore, as presented in Section 4.5.1, four test functions with different characteristics and different levels of complexity were used for the comparative assessment. Two computationally intensive water resources optimization problems, as presented in Section 4.5.2, were also used as representatives of real-world problems. The experiments were conducted within different computational budget availability scenarios. Details of experimental settings are presented in Section 4.5.3.

4.5.1 Test Functions

Four mathematical functions commonly used as performance test problems for optimization algorithms, namely the Griewank, Ackley, Rastrigin, and Schwefel functions were used in this study. Figure 4-5 shows the perspectives of the 2-dimensional versions of these test functions which have different characteristics. As can be seen in Figure 4-5, the general form of Griewank's function is well-behaved and convex with numerous local minima attached to the global form. Ackley's function is a more difficult function which has a large non-informative area (due to the exponential form of the function) containing numerous local minima where there is not a detectable trend towards the global region of attraction. Rastrigin's function is a fairly difficult, highly multi-modal function with regularly distributed local minima and a large search space. Schwefel's function consisting of a large number of peaks and valleys is a difficult and deceptive function in a way that the global minimum, which is located near the bound of feasible space, is geometrically distant from the next best local minima; as such, optimization algorithms are potentially prone to converging far from the true optimum.

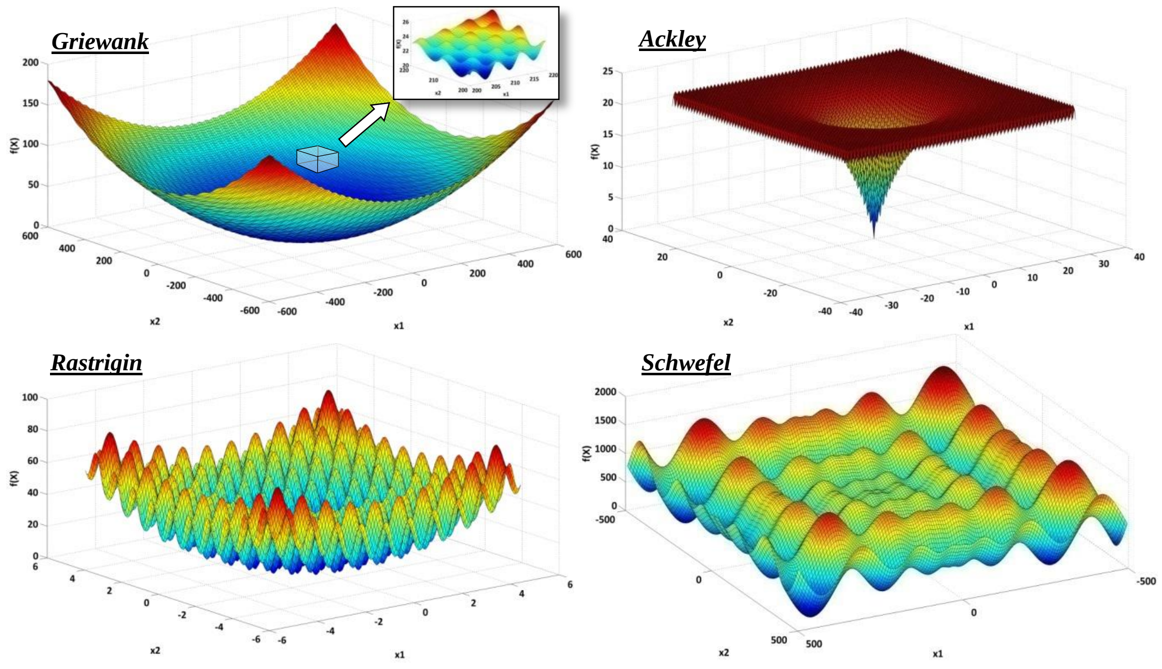


Figure 4-5. Perspectives of the 2-D versions of the applied test functions

Table 4-1 presents the formulations, bound constraints, and arbitrarily selected numbers of dimensions used in this study of these test functions. As stated in Section 4.2.3, the number of dimensions (i.e., number of decision variables of the optimization problem) in metamodel-enabled optimizers cannot typically be large (typically less than 15-20), and the performance of metamodel-enabled optimizers is expected to degrade when applied to higher dimensional problems. The global optima of the Griewank, Ackley, and Rastrigin functions are located at the center of feasible space ($x_i = 0$, i=1,…,D), while the global minimum of the Schwefel function is at $x_i = 420.9687$, i=1,…,D. Note that increasing the number of dimensions does not necessarily correspond to increasing complexity. For example in the Griewank function, although the number of local minima increases exponentially with the number of dimensions, *Locatelli* [2003] has shown that the function becomes very easy to optimize for large numbers of dimensions (because local minima become extremely small, resulting in an almost uni-modal form of the function), by any optimization algorithms, especially derivative-based ones. On the other hand, in the Ackley function, the ratio of the size of the main region of attraction to the size of the entire feasible space becomes smaller as the number of dimensions increases – for example from roughly 25% for the 2-D Ackley function to less than 0.5% for the 15-D Ackley function (obtained through Monte-Carlo analysis). It is worth noting that there are metamodelling studies optimizing the Ackley function in a reduced DV range, while the reduced range only covers the informative part around the global minimum having a clear almost linear general trend (with smaller scale sinusoidal fluctuations) towards the optimum. As a result, metamodel-enabled optimizers have substantially less difficulty optimizing the Ackley function in the reduced range.

Table 4-1. Summary of Optimization Test Functions

Name	Equation	Number of dimensions (*D*)	Bound Constraints	Min
Griewank	$f(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}}) + 1$	10	[-600, 600]^D	0
Ackley	$f(x) = -20 \exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - exp\left(\frac{1}{D}\sum_{i=1}^{D} \cos(2\pi x_i)\right) + 20 + \exp(1)$	15	[-32.768, 32.768]^D	0
Rastrigin	$f(x) = 10\,D + \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i)]$	10	[-5.12, 5.12]^D	0
*Schwefel**	$f(x) = 418.9829\,D + \sum_{i=1}^{D} \left[-x_i \sin(\sqrt{\lvert x_i \rvert}\right]$	15	[-500, 500]^D	0

* the first term in the equation does not exist in the original form of the function. Here, it is added to set zero as the minimum of the function.

4.5.2 Computationally Intensive Calibration Problems

In addition to test functions, three benchmark real-world computationally expensive optimization problems were utilized in this study. The first one is an automatic calibration problem of SWAT2000 streamflow model being calibrated over the Cannonsville Reservoir watershed in Upstate, New York. This case study, which has been originally developed by *Tolson and Shoemaker* [2007b], seeks to calibrate 14 parameters by maximizing the Nash-Sutcliffe coefficient for daily flow at the Walton gauging station. Details of this case study are available in *Tolson and Shoemaker* [2007b]; for range constraints see Table 2 therein. This case study, which is called SWAT hereafter, is exactly the same as the so-called SWAT-1 case study in Chapter 2 and *Razavi et al.* [2010] used to demonstrate the performance of DDS with preemption. A single evaluation of this watershed model requires about 1.8

minutes on average to execute on a 2.8GHz Intel Pentium processor with 2 GB of RAM and running the Windows XP operating system.

The second benchmark problem is based on a recently introduced groundwater flow and reactive transport model that is designed to aid in the interpretation of aquifer tests. Dipole flow and reactive tracer test (DFRTT) is a single-well test proposed for in situ aquifer parameter estimation to aid in the design of remedial systems for contaminated sites. The observed breakthrough curve (BTC) obtained through this test is analyzed by a DFRTT interpretation model (DFRTT-IM) to estimate aquifer parameters. DFRTT-IM is a high-resolution two-dimensional radially symmetric finite volume model. For details, interested readers are referred to Thomson et al. [2010]. This case study seeks 7 aquifer parameter values of an unconfined sand aquifer at the Canadian Forces Base (CFB) Borden near Alliston, ON, Canada. The objective is to minimize a weighted sum of squared deviations of DFRTT-IM outputs from an observed BTC. Detailed description as well as parameter range constraints are available in Chapter 2 and *Razavi et al*. [2010]. A single evaluation of this model takes about 37 minutes on average to execute on a 2.8 GHz Intel Pentium processor with 2 GB of RAM and running the Windows XP operating system.

The third case study is the automatic calibration problem of a three-layer confined groundwater model with 122*km* x 167*km* area located in northeast Alberta, Canada (see Figure 4-6). The first and third layers are aquifers characterized by the same parameters and the second layer is a shale unit with incised channels that threaten the integrity of this unit as an aquitard. This case study seeks to calibrate 56 parameters to transient data of water withdrawal and associated pressure response by minimizing weighted sum of squared errors (WSSE) in drawdowns. The parameters include horizontal hydraulic conductivities in the aquifer at 52 pilot points (located in local study area), horizontal hydraulic conductivity in the shale layer, vertical hydraulic conductivities in the aquifer and shale layers, and specific storage. Horizontal hydraulic conductivity of the aquifer in the regional study area is the average of 52 pilot point values. This case study is set up with FEFLOW groundwater modelling software package and is called the “FEFLOW” case study hereafter. A single evaluation of this model takes about 30 seconds on average to execute on a 2.8 GHz Intel Pentium processor with 2 GB of RAM and running the Windows XP operating system.

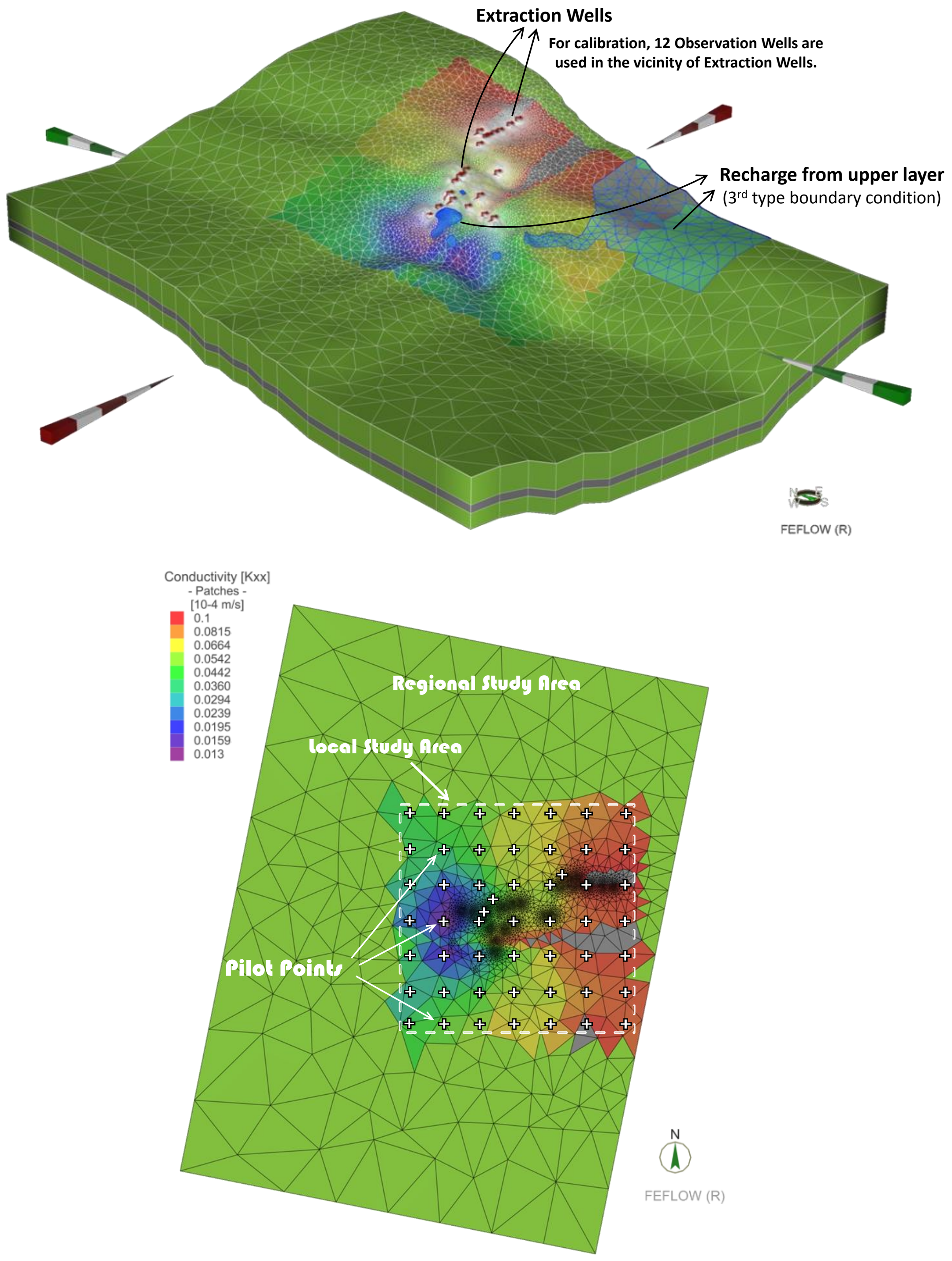


Figure 4-6. The 3D view and the map of the groundwater model implemented with FEFLOW – this case study seeks to calibrate 56 parameters (including 52 pilot points for horizontal hydraulic conductivities) to transient drawdown data

4.5.3 Experimental Setting

Computational budget availability is a limiting factor in solving computationally intensive optimization problems. In such problems, practitioners sometimes are restricted to find a reasonably good solution within a very small number of function evaluations (for instance, as small as 100). To replicate possible common practices, the following four computational budget availability scenarios based on the total number of original function evaluations were assumed to be available: 100, 200, 500, and 1000. Optimization within only 100 or 200 original function evaluations are consistent with the benchmark metamodelling study by *Regis and Shoemaker* [2007b] in which MLMSRBF (see Section 4.3.1) was proposed. One thousand function evaluations was deemed as the maximum practical budget available as it might require a very long computational time; for example, about 26 days of serial processing are needed to run the DFRTT model 1000 times which might be practically infeasible. The performance of the optimization algorithms (with and without metamodelling) within each computational budget scenario can be accurately compared in terms of algorithm effectiveness – the final solution quality attained by different algorithms at each computational budget scenario can be directly compared.

Due to the excessively high computational demand of the experiment with DFRTT, only the two very limited computational budget scenarios (i.e., the scenarios with 100 and 200 function evaluations) were evaluated for this case study. Note that in all cases, experiments in each computational budget scenario were independent from the experiments for other scenarios. For example, for the budget of 200 function evaluations, we ran the experiments from scratch, without getting any feedback from the experiments with the budget of 100 function evaluations.

As stated in Section 4.4.3, DDS was used in conjunction with the model preemption strategy (i.e., DDS with preemption) for the SWAT and DFRTT case studies. Chapter 2 and *Razavi et al*. [2010] show that deterministic model preemption can save approximately 15% and 50% of the computational budgets required by DDS to run on the SWAT and DFRTT case studies, respectively, while it can find exactly the same near-optimal solutions as when preemption was not applied. Therefore, for DDS with preemption, we add 15% and 50% to the total number of function evaluations allocated in each scenario for the SWAT and DFRTT case studies. For example, we consider 230 ($200+0.15\times200$) function evaluations for DDS with preemption on the SWAT case study when the computational budget is equivalent to 200 full function evaluations scenario.

To ensure a fair comparison and make statistically valid conclusions, multiple replicates with different initial conditions (i.e., random seeds) should be conducted. On the test functions, all optimization algorithms were conducted for 30 replicates; however, only five optimization replicates were conducted for each algorithm on the real-world computationally intensive case studies, SWAT and DFRTT, due to their high computational demand. For the FEFLOW case study, only MLMSRBF and DDS were run and only with the computational budgets of 200 and 500 function evaluations. This case study is not included for the full analysis with all the algorithms and computational budget scenarios, as the number of DVs in this case study is considerably larger than practical numbers of DVs in the context of metamodelling.

For DDS, DDS with preemption, and MLMSRBF, there is no need to tune algorithm parameters; while GA (in GA without metamodelling and in DACE-GA and ANN-GA) has multiple algorithm parameters that require tuning. In this study, tournament size and crossover ratio were fixed equal to 4 and 0.8, respectively [the MATLAB default values, *MathWorks*, 2010], and population size, number of generations, and elite count were considered as the GA algorithm parameters that need to be tuned for any specific problem. In DACE-GA and ANN-GA, where the GA runs over a fast-to-run metamodel (no strict limit on the total number of metamodel evaluations), population size, number of generations, and elite count were selected to be large and equal to 100, 100, and 4, respectively. However, as GA without metamodelling was to run on an original function (which is assumed computationally expensive), selection of a proper GA algorithm parameter set needed more careful attention. Obviously, when computational budget is limited and fixed (i.e., total number of function evaluations is fixed in advance), the number of generation is a known function of population size, elite count and total number of function evaluations. For the test functions, different configurations of these GA parameters were tested and only the best GA results were presented in the Results Section (Section 4.6). Importantly, the algorithm parameter tuning process substantially adds to the computational burden of an experiment, and it is not feasible when the original simulation model is computationally expensive. However, as the GA parameters highly affect the algorithm performance, we wanted to solicit (almost) the best performance of the GA on the test functions which was to be compared with the performance of other algorithms. For the real-world computationally intensive case studies, the GA parameters were selected based on our evaluation of the GA results over the test functions.

In the metamodelling part of DACE-GA, we used the metamodelling parameters presented in Section 4.3.2, while for ANN-GA we still had to determine a proper number of hidden neurons for any given case study by conducting multiple trials with various numbers of hidden neurons. We also had to determine a proper number of initial design sites for ANN-GA in a given computational budget, and this was also selected by trial-and-error experiments. The selected numbers of hidden neurons and initial design sites for ANN-GA is presented in Section 4.6.

To quantify the computational budget required for a metamodelling experiment, especially for comparison purposes, it is common and convenient to only consider and compare the total number of original model evaluations in a given optimization trial [*Mugunthan and Shoemaker*, 2006; *Regis and Shoemaker*, 2007b; *Zou et al.*, 2007; 2009]. Similarly, in the comparative assessments performed in this study, we also ignored the analyst time and the metamodelling time in our experiments and comparisons were based on number of original model evaluations.

4.6 Results

The average performance of the metamodel-enabled optimizers (DACE-GA, ANN-GA, and MLMSRBF) as well as the optimizers without metamodelling (DDS and GA) over the test functions and the SWAT and DFRTT real-world case studies are shown in Figure 4-7. Average performance for each computational budget scenario is the average of the best original objective function values found in all replicates. The empirical cumulative distribution functions (CDFs) of the final best function values found within various computational budget scenarios from all 30 optimization replicates are shown in Figure 4-8, Figure 4-9, Figure 4-10, and Figure 4-11 for the Griewank, Ackley, Rastrigin, and Schwefel functions, respectively. As an example on how to interpret the CDFs, according to Figure 4-8, the probability that the GA with 100 function evaluations attains an objective function value of at most 30 on the Griewank function is about 0.15. A more vertical CDF indicates less variable algorithm performance (more robust), and as such, CDFs that are vertical and as far to the left as possible are ideal. When comparing algorithms A and B in terms of their respective empirical CDFs of best (minimum) objective function values attained, F_A and F_B, algorithm A dominates algorithm B stochastically at first order if, for any desired objective function value *f*, $F_A(f)$ ≥¸ $F_B(f)$. As for the real-world case studies the number of replicates is small (i.e., 5), instead of CDFs, the dispersion of the final solutions found through each algorithm are compared using the simple plots shown on Figure 4-12 and Figure 4-13, for the SWAT and DFRTT case studies, respectively. When the final objective function values found in all replicates through an algorithm are closer to each other, the algorithm is less variable. In the following, we have allocated a sub-section to each test function and each real-world case study to elaborate the different and function-dependent performance of the metamodel-enabled optimizers versus the optimizers without metamodelling. Due to the reasons outlined in Section 4.6.1, ANN-GA was only assessed on the Griewank function.

4.6.1 Griewank Function

In the Griewank function, DACE-GA and MLMSRBF drastically outperformed DDS and GA and both were comparable in terms of approximating the global minimum very fast and efficiently. Within only 100 original function evaluations, the average function values found through both these metamodel-enabled algorithms are approximately 1 and the standard deviations are less than 0.2. However, the ANN-GA was completely unable to find a trajectory toward optimality within 100 function evaluations. In other words, the ANN metamodel was misleading in this very limited budget and the ANN-GA search was hardly ever able to find a better solution based on the metamodelling guidance than the solutions initially evaluated though original function for initial DoE. As a result, the final best function values reported for ANN-GA with 100 function evaluations are almost always the best point in the initial design sites. In 200 function evaluations scenario, the ANN metamodel was not as misleading and ANN-GA performance was almost the same as GA in terms of mean function value but inferior in terms of robustness (see associated CDFs in Figure 4-8). In the 500 and 1000 function evaluations scenarios, ANN became able to play the metamodelling role properly although it was still outperformed by DACE-GA and MLMSRBF.

The relatively poor performance of ANN-GA was observed despite the fact that at least four extra parameters/decisions existing in ANN-GA, which are not in DACE-GA and MLMSRBF, (i.e., how to handle over-training prevention and inexact emulation behaviour as well as determination of the number of hidden neurons and the initial DOE size) were manipulated and fine-tuned to optimize its performance. As such, substantially more analyst time and metamodelling time were spent for ANN-GA fine-tuning compared to the other metamodel-enabled optimizers used herein. As a result, and considering its other shortcomings pointed out in Section 4-3-3, ANN-GA was deemed unsuitable for such computationally intensive problems and was not assessed on the other test functions and real-world case studies.

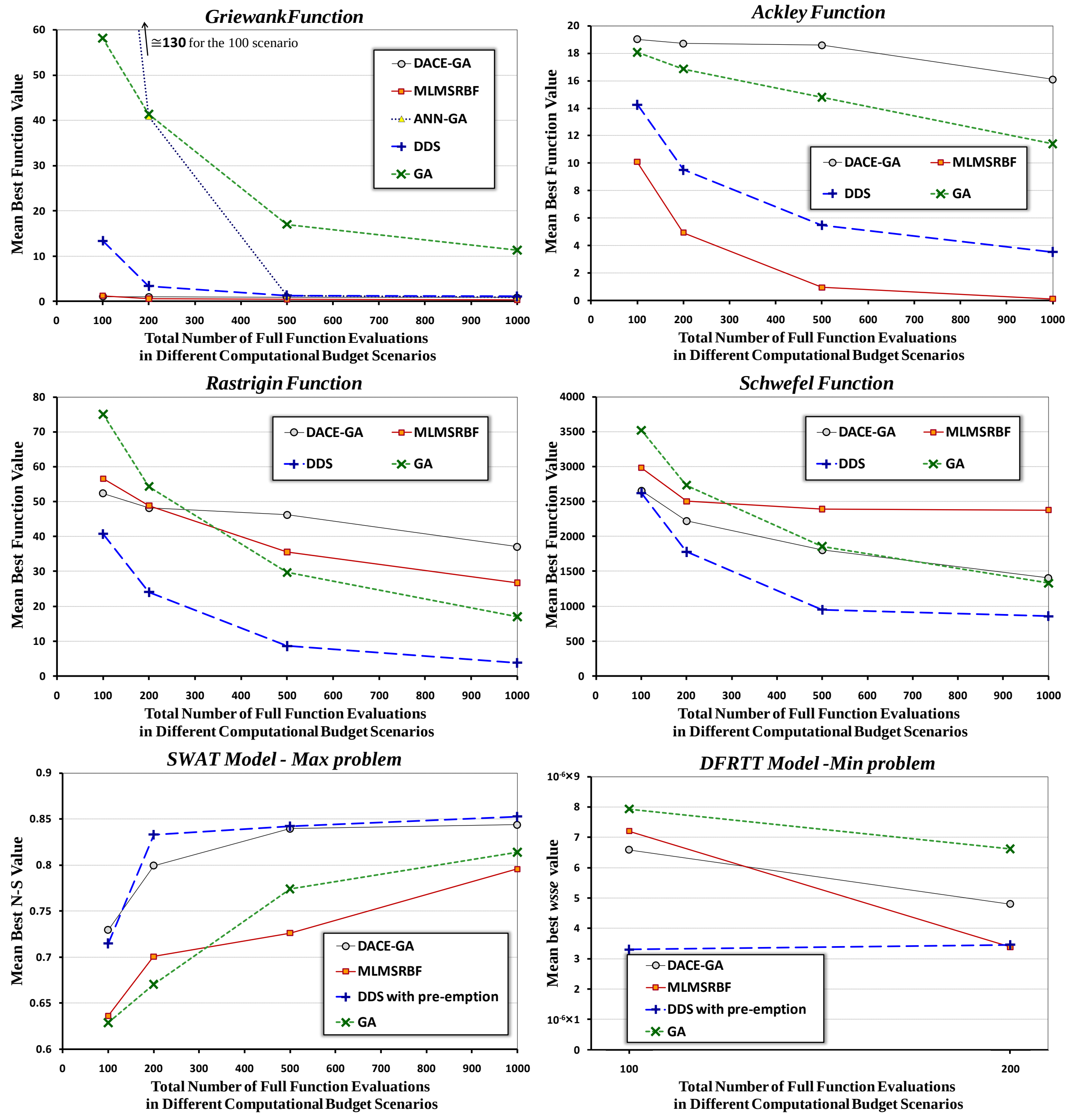


Figure 4-7. Algorithms average performance comparisons on the 10-D Griewank, 15-D Ackley, 10-D Rastrigin, and 15-D Schwefel functions (over 30 replicates) as well as SWAT and DFRTT case studies (over 5 replicates) at different computational budget scenarios

According to Figure 4-8, in all computational budget scenarios, both DACE-GA and MLMSRBF are stochastically dominant over the optimizers without metamodelling, and their relative performance lies under Case A introduced in Section 4.2.1 (idealized relative metamodel performance). Regarding the optimizers without metamodelling, DDS substantially outperformed GA and approached the performance of the metamodel-enabled optimizers in the 500 and 1000 function evaluations scenarios.

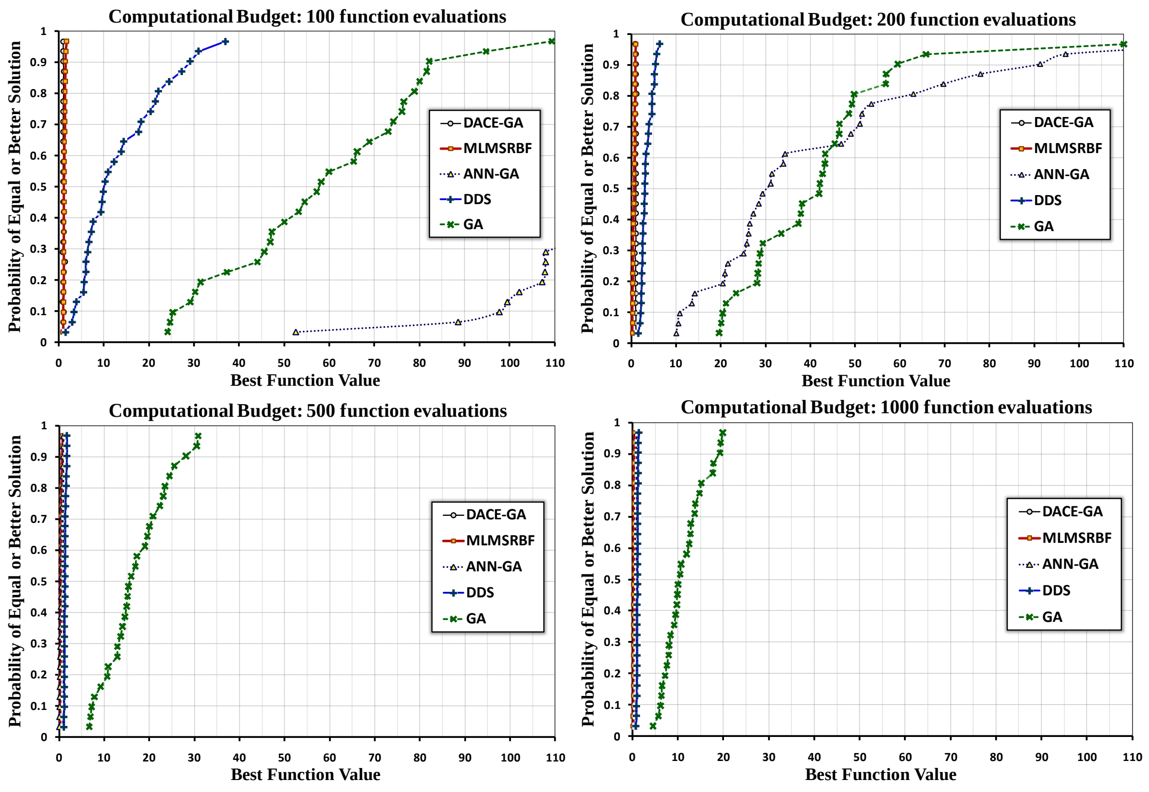


Figure 4-8. Empirical cumulative distribution function of final best function values on the 10-D Griewank function

4.6.2 Ackley Function

On the Ackley function, the metamodel-enabled optimizers, DACE-GA and MLMSRBF, demonstrated completely different behaviours. MLMSRBF outperformed DDS and GA, while DACE-GA performance was the worst of all - DDS and GA stochastically dominate DACE-GA in the 100, 200, and 500 scenarios (see Figure 4-9). In other words, the relative performance of MLMSRBF compared to both GA and DDS is in Case A (idealized relative metamodel performance), whereas the relative performance of DACE-GA compared to GA and DDS is in Case C (failure of metamodelling). As noted in Section 4.5.1, a very small portion of the feasible space of the 15-D

Ackley function is informative, and there is a very large region having an almost flat (plateau) general form with numerous regularly distributed local valleys. Fitting a metamodel on the details of the non-informative region can be quite misleading and it does not give useful information to locate the main region of attraction. In other words, as there is not an effective clue in the general form for locating the main region of attraction, a metamodel-based search may wander around wasting the entire computational budget unless a point on the main region of attraction is evaluated through the original function either in the initial DoEs or while searching. As MLMSRBF restarts once it converges to a local minimum, the number of points on the feasible space being evaluated through DoE in a single optimization is relatively high in comparison to DACE-GA. In addition, MLMSRBF searches locally around the best solution found so far in a trial, therefore, since the main region of attraction is located at the center of the feasible space, the chance of finding it is higher (compared to the case when it is located near bounds or at the corner – this was verified through an experiment not reported here). Once MLMSRBF locates the main region of attraction, it can reach a good solution efficiently due to its local search capability. Accordingly, as can be seen in Figure 4-9, the probability of returning a relatively poor solution for MLMSRBF from the non-informative region diminishes as the computational budget increases – in the 100 and 200 function evaluations scenarios 3-4 replicates (out of 30), with 500 function evaluations only one replicate, and with 1000 none of the 30 replicates are from the non-informative region.

Unlike MLMSRBF, DACE-GA starts with an initial DoE and then searches globally around the feasible space until it finishes the computational budget. As such, DACE-GA performance is drastically inferior compared to MLMSRBF because firstly, the probability of finding a point on the main region of attraction through the initial DoE is lower. Secondly, DACE-GA involves a global search method as opposed to MLMSRBF, and it is not as efficient in improving a solution found in the main region of attraction. As a result in DACE-GA, the metamodel is mostly focused on emulating the deceptive local valleys distributed over the non-informative region. According to Figure 4-9, DACE-GA in the 100 and 200 function evaluations scenarios became mired in the non-informative region in all 30 replicates, while in larger computational budgets, particularly in 1000 function evaluations, some replicates can find the main region of attraction but are unable to efficiently approach the global minimum.

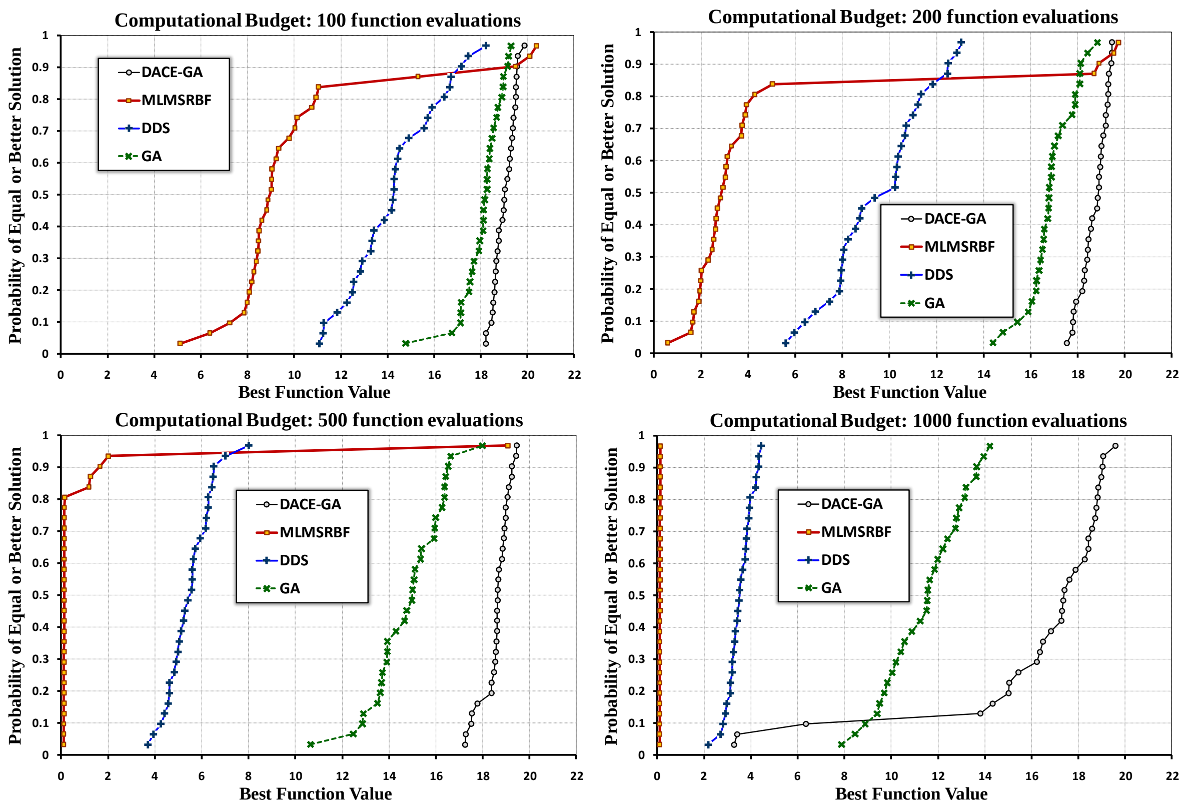


Figure 4-9. Empirical cumulative distribution function of final best function values on the 15-D Ackley function

4.6.3 Rastrigin Function

On the Rastrigin function, DDS stochastically dominates the metamodel-enabled optimizers, DACE-GA and MLMSRBF, and GA in all computational budget scenarios (see Figure 4-10 – note that only CDFs for 100 and 1000 function evaluations are depicted). Therefore, compared to DDS, the performance of both metamodel-enabled optimizers are in Case C (failure of metamodelling). In the 100 and 200 function evaluations scenarios, the metamodel-enabled optimizers outperformed GA, whereas interestingly, in the 500 and 1000 function evaluations scenario, GA surpasses DACE-GA and MLMSRBF. This behaviour confirms that the superiority/suitability of a metamodel-enabled optimizer for a specific problem can be a function of computational budget, as in Case B (computational budget dependent relative metamodel performance) discussed in Section 4.2.1. Equivalence time, *t**, in this case is between the 200 and 500 function evaluations scenarios for both MLMSRBF and DACE-GA. Moreover, DDS and GA are more robust compared to metamodel-enabled optimizers especially in larger computational budgets. The performance of DACE-GA is superior to MLMSRBF in the 100 function evaluations scenario, while it is inferior in the 500 and 1000 function evaluations scenarios. DACE-GA and MLMSRBF performed comparably in the 200 function evaluations scenario.

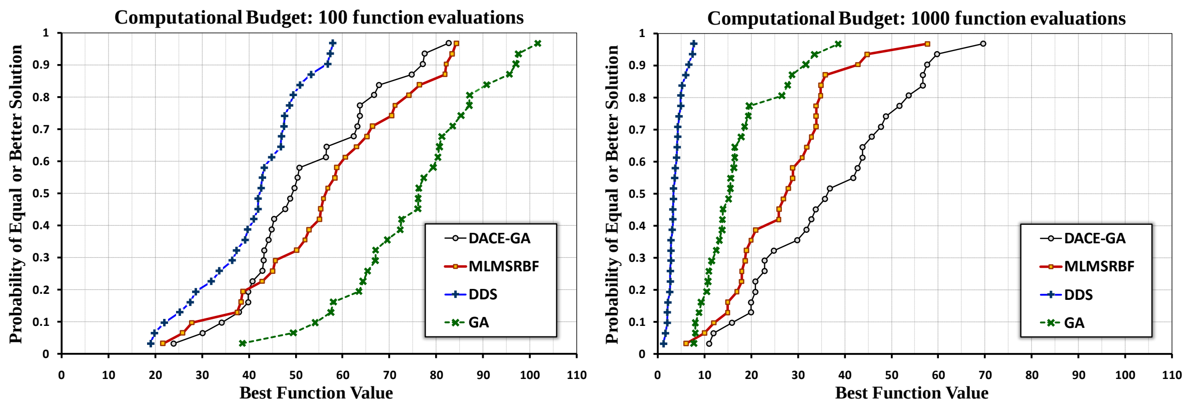


Figure 4-10. Empirical cumulative distribution function of final best function values on the 10-D Rastrigin function

4.6.4 Schwefel Function

For the Schwefel function, the performance of DACE-GA and DDS in the 100 function evaluations scenario were comparable and both superior to MLMSRBF and GA, but in larger computational budgets, DDS outperformed all other algorithms. DACE-GA performance is considerably better (almost stochastically dominant, see Figure 4-11 - note that only CDFs for 100 and 1000 function evaluations are depicted) than MLMSRBF in all computational budget scenarios. GA is the least effective optimizer in the 100 and 200 function evaluations scenarios, while it outperforms MLMSRBF in the 500 and 1000 function evaluations scenarios. Although GA and MLMSRBF performed comparably in terms of mean best function values, GA is more robust as the associated CDFs are more vertical. Note that none of the optimizers (with and without metamodelling) were able to reach close to the global minimum function value of zero in the allocated computational budgets. As can be seen in Figure 4-7, MLMSRBF reached a plateau in performance after the 200 function evaluations scenario. One possible reason for this poor behaviour can be that the main regions of attraction of the Schwefel function are located at the corners of the feasible space geometrically distant and separated from each other, and besides, MLMSRBF searches locally around the best solution found. As a result, MLMSRBF can easily get trapped in the low quality local minima located far from the main regions of attraction. Compared to GA, the relative performance of both metamodel-enabled optimizers is in Case B (equivalence time is between the 200 and 500 function evaluations scenarios for MLMSRBF and between 500 and 1000 for DACE-GA), while compared to DDS, their relative performance is in Case C.

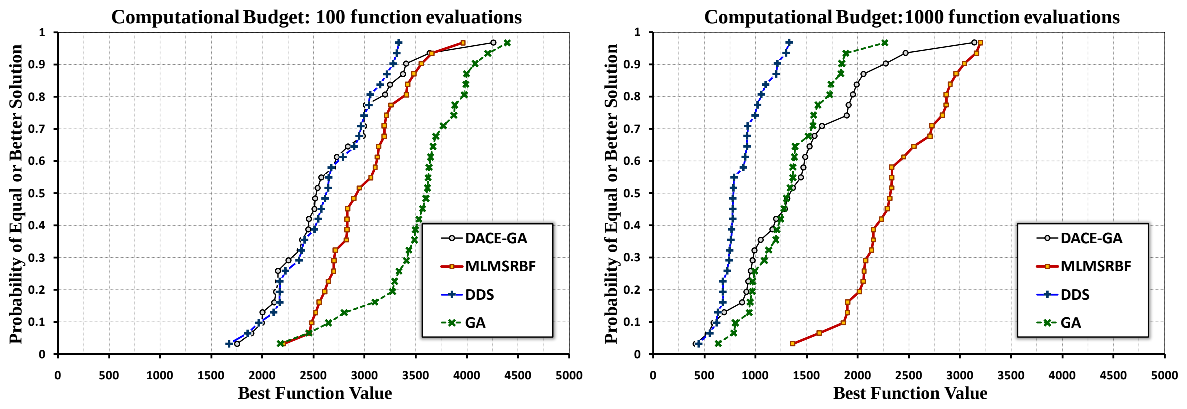


Figure 4-11. Empirical cumulative distribution function of final best function values on the 15-D Schwefel function

4.6.5 SWAT Case Study

On the SWAT case study, DACE-GA outperformed all other algorithms in the 100 function evaluations scenario, but DDS with preemption performed the best in larger budget scenarios in terms of both the mean objective function value and their dispersion over the 5 replicates. GA performance was the worst in the 100 and 200 function evaluations scenarios, but it surpassed MLMSRBF in the 500 and 1000 function evaluations scenarios. In the 100 function evaluations scenario, the dispersion of the final solutions found in the 5 replicates is relatively high for all algorithms (see Figure 4-12). However, for the higher computational budget scenarios, DDS with preemption resulted in final solutions with relatively close quality (close objective function values) indicating that among all, DDS with preemption is the most robust and reliable algorithm on this case study. In addition, DACE-GA, GA, and MLMSRBF are the second-, third- and fourth-ranked algorithms, respectively, in terms of robustness and reliability. Compared to GA, the relative performance of MLMSRBF is in Case B (equivalence time is between the 200 and 500 function evaluations scenarios), and the relative performance of DACE-GA is in Case A. Compared to DDS, the relative performance of MLMSRBF is in Case C and the relative performance of DACE-GA is in Case B (equivalence time is between 100 and 200).

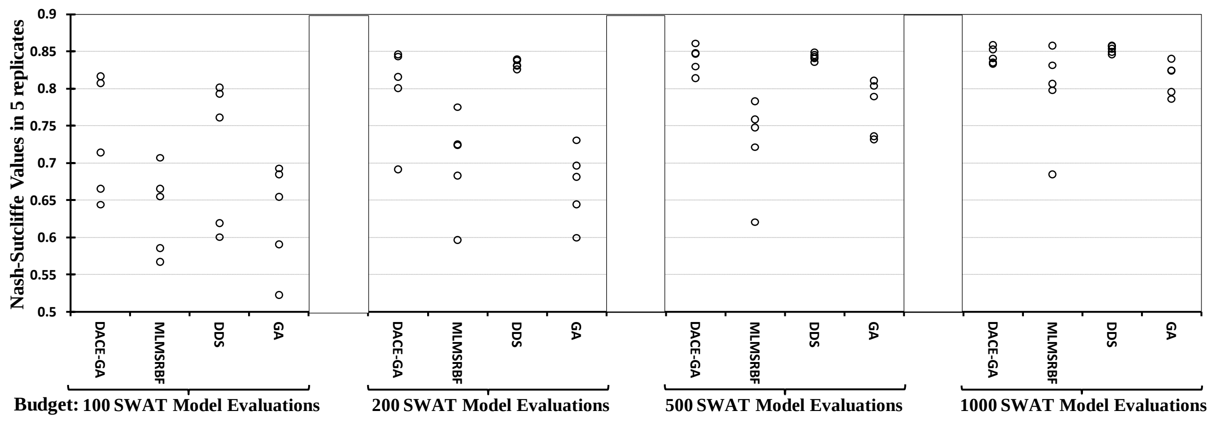


Figure 4-12. Algorithms performance comparisons on the SWAT model – a maximization problem

4.6.6 DFRTT Case Study

The DFRTT case study, as stated in Section 4.5.3, was only used to assess the optimizing algorithms within the two very limited computational budget scenarios with 100 and 200 function evaluations. Overall, DDS with preemption was the most effective and robust optimization algorithm in both 100 and 200 function evaluations scenarios. MLMSRBF was almost as effective; however, as shown on Figure 4-13, in one of the five replicates in the 100 function evaluations scenario, it failed to approach a good quality solution. This one poor result or outlier is why the mean best objective function value obtained through MLMSRBF in the 100 function evaluations scenario is not desirable in comparison with the other algorithms. The MLMSRBF outlier emphasizes the importance of running multiple replicates in an algorithm assessment process. GA performance was the worst in both 100 and 200 function evaluation scenarios. Compared to GA, the relative performance of both metamodel-enabled optimizers lies in Case A, while their relative performance compared to DDS lies in Case C.

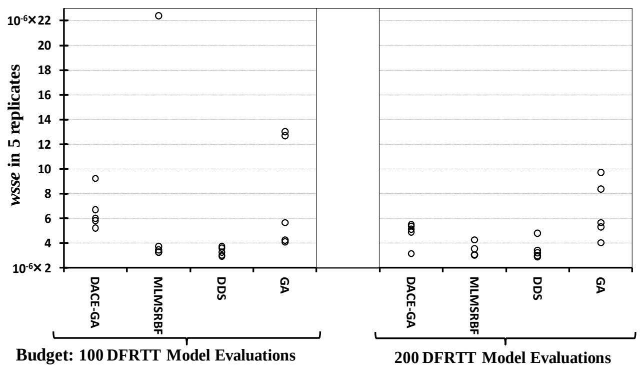


Figure 4-13. Algorithm performance comparisons on the DFRTT model – a minimization problem

4.6.7 FEFLOW Case Study

Only MLMSRBF and DDS were compared on this case study. As can be seen in Figure 4-14, DDS outperformed MLMSRBF in the two computational budget availability scenarios. As such, the performance of this metamodel-enabled optimizer lies in Case C when compared to the DDS performance. This case study is different from the test functions used and the other two real-world case studies in that it has a relatively large number of decision variables. Given the inherent weakness of metamodelling strategies in high-dimensional problems, the results of this case study is not included in the discussion in the next section.

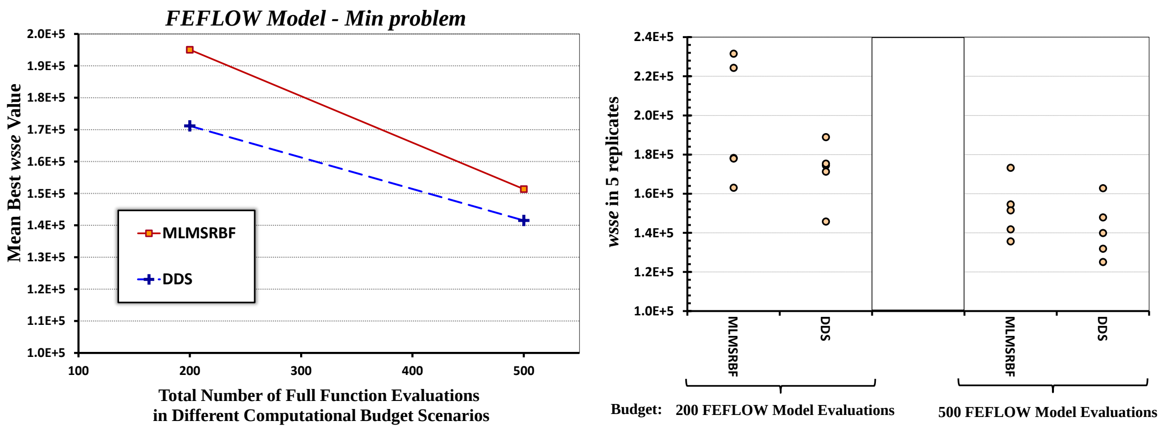


Figure 4-14. Algorithm performance comparisons on the FEFLOW model – a minimization problem

4.7 Discussion

Metamodelling is *not always* a solution to computationally intensive optimization problems. As observed, a given metamodelling strategy can sometimes be completely misleading and a waste of computational budget (see e.g., DACE-GA performance on the Ackley function or MLMSRBF performance on the Schwefel function). In other cases, they can be very effective and efficient (see e.g., the performance of DACE-GA and MLMSRBF on Griewank and MLMSRBF performance on the Ackley function). Experiments on the Rastrigin and Schwefel test functions as well as the SWAT and DFRTT case studies confirm that efficient optimizers without metamodelling, such as DDS/DDS with preemption, can outperform or perform equally well compared to metamodel-enabled optimizers. Note that with comparable performance, optimizers without metamodelling are preferred over metamodel-enabled optimizers, because they do not require the extra metamodelling computational burden (e.g., metamodelling time and analyst time, both of which were not explicitly evaluated in this study).

As the experimental results on the four test functions confirm, the more complex the original response surface, the less effective the metamodelling performance. In other words, when the original response surface is very complex, the metamodel inaccuracy may drive the search process towards poor regions in the search domain. This risk is higher especially when the original response surface has multiple geometrically distant regions of attraction (like the Schwefel function).

Importantly, it is likely that the relative performance of a metamodel-enabled optimizer depends on the allocated computational budget (Case B as discussed in Section 4.2.1 and shown in Figure 4-1). Therefore, computational budget availability (i.e., the total number of original function evaluations) is an important factor affecting the suitability and superiority of a metamodel-enabled optimizer over an optimizer without metamodelling. Typically, the quality of the final solution found through an optimizer without metamodelling is enhanced with a reasonably considerable improving rate as the computational budget availability increases; whereas, this improving rate with computational budget in a metamodel-enabled optimizer can be less if metamodel inaccuracies are substantial (see e.g., MLMSRBF on Schwefel's function or DACE-GA on Rastrigin's function). As a result and as can be seen in Figure 4-7, the performance plots at some computational budget may cross each other; this *"crossing behaviour"* (Case B) was observed 6 times (out of 24 one by one comparisons over the 6 case studies) between the metamodel-enabled optimizers and optimizers without metamodelling. For example, on the Rastrigin function, in the very limited budget scenarios (i.e., with 100 and 200 function evaluations) both DACE-GA and MLMSRBF were superior to GA, while for larger budget scenarios (i.e., with 500 and 1000 function evaluations) GA outperformed the metamodel-enabled optimizers. This result suggests that when the total number of function evaluations can be high (in cases when a relatively large computational budget is available or the original simulation model is not very computationally expensive), the optimizers without metamodelling may be more appealing. The results also confirm that equivalence time, t^*, in Case B is case study- and algorithm-specific and requires extensive numerical experiments to be determined.

The choice of the benchmark optimizer (without a metamodel) to which a metamodel-enabled optimizer is compared can have a large impact on relative performance conclusions. We selected DDS as an appropriate benchmark based on our experience developing and using the algorithm to solve computationally intensive optimization problems. We selected GA as another benchmark despite the fact that it is not an appropriate algorithm for limited computational budgets largely because it appears as a benchmark optimizer without metamodelling in multiple metamodel-enabled optimization studies [*Broad et al.*, 2005; *Fen et al.*, 2009; *Khu et al.*, 2004; *Ostfeld and Salomons*, 2005; *Zou et al.*, 2007]. If we consider the GA as the benchmark optimizer without metamodelling, relative performance of the metamodel-enabled optimizers (MLMSRBF and DACE-GA) in 6 out of 12 comparisons (12 = 6 case studies × 2 metamodel-enabled optimizers) are in Case A (idealized relative metamodel performance), 5 times in Case B (computational budget dependent relative metamodel performance), and only once in Case C (failure of metamodelling). However, when DDS/DDS with preemption is considered as the benchmark optimizer without metamodelling, the relative performance of the metamodel-enabled optimizers lies 3 times in Case A (out of 12 comparisons), once in Case B, and 8 times in Case C. Thus, using an inappropriate benchmark optimizer (the GA) for this study showed only an 8% failure rate of metamodelling versus a 67% failure rate of metamodelling when a more appropriate benchmark optimizer (DDS) was utilized.

We believe that future metamodelling studies evaluating relative metamodel-enabled optimizer efficiency and/or effectiveness should be selecting benchmark optimizers without metamodelling which are designed or demonstrated to work well with a limited computational budget. At least, instead of simply comparing a GA to metamodel-enabled optimizer performance, the GA parameters should first be tuned in a way to improve GA performance at the computational budget of interest (i.e., via multiple optimization test functions). A much better approach would be to benchmark the performance of a metamodel-enabled optimizer against metamodel independent algorithms designed to work relatively efficiently such as DDS/DDS with preemption as demonstrated herein or other algorithms such as the MicroGA technique [see *Nicklow et al.*, 2010 for discusion] or multistart derivative-based algorithms [e.g. *Doherty*, 2005]. Other algorithms we believe to be more effective than the GA such as CMA-ES [*Hansen et al.*, 2003] or AMALGAM [*Vrugt et al.*, 2009a] might be good additional algorithms in the comparison and in particular for computational budgets above 1000 function evaluations.

Numerical experiments in this study also suggest that none of our metamodelling implementations work best on all problems and all computational budgets, as MLMSRBF performed better than DACE-GA in some cases (e.g., on the Ackley function in all computational budget scenarios and in the Rastrigin function in larger budget scenarios with 500 and 1000 function evaluations) and worse in some other cases (e.g., on the Schwefel function and SWAT in all computational budget scenarios). Furthermore, the number of times that the performance of MLMSRBF relative to GA lies in Case A, Case B, and Case C is 3, 3, and 0, respectively, and relative to DDS these numbers are 2, 0, and 4. Similarly for DACE-GA relative to GA, the numbers of times in Case A, Case B, and Case C are 3, 2, and 1, respectively, and for DACE-GA relative to DDS, the same figures are 1, 1, and 4.

Figures 4-8 to 4-13 show the dispersion of the best found solutions across the case studies and confirm the essential need of having multiple replicates in the (comparative) assessment of optimizing algorithms despite the obvious extreme computational burden that results with computationally intensive optimization problems. As an example, consider if the performance of MLMSRBF in the 100 function evaluations scenario in Figure 4-13 was judged only based on the highly inferior replicate instead of all the five replicates (four other solutions are all of much higher quality).

4.8 Conclusions

To provide the metamodelling practitioners with a clear view of metamodelling characteristics, benefits and shortcomings, this study conducted a numerical assessment of three well-established metamodelling strategies involving radial basis functions, kriging and neural networks as metamodels within different computational budget availability scenarios on four commonly used test functions and three real-world water resources case studies. DDS/DDS with preemption and a GA were used as the benchmark optimizers without metamodelling to provide the baseline for assessment. The results clearly show that developing a new or approximately replicating an existing metamodel-enabled optimization framework is not enough to warrant the assumption that such a product is a more effective approach than optimizers without metamodelling. In fact, our results show multiple instances where optimizers without metamodels clearly outperform metamodel-enabled optimizers. Such a result has been rarely reported in literature; nonetheless, it is consistent with *Willmes et al*. [2003] who developed two metamodel-enabled optimizers and applied them to three test functions and concluded that "Neither the kriging model nor the neural network could clearly demonstrate an advantageous performance over an evolutionary optimization without metamodels. Very often, optimization assisted with a metamodel leads to a degraded performance." With these metamodel-enabled optimization failures relative to optimization without metamodels in mind, we hope this study motivates similar robust comparisons between new metamodel-enabled optimizers and optimizers without metamodelling so as to better focus metamodelling research on only the most promising metamodel-enabled optimization frameworks.

This study proposed a comparative assessment framework which presents a clear computational budget dependent definition for success/failure of the metamodelling strategies. Analyzing our results within such a framework empirically demonstrated that the suitability and superiority of metamodelling strategies can be affected by computational budget availability (i.e., the total number of original function evaluations). For example, the success/failure characterization of metamodelling strategies often changed in our results when computational budgets were varied between 100 to 1000 original model evaluations. Typically, the likelihood that a metamodel-enabled optimizer outperforms an optimizer without metamodelling is higher when a very limited computational budget is available; however, this is not the case when the metamodel is a neural network. In other words, neural networks are severely handicapped in limited computational budgets, as their effective training typically requires a relatively large set of design sites, and thus are not recommended for use in these situations.

Furthermore, the numerical results confirmed that metamodelling is an effective and efficient approach when the original response surface is relatively simple. However, the metamodelling performance degrades considerably in case of more complex original response surfaces. In addition, our results do not identify a single preferred metamodelling strategy between the MLMSRBF and DACE-GA metamodel-enabled optimizers. In practice, as the general form and the level of complexity of a simulation model response surface is not usually known *a priori*, any decision on the appropriate type of metamodelling strategy as well as any prediction about its performance are non-trivial.

Future research efforts could be focused on predicting the success/suitability of metamodel-enabled optimizer for example based on preliminary experiments measuring metamodel accuracy and generalizability (e.g., through the initial DoE in Figure 4-3). Conceptually, we believe metamodel-enabled optimizers can be developed that would almost always show computational budget dependent relative performance (Case B in Figure 2) such that they would be preferred over any metamodel independent optimizer for at least some limited range of reduced computational budgets. Therefore, future research should continue to strive for improved metamodel-enabled optimization algorithms.

Chapter 5
A Framework Utilizing Surrogate Calibration Data for Model Calibration on Long Data Periods

This chapter is a mirror of the following article which is currently under review. Only minor organizational changes were made to be consistent to the body of the thesis. References are unified at the end of the thesis.

Razavi, S. and B. A. Tolson (2012), An Efficient Framework for Hydrologic Model Calibration on Long Data Periods, submitted to *Water Resour. Res*. in December 2012.

Summary

Long periods of hydrologic data records have become available in many watersheds around the globe. Hydrologic model calibration on such long, full-length data periods are typically deemed the most robust approach for calibration but at larger computational costs. Determination of a representative short period as a “surrogate” of a long data period that sufficiently embeds its information content is not trivial and is a challenging research question. The representativeness of such a short period is not only a function of data characteristics but also model dependent. Unlike previous studies, this study goes beyond identifying the best surrogate data period to be used in model calibration and proposes an efficient framework that calibrates the hydrologic model to full-length data while running the model only on a short period for the majority of the candidate parameter sets. To this end, a mapping system is developed to approximate the model performance on the full-length data period based on the model performance for the short data period. The basic concepts and the promise of the framework are demonstrated through a computationally expensive hydrologic model case study. Three calibration approaches, namely calibration solely to a surrogate period, calibration to the full period, and calibration through the proposed framework, are evaluated and compared. Results show that within the same computational budget, the proposed framework leads to improved or equal calibration performance compared to the two conventional approaches. Results also indicate that model calibration solely to a short data period may lead to a range of performances from poor to very well depending on the representativeness of the short data period which is typically not known a priori.

5.1 Motivation and Objective

The appropriate length of observation data for effective calibration of continuous hydrologic models has been a challenging research question [*Andreassian et al.*, 2001; *V K Gupta and Sorooshian*, 1985; *Juston et al.*, 2009; *Perrin et al.*, 2007; *Singh and Bardossy*, 2012; *Vrugt et al.*, 2006b; *Xia et al.*, 2004; *Yapo et al.*, 1996]. In general, longer calibration periods are deemed more robust and reliable in identifying the model parameters and quantifying their uncertainty [*Perrin et al*., 2007]. However, longer periods would demand longer model run-times; given that in the calibration process a hydrologic model is required to run a large number of times, utilizing the full period of data available for model calibration and uncertainty estimation may sometimes become computationally demanding or even infeasible. Such computational burdens may become prohibitive for some modern high-fidelity hydrologic models which simulate detailed representations and processes of the real-world system. As such, computational limit is typically one motivation of seeking a representative "short period" out of longer available data for model calibration in any watershed of interest. A representative short period needs to embed diverse climatic and hydrologic conditions to adequately represent hydrologic variability in the watershed of interest such that all simulated processes in the model are well-activated [*Juston et al*., 2009; *Perrin et al*., 2007; *Singh and Bardoss*y, 2012]. There have been numerous research studies attempting to determine the minimum adequate length of data for hydrologic model calibration. These studies typically calibrated and compared model parameters for different sub-periods with different lengths of a long data period and concluded that several years of data could be adequate but their proposed required lengths vary relatively widely in the range of two to eight years [*Juston et al*., 2009]. *Singh and Bardossy* [2012] pointed out that any of the proposed required lengths cannot be generalized as different models have different complexity levels and different watersheds have different information content in each year of data period. In addition, *Xia et al*. [2004] found that different model parameters may require different lengths of data to achieve calibration results insensitive to the period selected. Moreover, the information contained in observation data is not uniformly distributed along the period, and certain sub-periods may contain information useful for identifying some specific parameters while irrelevant for other parameters [*Singh and Bardossy*, 2012]. As a result, there is a tendency among hydrologists to utilize the full period of available data for calibration of hydrologic models despite possible computational burdens.

This chapter develops a computationally efficient framework for hydrologic model automatic calibration for watersheds with long periods of available data. This framework aims to utilize full-length data periods in model calibration (called “full period” hereafter) while reducing computational burdens by using representative short periods for most of the model runs and running the model over the full period *only* for selected candidate parameter sets. Most of these candidates are identified to be promising solutions based on the information obtained from model runs on a short period. The proposed framework has roots in surrogate modelling which is concerned with developing and utilizing more efficient models as *surrogates* of computationally intensive high-fidelity (original) models (see Chapter 3). There are two general families of surrogate modelling strategies: 1- response surface surrogates which utilize statistical or data-driven function approximation methods such as kriging and neural networks to emulate one (or multiple) response(s) of an original model [*Broad et al.*, 2005; *Johnson and Rogers*, 2000; *Regis and Shoemaker*, 2007b; *Yan and Minsker*, 2011], and 2- lower-fidelity physically-based surrogates which are simplified models of the original system of interest and conceptually or directly preserve its governing physics [*Forrester et al.*, 2007; *Mondal et al.*, 2010; *Robinson et al.*, 2008]. The proposed framework in this study shares some features with lower-fidelity physically based surrogate modelling when used for model calibration. However, a major difference is that instead of using surrogate models, the proposed framework solely works with original high-fidelity models but defines and uses “surrogate data” (i.e., short period data).

The organization of this chapter is as follows. Section 5.2 presents the basic concepts and principles behind the proposed framework. Section 5.3 introduces a computationally demanding hydrologic model calibration case study that is used to illustrate the concepts and methodology. Section 5.4 presents the proposed framework and the components involved. Results of the numerical experiments with this framework in presented in Section 5.5. Section 5.6 is dedicated to outline the common features and differences between the proposed framework and surrogate modelling strategies. This paper ends with conclusions in Section 5.7.

5.2 Basic Concepts

The proposed framework is based on the principle that the performances of a hydrologic model on different data periods are correlated. The magnitude of such correlations varies for different pairs of data periods from very low (perhaps even negligible) for periods with distinctly different hydrologic conditions to very high for periods experiencing similar conditions. Assuming the performance of a hydrologic model is measured by one error function (e.g., mean squared errors), Figure 5-1 compares the true error function of a hydrologic model (i.e., running on a full calibration period) with three possible surrogate error functions of the same model when running on three different short periods. In this conceptual example, the error function to be minimized is assumed to be dependent on only one model parameter. Figure 5-1a represents a case where the short period is a good (almost ideal) representative of the full period such that there is a high level of correlation (agreement) between the model performance over the full and short periods; there may be still a reasonably small displacement in the optimal parameter value. Figure 5-1b shows a moderate level of correlation where the model performance over the short period almost captures both modes (local and global optima) of the model performance over the full period but the global optimum in the short period performance corresponds to the local optimum of the performance on the full period. The short periods represented in Figure 5-1a and Figure 5-1b can be very helpful in calibrating the hydrologic model. Conversely, Figure 5-1c represents a case where the short period is quite misleading as there is a large level of discrepancy between the two functions. This case may happen when the surrogate data period is unreasonably short and/or when it does not include information that can activate most of the simulated processes that are active in the full period.

Figure 5-2 hypothetically depicts possible relationships between the performance of a hydrologic model on a full data period (i.e., full-period error function - e.g., mean squared errors) versus the performance of the same model on a short data period (i.e., short-period error function). In general the full-period error function value is monotonically increasing (linearly or non-linearly) with the short-period error function value (the global relationship – see Figure 5-2a). Such a relationship can be very useful in predicting the performance of a model on the full period using the results of a model simulation over a short period and is the basis of the framework developed in the study. However, there is some level of uncertainty (i.e., discrepancy from the global relationship) associated with this function; this uncertainty becomes larger for less representative short periods. When comparing two parameter sets on the basis of their performance on a short period, if their short-period error function values are relatively close, it would not be trivial to identify the parameter set with better full-period error function value through this global relationship due to this uncertainty (see e.g., the sample of local relationship in Figure 5-2a). Figure 5-2b demonstrates a case where the short period is very poor such that no clear global relationship between the full-period and short-period error functions is identifiable – the uncertainty level is excessively large. Conceptually, Figure 5-2a corresponds to the cases depicted in Figure 5-1a and Figure 5-1b, while Figure 5-2b corresponds to the case demonstrated through Figure 5-1c.

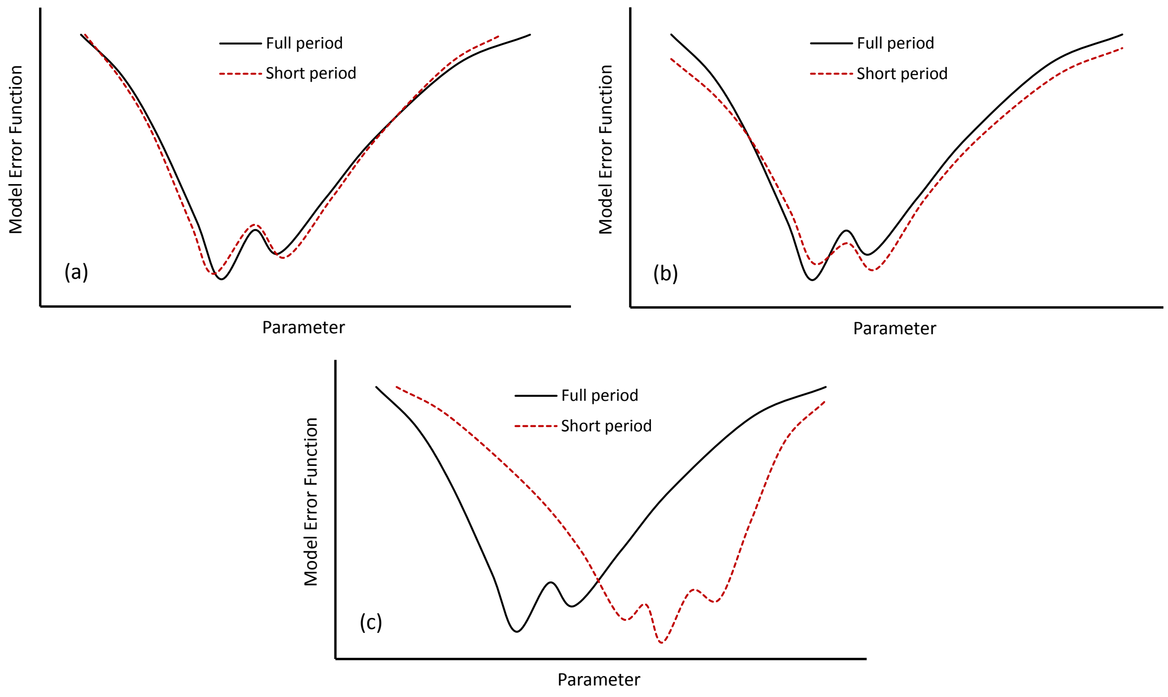


Figure 5-1. Conceptual examples of the performance of a hydrologic model (with one parameter) running over different short calibration periods out of a long data period (full period) along with the model performance over the full period: (a) a well representative short period, (b) a moderately representative short period, and (c) a poor (misleading) short period

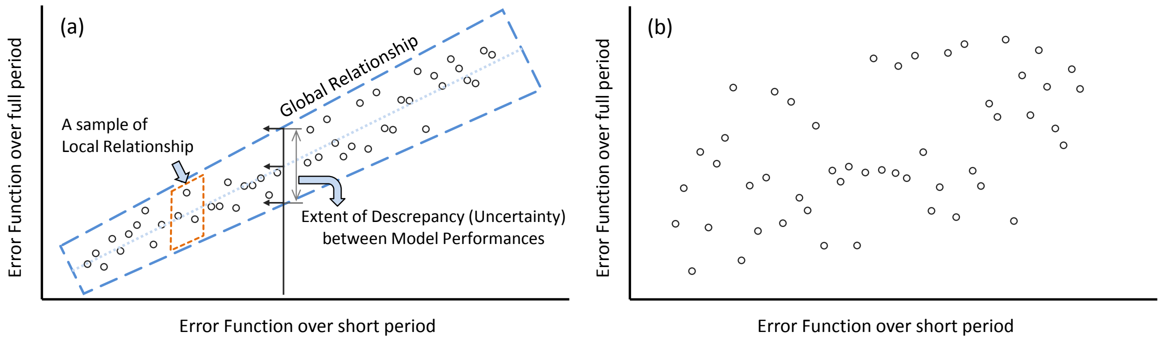


Figure 5-2. Hypothetical relationship examples between the performance of a hydrologic model over a full period and the performance of the same model over (1) a representative and (2) a poor short periods – each point in the scatter plots represents a random parameter set that is evaluated by the model over the different periods

The framework proposed in this thesis utilizes such relationships in an attempt to efficiently calibrate hydrologic models to full data periods. Note that if a short period is a perfect surrogate of a full period, solely calibrating a hydrologic model to the short period would be accurate. Figure 5-3 depicts a conceptual (but presumably typical) example of the convergence trend of an automatic calibration experiment when the optimization objective function to be minimized is the short-period error function. The corresponding full-period error function is also shown in this figure - the parameter sets found on the optimization convergence trajectory are also evaluated by running the model over the full period. As depicted in Figure 5-3, early on in the search, reducing the short-period error function would result in reducing the full-period error function as well; however, after some point in the search, further reduction in the short-period error function would result in increasing the full-period error function. This behaviour, which is analogous to “over-fitting” in statistics and curve-fitting, indicates that solely relying on a representative short period may be efficient to initially guide a calibration algorithm to the vicinity of the promising regions in the parameter space, but it may become ineffective or even misleading in extracting near optimal parameter sets. The proposed framework explained in Section 5.4 is designed to address such behaviour.

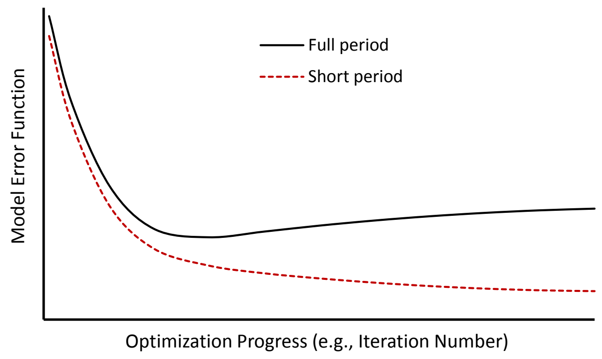


Figure 5-3. A hypotetical example of optimization convergence trend when minimizing the model error function over a short period of data along with the coresponding error function values calculated over the full period

5.3 An Example Case Study

To illustrate the framework proposed, a SWAT2000 (Soil and Water Assessment Tool, version 2000) [*S. L. Neitsch et al.*, 2001] hydrologic model calibration case study, originally developed by *Tolson and Shoemaker* [2007a] to simulate streamflows into the Cannonsville Reservoir in New York, is used. This automatic calibration case study seeks to calibrate 14 SWAT2000 model parameters, subject to various range constraints [*Tolson and Shoemaker*, 2007b, see Table 2 therein], to measured flow at the Walton gauging station (drainage area of 860 km^2) by maximizing the Nash-Sutcliffe coefficient for daily flow. A total of nine years of data from January 1987 to December 1995 are used in simulations such that the first three years are used as the initialization period and the remaining six years are used as the calibration period (see Figure 5-4). In this case study, a single model run over the nine-year period requires about 1.8 minutes on average to execute on a 2.8GHz Intel Pentium processor with 2 GB of RAM running the Windows XP operating system.

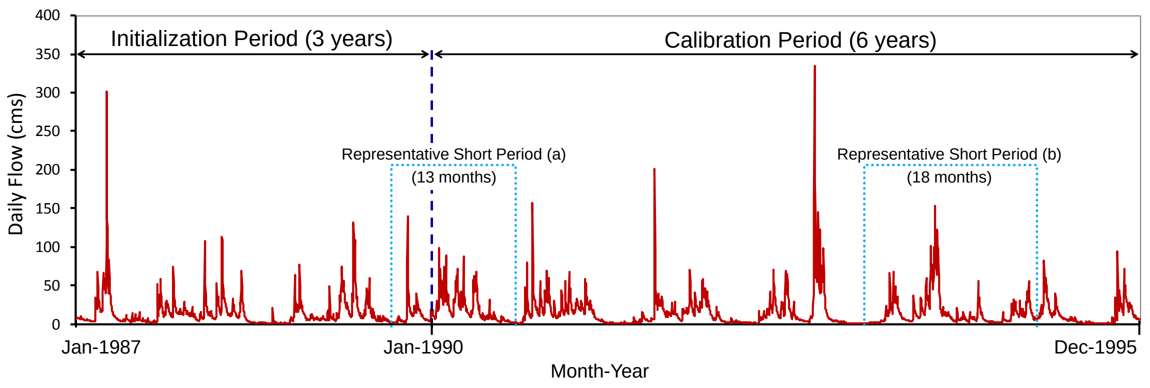


Figure 5-4. Streamflow time series at Walton gauging station in Cannonsville Reservoir watershed over a nine-year period used for SWAT model calibration

5.4 Methodology

The framework developed in this study links an optimization algorithm to a hydrologic simulation model which is set up to run over a full period of data and a representative short period of data (see Figure 5-5). Any optimization algorithm may be used in the framework; however in this study, the covariance matrix adaptation-evolution strategy (CMA-ES) [*Hansen and Ostermeier*, 2001] is used for demonstration. CMA-ES is a stochastic, derivative-free optimization algorithm that generates candidate solutions according to an adaptive multivariate normal distribution. The covariance matrix of this distribution which represents pairwise dependencies between variables is updated iteratively in the course of optimization. Notably, CMA-ES does not need the actual objective function values of the candidate solutions, and instead it only uses their rankings to learn the specifications of the multivariate normal distribution. The algorithm has the form of (μ, λ)-CMA-ES where μ and λ are the parent number and population size, respectively. At each iteration, λ new candidate solutions are generated following the multivariate normal distribution, and then a weighted combination of the μ

best out of λ solutions is used to adapt the parameters of the distribution (i.e., mean and covariance matrix) such that the likelihood of previously successful candidate solutions and search steps is maximized/increased. Following the recommendations of *Hansen and Ostermeier* [2001] for algorithm parameters, (5, 11)-CMA-ES is used for our case study.

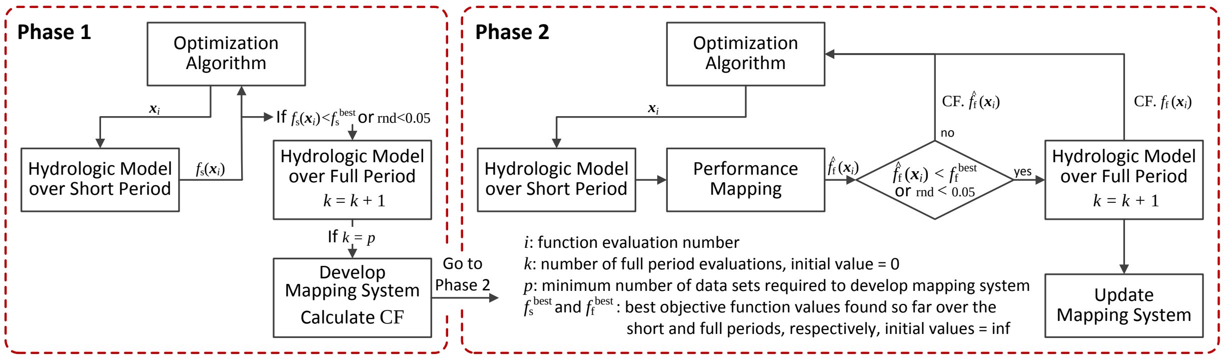

Phase 1
Optimization Algorithm
x_i
If $f_s(x_i)<f_s^{best}$ or rnd<0.05
Hydrologic Model over Short Period
$f_s(x_i)$
Hydrologic Model over Full Period $k = k + 1$
If $k = p$
Develop Mapping System Calculate CF
Go to Phase 2
Phase 2
Optimization Algorithm
x_i
CF. $\hat{f}_f(x_i)$
CF. $f_f(x_i)$
no
Hydrologic Model over Short Period
Performance Mapping
$\hat{f}_f(x_i)$
$\hat{f}_f(x_i) < f_f^{best}$ or rnd < 0.05
yes
Hydrologic Model over Full Period $k = k + 1$
Update Mapping System
i: function evaluation number
k: number of full period evaluations, initial value = 0
p: minimum number of data sets required to develop mapping system
f_s^{best} and f_f^{best} : best objective function values found so far over the short and full periods, respectively, initial values = inf

Figure 5-5. The proposed framework for efficient model calibration over a long period of data by utilzing representative short periods (CF stands for Correction Factor)

5.4.1 Framework

The framework developed in this study consists of two phases (see Figure 5-5). Phase 1 is basically a conventional automatic calibration procedure which simply links the optimization algorithm to the hydrologic model but running over the representative short period. As such, the optimization algorithm generates candidate parameter sets, $\boldsymbol{x}_i$, for the hydrologic model with the objective of minimizing the short-period error function, $f_s(\boldsymbol{x}_i)$. In phase 1, only the parameter sets that improve the current best short-period error function value (i.e., f_s^{best}) are also evaluated through the hydrologic model over the full period of data. Phase 1 continues until a sufficient number of parameter sets are evaluated over both short and full periods; the simulation results of these parameter sets are then used to develop the performance mapping system. The performance mapping system enables the framework to map the performance of new candidate parameter sets on the short period (short-period error function) to their corresponding performance on the full period (i.e., full-period error function). Details of the proposed mapping system are presented in Section 5.4.3. Notably, phase 1 is only active early in the search when minimizing the short-period error function mainly corresponds to minimizing the full-period error function (falling part of the full-period error function in Figure 5-3).

In phase 2, the optimization algorithm seeks the minimum of a new response surface which is a surrogate of the full-period error function such that the majority of candidate parameter sets are still only evaluated by the hydrologic model over the short period, but the approximated full-period error function values (by the mapping system) are returned to the optimizer. If the approximated full-period error function value for a candidate parameter set is better than the current best full-period error function value (i.e., $f_{\mathrm{f}}^{\mathrm{best}}$), the parameter set is also evaluated by running the hydrologic model over the full period.

Switching from one objective function surface to a different one when moving from phase 1 to phase 2 may corrupt some of the information the optimization algorithm has collected and as such degrades the optimizer performance. For example, the mean and covariance matrix of the multivariate normal distribution, which have evolved in the course of a CMA-ES run in phase 1, may change abruptly at the beginning of phase 2 if it was to directly minimize the new surface function. Defining the "correction factor" at the end of phase 1 is intended to address the discrepancies between the two surfaces. The correction factor defined herein is inspired and acts similar to multiplicative correction functions that are commonly used in lower-fidelity surrogate modelling [*Alexandrov and Lewis*, 2001; *Razavi et al.,* 2012b; *Thokala and Martins*, 2007]. Moreover, depending on the optimization algorithm used, there may be other conditions to be accounted for to better preserve the optimization algorithm behaviour. For example, in population-based optimization algorithms like CMA-ES, it is advisable to evaluate all the individuals in a generation by the same function. To this end, the framework developed only proceeds to phase 2 at the beginning of a new a CMA-ES generation. The correction factor, CF, is calculated as:

$$\mathrm{CF} = {\overline{f}_{\mathrm{s}}}\big/{\overline{f}_{\mathrm{f}}} \tag{5-1}$$

where $\overline{f}_{\mathrm{s}}$ is the best short-period error function value in the last population of the optimization algorithm in phase 1 that its corresponding full-period error function value, $\overline{f}_{\mathrm{f}}$, is available.

Both phase 1 and phase 2 reserve a small chance for any candidate parameter set to be evaluated over the full period regardless of its performance over the short period. This chance is controlled by a number varying in the range of zero and one, zero indicating no chance and one indicating all parameter sets are to be also evaluated over the full period. This capability mainly intends to minimize the bias of the performance mapping system by providing a range of possible performances.

Our preliminary experiments with the number of zero demonstrated that generating a biased mapping system was possible and that could lead to a framework failure. In addition, this capability diminishes the risk of possible framework failures in case the short period selected is not very informative. Larger numbers would increase the sample size of data used for developing the mapping system and make the framework more robust but this would come at the expense of increasing the overall computational burden. The optimal number is problem-specific and not known without extensive experimentation. This number was arbitrarily selected to be 0.05 in the experiments of this study.

5.4.2 What is a representative short period?

Ideally, a short period of the full data period is deemed “representative” if the information contained in the short period is sufficiently complete and diverse such that model calibration experiments using the short period and the full period would yield almost identical results. Such an ideal representative period with a short duration (reasonably shorter than the full data period) may not necessarily exist in any watershed. In this study, “representative short period” refers to a reasonably short sub-period (contiguous in time) of the full period of data that contains sufficient information to be used in the proposed framework. *Singh and Bardossy* [2012] proposed a method to identify and extract some type of representative short periods, what they referred to as “unusual events”, out of the full data period based on the statistical concept of data depth. They demonstrated that calibration of a hydrologic model to unusual events may be only slightly inferior to the calibration of the same model to full periods of available data. According to *Singh and Bardossy* [2012], unusual events may include 1- periods containing extremes, 2- periods with low flows, 3- periods with strong dynamics (e.g., droughts ending with heavy rainfall), and 4- periods with temporal intermittence. They also pointed out that unusual events can be only a small portion of the full data periods (less than 10 percent). Although the method of *Singh and Bardossy* [2012] can be potentially used with the proposed framework, there are constraints that may limit its applicability in this framework. First of all, the unusual events defined in the work of *Singh and Bardossy* [2012] may be far apart along the full calibration period. In such cases, it is not possible to have a reasonably short period contiguous in time including all the unusual events. Moreover, as these events are “unusual”, there is typically minimal relationship between the model performance on these events and the model performance on the full calibration period.

In this study, a representative short period is a period over which the model performance is reasonably well correlated with the model performance over the full period. For demonstration, two different parts of the full period of data available for our case study were selected as representative short periods (see Figure 5-4 and Figure 5-6). The representative short period (a) was selected arbitrarily without any prior knowledge/analysis of the system to represent a case when the framework user cannot identify the most appropriate representative short period. The first four months of this 13-month period was used for model initialization. A preliminary analysis was conducted to evaluate the correlation of the hydrologic model performance over this short period with the performance of the same model over the full period. A total of 1000 (uniformly) random parameter sets were generated in their specified ranges and evaluated through running the hydrologic model over the full period. Values of sum of squared errors (deviations of the simulated flows from observations, SSE) were calculated over the full calibration period (i.e., six years) and the last nine months of the representative short period (a) and plotted in Figure 5-7. According to this figure, the linear correlation between the SSE values over the two periods is not very large and the linear regression has an R^2 value of 0.64. The lack of high correlation would imply that the error function surfaces for the two periods are somewhat dissimilar and the minimizer of one may not be close to the minimizer of the other. Utilizing the simulation results of the 1000 random parameter sets, the representative short period (b) was selected such that the linear correlation between the short-period SSE values and the full-period SSE values was maximized (see Figure 5-7) - a 12-month window was moved along the six years of data to find a linear regression with maximum R^2 value (i.e., 0.94). In general, higher correlations indicate higher degrees of similarity between the short-period and full-period error function surfaces. When the two surfaces are very similar, model calibration on the short period would be almost equivalent to calibration on the full period. As such, representative short period (b) may be deemed the best one-year-long surrogate of the full data period. As explained in Section 5.4.3, the performance mapping system attempts to address and formulate the dissimilarities between the two surfaces such that any short period demonstrating a moderate level of correlation with the full period, such as representative short period (a), can be used in the proposed framework.

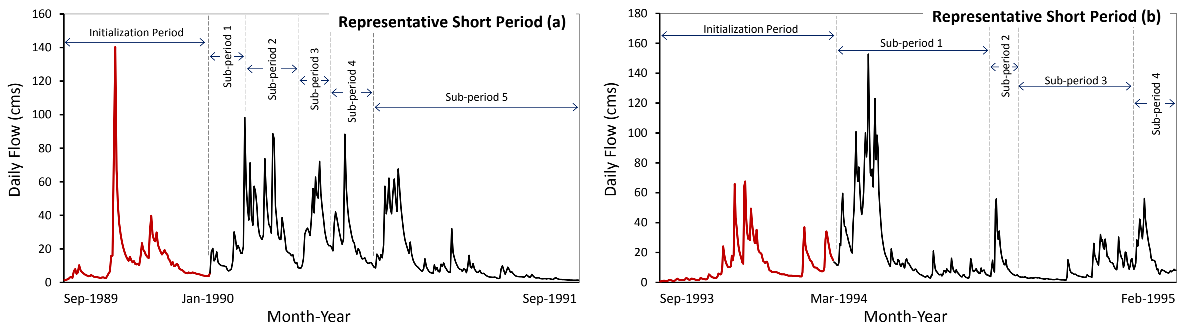


Figure 5-6. Two different representative short periods of the full period of data and the sub-periods used for performance mapping

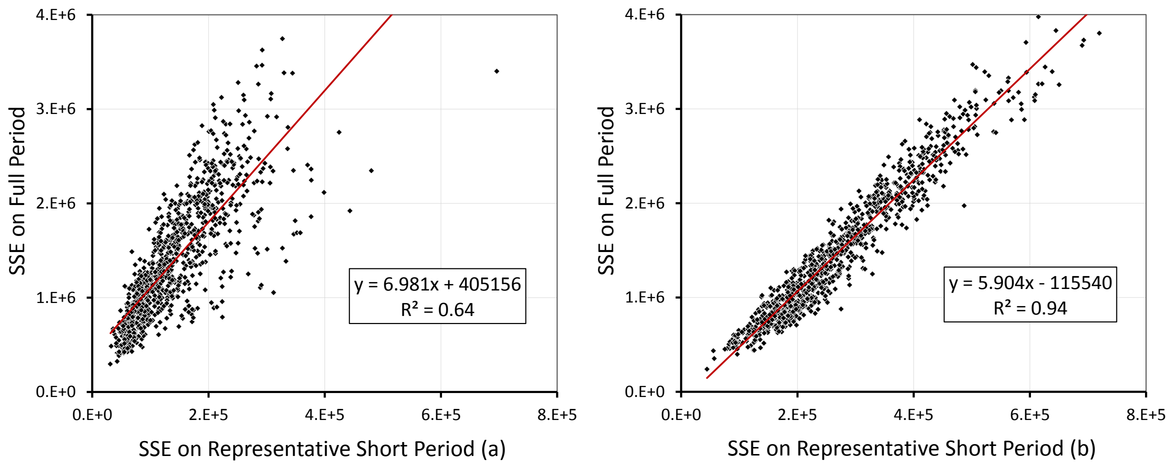


Figure 5-7. Scatter plots of the sum of squared errors (SSE) calculated over the full period versus the SSE calculated over the representative short periods for 1000 randomly generated parameter sets

5.4.3 Performance Mapping

The performance mapping system developed in this study is on the basis of the relationships conceptualized in Figure 5-2 and numerically derived for our case study in Figure 5-7. However, instead of considering the short-period error function (e.g., MSE value over the short period) as one single predictor, the mapping system disaggregates this function value into multiple values corresponding to different sub-periods of the representative short period. Each sub-period is to represent a distinct hydrologic behavior/aspect of the data, for example, flood events, recession periods, dry periods, etc. This disaggregation enables the mapping system to extract the most of the information contained in a representative short period. Figure 5-6 shows the sub-periods used in the representative short periods of our case study. The performance mapping system is to relate the performance of the model over all these sub-periods to the performance of the model over the full period of data. The performance mapping system consists of a regression model enhanced by a localized Gaussian deviation model, $Z(.)$, based on the kriging concept in the form of:

$$g_{\mathrm{f}}(x_i) = \sum_{j=1}^{n} \beta_j g_j(x_i) + Z\big(g_1(x_i), \dots, g_n(x_i)\big) \tag{5-2}$$

where $\boldsymbol{x}_i$ is the ith candidate parameter set, $g_j(\boldsymbol{x}_i)$ is the model error function value for sub-periods j, n is the number of sub-periods, and $g_{\mathrm{f}}(\boldsymbol{x}_i)$ is the full-period error function. We considered $g_j(\boldsymbol{x}_i) = \log(\mathrm{MSE}_j)$ where MSE_j is the mean squared of model errors over the sub-period j and $\hat{f}_{\mathrm{f}} = \exp\big(g_{\mathrm{f}}(\boldsymbol{x}_i)\big)$ where $\hat{f}_{\mathrm{f}}$ is the approximated MSE value over the full period. The logarithmic form is used mainly to generate a mapping system more sensitive and accurate for smaller MSE values. In automatic calibration, accuracy of the mapping system becomes more important when the search approaches the main regions of attractions (regions containing parameter sets with reasonably small error functions).

The regression-Gaussian model in equation 5-2 was adopted from *Sacks et al.* [1989] and implemented with the software package developed by *Lophaven et al.* [2002]. This mapping function represents the global relationships by the regression component, while the Gaussian component attempts to represent the residuals in the regression and also the non-linear local relationships. The minimum number of input-output data sets that is mathematically required to develop the mapping function and provide confidence intervals is n+2 (n is the dimension of the function input space). Interested readers are referred to *Sacks et al.* [1989] and *Lophaven et al.* [2002] for details on how to develop and fit the model. Notably, unlike regression models, the mapping system developed is an exact interpolator for the input-output sets used for developing/updating the system – the system would generate the exact model performance (i.e., full-period MSE values) for the parameter sets that have been already evaluated over the full data period.

To select multiple sub-periods out of each representative short period, (a) and (b), shown in Figure 5-6, the residual time series generated for 1000 random parameter sets in Section 5.4.2 were used. All possible configurations of division points in time for different numbers of sub-periods ranging from 2 to 6 were evaluated for both representative short periods by fitting the mapping system with only the regression component ($Z(.)$ was assumed to be constant). The sub-period configurations that maximized the regression R^2 value were selected. In our case study, the representative short periods (a) and (b) were disaggregated into 5 and 4 sub-periods, respectively. Notably, all the division points of the configurations for sub-periods which generated high R^2 values coincide with either the lowest or peak flows. We noticed that for a given representative short period, there are many configurations for sub-periods starting or ending at peaks or lowest flows that can generate a reasonably reliable regression model (i.e., with high R^2).

To demonstrate the accuracy of the developed mapping system, 1000 new parameter sets were generated randomly in their specified ranges and then the model was run for each parameter set over the full period and the representative short periods (a) and (b) independently. For each representative short period, the mapping system was developed using 100 of these generated input-output sets (selected randomly) and then tested over the remaining 900 input-output sets. Figure 5-8 shows the scatter plots of the actual full-period sum of squared errors (SSE) versus the predicted full-period SSE values for the 900 testing parameter sets. As can be seen, the results of the mapping system are almost unbiased as the linear regressions plotted on this figure are very close to the ideal (1:1) line. The dispersity of the points around the ideal line is also reasonably low for both representative short periods as the R^2 values are 0.92 and 0.96 for the representative short periods (a) and (b), respectively.

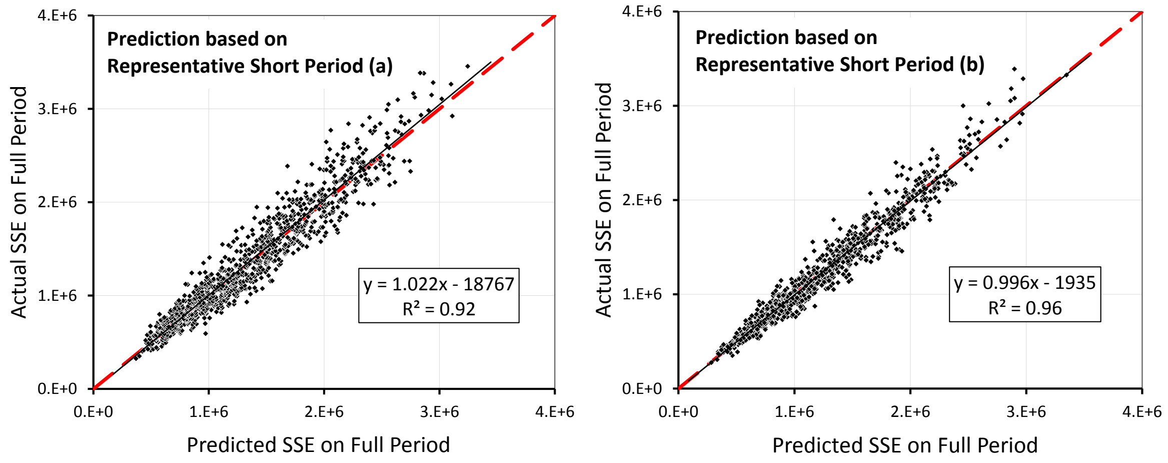
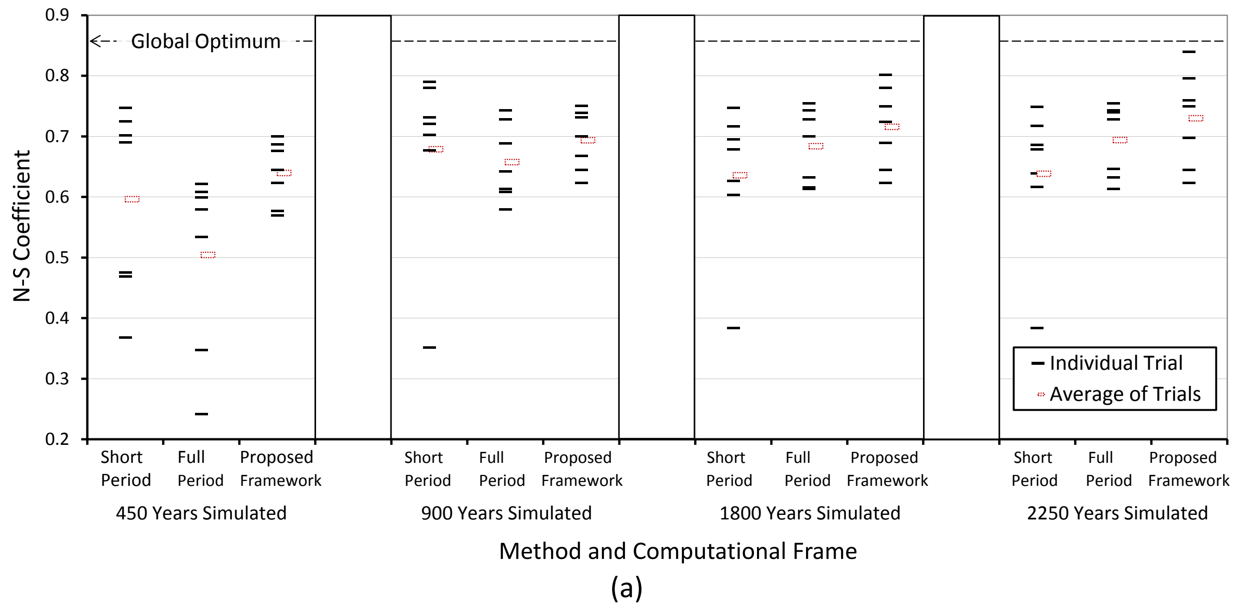


Figure 5-8. Scatter plots of the actual sum of squared errors (SSE) values over the full period versus the predicted SSE values for the same data period for 900 randomly generated parameter sets – the mapping (prediction) system is developed based on the model performance results over 100 randomly generated parameter sets (independent of the 900 parameter sets)

5.5 Numerical Experiments and Results

The three different approaches to model calibration, namely calibration solely on a representative short period, calibration on the full period of data, and calibration through the proposed framework, were evaluated and compared. Seven independent calibration experiments were run for each approach with seven different random seeds and random starting points. The comparisons were made within four different computational budget scenarios. Computational budget was quantified by the total number of years that were simulated by the hydrologic model over the course of each automatic calibration experiment and budgets of 450, 900, 1800, and 2250 years were considered. The smallest computational budget is equivalent to 50, 415, and 300 hydrologic model runs (optimization function evaluations) over the full period, representative short period (a), and representative short period (b), respectively. The computational budget of 2250 years of simulation is equivalent to 250, 2075, and 1500 objective function evaluations with full period, representative short period (a), and representative short period (b), respectively.

Figure 5-9 shows the results of the numerical experiments. The maximum Nash-Sutcliffe coefficient value (referred to as "global optimum" in this figure) for this case study that has been previously identified through extensive optimization is 0.85. As can be seen, model calibration on the full period is improving with the increase in the computational budget. However, model calibration solely on representative short period (a) is not as robust, since the performance results of the calibrated model when tested over the full period are more variable, and more importantly, are not improving (may be even degrading) with the increase in computational budget. Such degradations are obvious between the computational budgets of 900 and 1800 years of simulation as the average N-S value degrades from 0.68 to 0.63. This degradation is the result of over-fitting as illustrated in Section 5.2 and Figure 5-3. Moreover, in one experiment, model calibration on representative short period (a) failed to find a parameter set with an N-S value of greater than 0.4. The proposed framework with representative short period (a) performed consistently well in all the computational budgets and outperformed the other two approaches in terms of the average performance over the seven calibration experiments. At the largest computational budget considered, the proposed framework with representative short period (a) was able to find a parameter set close to the global optimum.



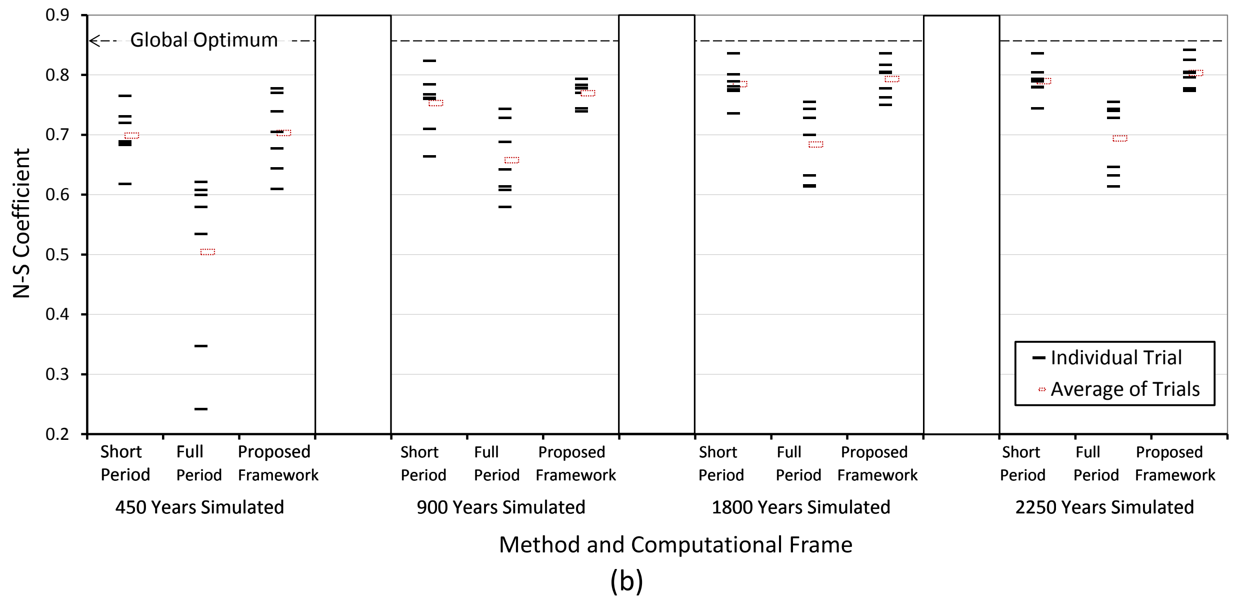


Figure 5-9. Results of the model calibration experiments with representative short period (a) and representative short period (b) through the three approaches: calibration solely on a short period, calibration on the full period, and the proposed framework – Seven independent calibration experiments for each approach were conducted, and Nash-Sutcliffe (N-S) coefficient over the full period of data was of interest at the end of each experiment.

Figure 5-9 also demonstrates that representative short period (b) is a near perfect surrogate of the full period as model calibration solely on this short period performed consistently well and better than calibration on full period in all computational budgets. The proposed framework with representative short period (b) also performed very well. In terms of the average N-S values over the seven calibration experiments, the proposed performance worked better or almost equally well at all computational budgets compared to the short-period calibration performance. Moreover, the range of N-S values obtained by the proposed framework is smaller than the corresponding ranges obtained by the two other approaches in all the computational budgets except the smallest one.

The highest computational budget considered was not sufficient to enable the conventional full-period calibration to approach the global optimum (the maximum N-S value found with this approach is 0.75). Therefore, to further evaluate and compare the performance of this approach, its seven calibration experiments were continued with a total budget of 1000 optimization function evaluations (i.e., equivalent to 9,000 years of simulation). The average, best, and worst N-S values found in these seven calibration experiments were 0.79, 0.83, and 0.76, respectively. Whereas, the proposed framework with representative short period (b) at the computational budget of 2,250 years of simulation attained the average, best, and worst N-S values of 0.80, 0.84, and 0.77. As such, one may conclude that model calibration on the full period with the budget of 9,000 years of simulation and the proposed framework with the budget of 2,250 years of simulation performed comparably.

The results of this experiment demonstrate that a properly-designed framework to utilize surrogate data may compensate for the inadequacy (lack of representativeness) of surrogate data in calibration practice. For “excellent” surrogate data, the role of this framework in successful calibration is minimal. But when highly appropriate surrogate data are not available or not easily identifiable, users can heavily rely on a properly-designed framework to reliably utilize less appropriate surrogate data to calibrate the computationally intensive model.

5.6 Relations to Surrogate Modelling

The framework proposed in this work shares certain features with strategies for lower-fidelity physically based surrogate modelling. Lower fidelity surrogates are simplified models which directly or conceptually preserve the physical processes in the real-world system. In general, there are many different ways with multiple subjective decisions to develop a lower-fidelity surrogate model for a given case study. For example in the hydrologic modelling context, a fully-distributed hydrologic model may be deemed the highest-fidelity model, while a semi-distributed model or a lumped model may be considered as lower-fidelity models. A low-fidelity hydrologic model may also ignore some physical processes simulated in a high-fidelity hydrologic model. One common challenge with any surrogate modelling strategy is the identification of the best way to develop a surrogate model for a given case study. The other challenge in developing and utilizing some (not all) lower-fidelity surrogate models is how to map their parameter space to the original model parameter space [*Bandler et al*., 2004; *Robinson et al.*, 2008]. Such a mapping is not always trivial and may introduce a great deal of uncertainty in a calibration experiment.

The proposed framework parallels the primary objective of surrogate modelling which is to circumvent the computational burden of optimization attempts involving computationally demanding models. However, instead of a lower-fidelity model as the surrogate, the proposed framework utilizes the original, high-fidelity model but runs it over a surrogate calibration period which is a representative short period of the full available data period. As such, the proposed framework evades the two aforementioned challenges as it preserves the original model parameters, complexity and processes simulated.

As outlined in Chapter 3 (also in *Razavi et al.* [2012b]), there are various methodologies in the literature to make use of surrogate models in model calibration and optimization. In some studies, surrogate models after being developed are treated as if they are high-fidelity representations of the real-world systems of interest and fully replace the original models [e.g., *McPhee and Yeh*, 2008; *Siade et al.*, 2010; *Vermeulen et al.*, 2005; *Vermeulen et al.*, 2006]. Such a strategy can be deemed analogous to the strategy for hydrologic model calibration which identifies and solely relies on a representative short data period. There are some other studies addressing the discrepancies between surrogate and original models [e.g., *Bandler et al.*, 2004; *Cui et al.*, 2011; *Forrester et al.*, 2007; *Robinson et al.*, 2006]. Similar to the proposed framework, these studies develop strategies to predict the response (e.g., a performance metric) of a high-fidelity model as a function of parameter values and the corresponding response of a surrogate model. However, the performance mapping system developed in the proposed framework is unique in that, unlike common surrogate modelling strategies, model parameters are *not* directly involved in mapping as predictors and also the response of the faster-to-run model is disaggregated into multiple factors (i.e., performance over independent events) to extract the most out of the available response. Importantly, since the mapping system does not involve the model parameters in the approximation procedure (as inputs to the mapping function), the limitations of common surrogate modeling strategies associated with high-dimensional problems (e.g., calibration problems with large numbers of model parameters – see Chapter 3 for surrogate modelling limitations) may be reduced or diminished.

Unlike the proposed framework, most strategies for surrogate modelling use Design of Experiments (DoE) at the beginning of the search to empirically capture the general form of the underlying function, e.g., a "correction function" representing the difference between the responses of lower-fidelity and original models (see Chapter 3 or *Razavi et al.*, [2012]). DoE is an essential part of many surrogate modelling strategies and very helpful to generate an unbiased estimate of the general form of the function, but at some relatively large computational costs (e.g., 10-20% of the total computational budget). The proposed framework does not fundamentally need such computationally demanding experiments since unlike the common correction function approaches used in lower-fidelity surrogate modelling which are purely empirical (see Chapter 3 or *Razavi et al.* [2012] for details), the developed performance mapping system makes use of the hydrological and statistical characteristics of the model responses to establish the relationships.

5.7 Conclusions

This study proposed a framework for efficient hydrologic model calibration on long data periods (longer than a few years). This framework becomes appealing when a hydrologic model of interest is computationally expensive and its run-time on such long periods is a few minutes or more. The basis of the framework is the fact that there are certain relationships between a hydrologic model performance on the full-length data and its corresponding performance on a representative short period of data (i.e., surrogate data). By extracting such relationships, a mapping system was developed to approximate the full-period model performance given the model performance on the surrogate data. The proposed framework enabled with this mapping system was designed such that in the course of a calibration experiment the majority of candidate parameter sets are evaluated by running the model on the surrogate data while the full-length data are only used to evaluate selected candidate parameter sets. Numerical experiments showed that this framework can increase calibration efficiency while being robust, as within a smaller computational budget, it can lead to calibration solutions close to the optimal solution of the full-period calibration problem. Results also demonstrated that the framework can work very well when the selected surrogate data period has only a moderate level of representativeness (i.e., not the best representative sub-period of a full-length data period) which is typically identifiable by experts' knowledge.

The proposed framework may be categorized under the large umbrella of surrogate modeling with the major distinction that unlike common surrogate modeling strategies which utilize a fast-to-run *surrogate model* of an original high-fidelity model, this framework defines *surrogate data* and only uses the original high-fidelity model. Future extensions to the proposed framework may incorporate the approximation uncertainty estimate for model performance predictions (i.e., the variance of errors associated with each prediction) that the developed mapping system is capable of producing [see *Sacks et al.*, 1989]. Moreover, the framework proposed can be extended and modified to be used in conjunction with uncertainty-based approaches to hydrologic model calibration such as GLUE or different Markov chain Monte Carlo techniques. Such extensions may also involve the approximation uncertainty of performance mapping.

Chapter 6
Conclusions and Future Directions

This thesis focused on developing new strategies and validating some existing strategies for circumventing the computational burden associated with the calibration of computationally intensive environmental models. These strategies are basically applicable to a range of simulation-optimization methodologies where the embedded simulation model is computationally intensive to run. In the following, conclusions attained in each chapter are outlined.

- The deterministic model preemption strategy developed in this thesis is a simple, efficient, and highly reliable approach for opportunistically evading (i.e., terminating) unnecessary model evaluations in the course of a calibration process. The numerical results showed that the computational savings afforded by enabling search algorithms with deterministic model preemption can be modest (~20%) to incredible (~97%) depending on the search algorithm and simulation model used, and these computational savings can be realized with absolutely no change to the search algorithm behavior or calibration results. Some calibration algorithms are ideal for deterministic model preemption in that they have easily definable preemption thresholds and also stand to benefit considerably in terms of increased computational efficiency, while some other algorithms are moderately suitable and some are unsuitable for deterministic model preemption. Moreover, deterministic model preemption is applicable only when the objective function used monotonically degrades as the simulation model generates outputs through time (simulates more time steps in calibration period) or space (based on Chapter 2).
- Surrogate modelling has been the most commonly used approach in the literature to circumvent the computational burden imposed by computationally intensive simulation models. However, there are a suite of fundamental considerations, limitations, and subjective decisions associated with surrogate modelling strategies (sometimes ignored in the literature) that must be recognized and taken into account by users before any surrogate modelling practice to ensure reliable analyses and conclusions. When designed and implemented appropriately, surrogate modelling strategies may enable the users to optimizer/calibrate environmental models of interest within a very limited number (e.g, as few as 100-200) of original computationally intensive model evaluations. Probably the most important limitation of surrogate modelling in applicability is when the number

of dimensions in the explanatory variable space is large (e.g., more than 15-20). Response surface surrogates are conceptually much more vulnerable to this limitation than lower-fidelity physically-based surrogates are. Moreover, any conclusion on the efficiency of a surrogate modelling strategy should be based on appropriate benchmarks. Such efficiency conclusions were empirically demonstrated to be computational-budget-dependent such that often surrogate modelling becomes less appealing for larger computational budgets (based on Chapter 3 and Chapter 4).

- Response surface surrogates which involve statistical or data-driven function approximation techniques are the basis of the most popular surrogate modelling strategies. Although these strategies are commonly believed to be efficient and reliable, this thesis demonstrates that response surface surrogates cannot always reliably emulate original response surfaces, and such strategies in some cases can be misleading and a hindrance, in particular for complex original response surfaces. This risk also depends on the type of the function approximation technique used and the framework through which the surrogate model, original model, and search algorithm are coordinated (based on Chapter 4).
- Lower-fidelity physically-based surrogates that are simplified models of original real-world systems of interest are conceptually more reliable in exploring the unseen regions in the explanatory variable space compared to response surface surrogates. This reliability directly relates to the level of fidelity of a surrogate model and diminishes for the surrogates with very low fidelity. As there is typically a trade-off between the level of fidelity of a model and its computational demand, the lower-fidelity model developers should create a level of fidelity that is sufficiently faithful to the original model while being efficient enough to permit the case study specific analysis required. Model calibration utilizing lower-fidelity surrogate models becomes very complicated when the surrogate model is defined on a different variable space (based on Chapter 3).
- The efficient automatic calibration framework developed in this thesis, which defines and utilizes surrogate calibration data as a substitute for full calibration data, is an easy-to-implement and reliable alternative to lower-fidelity physically-based surrogate modelling. Original computationally intensive simulation models run faster on surrogate data. This framework preserves the benefits of lower-fidelity surrogate modelling without requiring users to develop a secondary model (i.e., a surrogate model) of the real-world system of interest. The efficiency of

the framework depends on the level of representativeness of surrogate data. Moreover, the framework is designed in a way that can handle surrogate data that are not an ideal representative subset of the full calibration data. The establishment of the mapping system that reliably relates the model performance on surrogate data to the model performance on full calibration data is a key to this framework (based on Chapter 5).

Computational burdens associated with experiments requiring to repeatedly run computationally intensive environmental models will likely remain a challenge in the future. This challenge may be even growing as environmental and earth system models are becoming more advanced and sophisticated with the increase in availability of new sources of environmental data. This thesis has provided the groundwork for more systematic development of efficient strategies to circumvent such computational burdens. The following remarks summarize some suggestions on future research directions:

- The strategies developed in this thesis such as model preemption have the potential capability to be implemented on parallel computing resources which are becoming more readily available. Future research may be directed at developing efficient ways for combining the strategies developed here with parallel computing. Such combinations would be expected to further increase the computational efficiency of environmental model calibration.
- Although response surface surrogate modelling has been widely explored and applied in the literature, further research efforts are still required to address their shortcomings and improve their reliability. Improved methods will likely require incorporating the approximation uncertainty associated with response surface surrogates into the analysis enabled with surrogate modelling. Future research may also be directed at developing systematic, more practical strategies to create lower-fidelity physically-based surrogate models for different types of environmental models. Lower-fidelity physically-based surrogate modelling is still underutilized in water resources applications and seems to be a very fruitful area of future research.
- Surrogate modelling has been largely underutilized in uncertainty-based calibration of environmental models. As methods for uncertainty-based calibration are inherently very computationally demanding requiring extremely large numbers of model evaluations (e.g.,

sometimes millions of model runs), surrogate modelling would be expected to deliver huge benefits to such calibration methods. Future research may also be aimed at incorporating the uncertainty introduced by surrogate models into the uncertainty-based calibration process.

- The concept of “surrogate data” introduced in this thesis can find applications in many environmental modelling studies. Large amounts of detailed environmental data are typically available and incorporated into environmental models today and that volume of data only increases with time. As such, developing methods to extract surrogate data (i.e., representative subsets of full data) and methods to utilize original models with the surrogate data in calibration problems seems to be a promising and much needed research direction.

Glossary

ACO	Ant Colony Optimization
ANN	Artificial Neural Networks
BTC	Breakthrough Curve
CF	Correction Function
CMA-ES	Covariance Matrix Adaptation-Evolution Strategy
DACE	Design and Analysis of Computer Experiments
DDS	Dynamically Dimensioned Search
DDS-AU	Dynamically Dimensioned Search-Approximation of Uncertainty
DFRTT	Dipole Flow and Reactive Tracer Test
DOE	Design of Experiment
EGO	Efficient Global Optimization
EI	Expected Improvement
GA	Genetic Algorithm
GEM	Gaussian Emulation Machine
GLUE	Generalized Likelihood Uncertainty Estimation
GP	Genetic Programming
kNN	k-Nearest Neighbors
LHS	Latin Hypercube Sampling
MARS	Multivariate Adaptive Regression Splines
MCMC	Markov Chain Monte Carlo
MLMSRBF	Multistart Local Metric Stochastic Radial Basis Function
MOR	Model Order Reduction
MSE	Mean Squared Errors
N-S	Nash-Sutcliffe
PI	Probability of Improvement
PSO	Particle Swarm Optimization
RBF	Radial Basis Function
ReNN	Reformulated Neural Network
RMSE	Root Mean Squared Errors
rnd	Random
SCE	Shuffled Complex Evolution
SSE	Sum of Squared Errors
SVM	Support Vector Machines
SWAT	Soil and Water Assessment Tool

References

Abbaspour, K. C., C. A. Johnson, and M. T. van Genuchten (2004), Estimating uncertain flow and transport parameters using a sequential uncertainty fitting procedure, *Vadose Zone Journal*, *3*(4), 1340-1352.

Alam, F. M., K. R. McNaught, and T. J. Ringrose (2004), A comparison of experimental designs in the development of a neural network simulation metamodel, *Simulation Modelling Practice and Theory*, *12*(7-8), 559-578.

Alexandrov, N. M., and R. M. Lewis (2001), An overview of first-order approximation and model management in optimization, *Optimization and Engineering*, *2*, 413–430.

Allaire, D. L. (2009), Uncertainty assessment of complex models with application to aviation environmental systems, MIT.

Aly, A. H., and P. C. Peralta (1999), Comparison of a genetic algorithm and mathematical programming to the design of groundwater cleanup systems, *water resources research*, *35*(8), 2415-2425.

Andreassian, V., C. Perrin, C. Michel, I. Usart-Sanchez, and J. Lavabre (2001), Impact of imperfect rainfall knowledge on the efficiency and the parameters of watershed models, *Journal of Hydrology*, *250*(1-4), 206-223.

Asadzadeh M., S. Razavi, B. A. Tolson, D. Fay, W. Werick and Y. Fan, Multi-objective, multi-scenario Lake Superior regulation (2013), to be submitted to *Environmental Modelling and Software*.

Audouze, C., F. De Vuyst, and P. B. Nair (2009), Reduced-order modeling of parameterized PDEs using time-space-parameter principal component analysis, *International Journal for Numerical Methods in Engineering*, *80*(8), 1025-1057.

Bakr, M. H., J. W. Bandler, and N. Georgieva (1999), An aggressive approach to parameter extraction, *IEEE Transactions on Microwave Theory and Techniques*, *47*(12), 2428-2439.

Bandler, J. W., and K. Madsen (2001), Editorial-Surrogate Modelling and Space Mapping for Engineering Optimization, *Optimization and Engineering*, *2*(4), 367-368.

Bandler, J. W., R. M. Biernacki, and S. H. Chen (1996), Fully automated space mapping optimization of 3D structures, in *1996 IEEE Mtt-S International Microwave Symposium Digest, Vols 1-3*, edited by R. G. P. R. D. Ranson, pp. 753-756.

Bandler, J. W., S. Koziel, and K. Madsen (2008), Editorial - Surrogate modeling and space mapping for engineering optimization, *Optimization and Engineering*, *9*(4), 307-310.

Bandler, J. W., R. M. Biernacki, S. H. Chen, P. A. Grobelny, and R. H. Hemmers (1994), Space mapping technique for electromagnetic optimization, *IEEE Transactions on Microwave Theory and Techniques*, *42*(12), 2536-2544.

Bandler, J. W., Q. S. S. Cheng, S. A. Dakroury, A. S. Mohamed, M. H. Bakr, K. Madsen, and J. Sondergaard (2004), Space mapping: The state of the art, *IEEE Transactions on Microwave Theory and Techniques*, *52*(1), 337-361.

Bastos, L. S., and A. O'Hagan (2009), Diagnostics for Gaussian Process Emulators, *Technometrics*, *51*(4), 425-438.

Bau, D. A., and A. S. Mayer (2006), Stochastic management of pump-and-treat strategies using surrogate functions, *Advances in Water Resources*, *29*(12), 1901-1917.

Bautista, D. C. (2009), A Sequential Design For Approximating The Pareto Front Using the Expected Pareto Improvement Function, Ph.D. thesis, Ohio State University.

Beale, M. H., M. T. Hagan, and H. B. Demuth (2010), Neural Network Toolbox™ 7 - User's Guide, edited, MathWorks, Inc. (downloadable at: www.mathworks.com/help/pdf_doc/nnet/nnet.pdf).

Behzadian, K., Z. Kapelan, D. Savic, and A. Ardeshir (2009), Stochastic sampling design using a multi-objective genetic algorithm and adaptive neural networks, *Environmental Modelling & Software*, *24*(4), 530-541.

Beielstein, T., K. E. Parsopoulos, and M. N. Vrahatis (2002), Tuning PSO parameters through sensitivity analysis: Technical Report*Rep.*, Reihe Computational Intelligence, Collaborative Research Center, Department of Computer Science, University of Dortmund, CI 124n/02 (available at http://ls11-www.cs.uni-dortmund.de/people/tom/).

Beven, K. (2006), A manifesto for the equifinality thesis, *Journal of Hydrology*, *320*, 18-36.

Beven, K., and A. Binley (1992), The future of distributed models - model calibration and uncertainty prediction, *Hydrological Processes*, *6*(3), 279-298.

Beven, K., and J. Freer (2001), Equifinality, data assimilation, and uncertainty estimation in mechanistic modelling of complex environmental systems using the GLUE methodology, *Journal of Hydrology*, *249*(1-4), 11-29.

Bieker, H. P., O. Slupphaug, and T. A. Johansen (2007), Real-time production optimization of oil and gas production systems: A technology survey, *Spe Production & Operations*, *22*(4), 382-391.

Blanning, R. W. (1975), Construction and implementation of metamodels, *Simulation*, *24*(6), 177-184.

Blatman, G., and B. Sudret (2010), Efficient computation of global sensitivity indices using sparse polynomial chaos expansions, *Reliability Engineering & System Safety*, *95*(11), 1216-1229.

Blazkova, S., and K. Beven (2009), A limits of acceptability approach to model evaluation and uncertainty estimation in flood frequency estimation by continuous simulation: Skalka catchment, Czech Republic, *Water Resour. Res.*, *45*.

Bliznyuk, N., D. Ruppert, C. Shoemaker, R. Regis, S. Wild, and P. Mugunthan (2008), Bayesian calibration and uncertainty analysis for computationally expensive models using optimization and radial basis function approximation, *Journal of Computational and Graphical Statistics*, *17*(2), 270-294.

Borgonovo, E., W. Castaings, and S. Tarantola (2012), Model emulation and moment-independent sensitivity analysis: An application to environmental modelling, *Environmental Modelling & Software*, *34*, 105-115.

Box, G. E. P., and K. B. Wilson (1951), On the experimental attainment of optimum conditions, *Journal of the Royal Statistical Society Series B-Statistical Methodology*, *13*(1), 1-45.

Brazier, R. E., K. J. Beven, J. Freer, and J. S. Rowan (2000), Equifinality and uncertainty in physically based soil erosion models: Application of the glue methodology to WEPP-the water erosion prediction project-for sites in the UK and USA, *Earth Surface Processes And Landforms*, *25*(8), 825-845.

Broad, D. R., G. C. Dandy, and H. R. Maier (2005), Water distribution system optimization using metamodels, *Journal of Water Resources Planning and Management-Asce*, *131*(3), 172-180.

Broad, D. R., H. R. Maier, and G. C. Dandy (2010), Optimal Operation of Complex Water Distribution Systems Using Metamodels, *Journal of Water Resources Planning and Management-Asce*, *136*(4), 433-443.

Castelletti, A., F. Pianosi, R. Soncini-Sessa, and J. P. Antenucci (2010), A multiobjective response surface approach for improved water quality planning in lakes and reservoirs, *water resources research*, *46*.

Chen, V. C. P., K. L. Tsui, R. R. Barton, and M. Meckesheimer (2006), A review on design, modeling and applications of computer experiments, *Iie Transactions*, *38*(4), 273-291.

Cheng, C. T., X. Y. Wu, and K. W. Chau (2005), Multiple criteria rainfall-runoff model calibration using a parallel genetic algorithm in a cluster of computers, *Hydrological Sciences Journal-Journal Des Sciences Hydrologiques*, *50*(6), 1069-1087.

Cheng, W.-C., M. Putti, D. R. Kendall, and W. W. G. Yeh (2011), A real-time groundwater management model using data assimilation, *water resources research*, *47*.

Cherkassky, V., and Y. Q. Ma (2004), Practical selection of SVM parameters and noise estimation for SVM regression, *Neural Networks*, *17*(1), 113-126.

Cieniawski, S. E., J. W. Eheart, and S. Ranjithan (1995), Using genetic algorithms to solve a multiobjective groundwater monitoring problem, *water resources research*, *31*(2), 399-409.

Clerc, M. (2006), *Particle Swarm Optimization*, ISTE, London, UK.

Conti, S., and A. O'Hagan (2010), Bayesian emulation of complex multi-output and dynamic computer models, *Journal of Statistical Planning and Inference*, *140*(3), 640-651.

Cressie, N. A. C. (1993), *Statistics for spatial data. New York: John Wiley & Sons.*

Crout, N. M. J., D. Tarsitano, and A. T. Wood (2009), Is my model too complex? Evaluating model formulation using model reduction, *Environmental Modelling & Software*, *24*(1), 1-7.

Cui, T., Fox, C. , and M. J. O'Sullivan (2011), Bayesian calibration of a large scale geothermal reservoir model by a new adaptive delayed acceptance Metropolis Hastings algorithm, *Water Resour. Res., doi:10.1029/2010WR010352, in press.*

de Villiers, J., and E. Barnard (1993), Backpropagation neural nets with one and two hidden layers, *IEEE Transactions on Neural Networks*, *4*(1), 136-141.

di Pierro, F., S. T. Khu, D. Savic, and L. Berardi (2009), Efficient multi-objective optimal design of water distribution networks on a budget of simulations using hybrid algorithms, *Environmental Modelling & Software*, *24*(2), 202-213.

Doherty, J. (2005), *PEST: model independent parameter estimation*, fifth ed., Watermark Numerical Computing, Brisbane, Australia.

Duan, Q. Y., S. Sorooshian, and V. Gupta (1992), Effective and efficient global optimization for conceptual rainfall-runoff models, *water resources research*, *28*(4), 1015-1031.

Duan, Q. Y., V. K. Gupta, and S. Sorooshian (1993), Shuffled complex evolution approach for effective and efficient global minimization, *Journal of Optimization Theory and Applications*, *76*(3), 501-521.

Efendiev, Y., A. Datta-Gupta, V. Ginting, X. Ma, and B. Mallick (2005), An efficient two-stage Markov chain Monte Carlo method for dynamic data integration, *water resources research*, *41*(12).

Efstratiadis, A., and D. Koutsoyiannis (2010), One decade of multi-objective calibration approaches in hydrological modelling: a review, *Hydrological Sciences Journal-Journal Des Sciences Hydrologiques*, *55*(1), 58-78.

Eldred, M. S., A. A. Giunta, and S. S. Collis (2004), Second-order corrections for surrogate-based optimization with model hierarchies, paper presented at In: 10th AIAA/ISSMO multidisciplinary analysis and optimization conference, Albany, New York, 30–31.

Fen, C. S., C. C. Chan, and H. C. Cheng (2009), Assessing a response surface-based optimization approach for soil vapor extraction system design, *Journal of Water Resources Planning and Management-Asce*, *135*(3), 198-207.

Feyen, L., J. A. Vrugt, B. O. Nuallain, J. van der Knijff, and A. De Roo (2007), Parameter optimisation and uncertainty assessment for large-scale streamflow simulation with the LISFLOOD model, *Journal of Hydrology*, *332*(3-4), 276-289.

Foresee, F. D., and M. T. Hagan (1997), Gauss-Newton approximation to Bayesian Regularization, In: Proceedings of the 1997 International Joint Conference on Neural Networks, Houston, Texas, IEEE, New York, pp. 1930–1935.

Forrester, A. I. J., and A. J. Keane (2009), Recent advances in surrogate-based optimization, *Progress in Aerospace Sciences*, *45*(1-3), 50-79.

Forrester, A. I. J., N. W. Bressloff, and A. J. Keane (2006), Optimization using surrogate models and partially converged computational fluid dynamics simulations, *Proceedings of the Royal Society a-Mathematical Physical and Engineering Sciences*, *462*(2071), 2177-2204.

Forrester, A. I. J., A. Sobester, and A. J. Keane (2007), Multi-fidelity optimization via surrogate modelling, *Proceedings of the Royal Society a-Mathematical Physical and Engineering Sciences*, *463*(2088), 3251-3269.

Freer, J. E., H. McMillan, J. J. McDonnell, and K. J. Beven (2004), Constraining dynamic TOPMODEL responses for imprecise water table information using fuzzy rule based performance measures, *Journal of Hydrology*, *291*(3-4), 254-277.

Fricker, T., J. Oakley, and N. M. Urban (2010), Multivariate Emulators with Nonseparable Covariance Structures, Tech. Rep. 10/06, Joint Research Project on Managing Uncertainty in Complex Models (MUCM).

Friedman, J. H. (1991), Multivariate adaptive regression splines, *Annals of Statistics*, *19*(1), 1-67.

Gano, S. E., J. E. Renaud, J. D. Martin, and T. W. Simpson (2006), Update strategies for kriging models used in variable fidelity optimization, *Structural and Multidisciplinary Optimization*, *32*(4), 287-298.

Gibbs, M. S., H. R. Maier, and G. C. Dandy (2011), Relationship between problem characteristics and the optimal number of genetic algorithm generations, *Engineering Optimization*, *43*(4), 349-376.

Ginsbourger, D., R. Riche, and L. Carraro (2010), Kriging Is Well-Suited to Parallelize Optimization

Computational Intelligence in Expensive Optimization Problems, edited by Y. Tenne and C.-K. Goh, pp. 131-162, Springer Berlin Heidelberg.

Goldberg, D. E. (1989), *Genetic Algorithms in Search, Optimization and Machine Learning*, 372 pp., Addison-Wesley Longman Publishing Co., Boston, MA.

Gorissen, D. (2007), Heterogeneous Evolution of Surrogate Models, Master's Thesis, Katholieke Universiteit Leuven.

Gramacy, R. B., and H. K. H. Lee (2008), Bayesian Treed Gaussian Process Models With an Application to Computer Modeling, *Journal of the American Statistical Association*, *103*(483), 1119-1130.

Gray, G. A., and T. G. Kolda (2006), Algorithm 856: APPSPACK 4.0: Asynchronous parallel pattern search for derivative-free optimization, *Acm Transactions on Mathematical Software*, *32*(3), 485-507.

Griffin, J. D., T. G. Kolda, and R. M. Lewis (2008), Asynchronous parallel generating set search for linearly constrained optimization, *Siam Journal on Scientific Computing*, *30*(4), 1892-1924.

Gu, C. (2002), *Smoothing Spline ANOVA Models*, Springer, New York, USA.

Gugercin, S., A. C. Antoulas, and S. IEEE Control Systems (2000), A comparative study of 7 algorithms for model reduction, in *Proceedings of the 39th IEEE Conference on Decision and Control, Vols 1-5*, edited, pp. 2367-2372.

Gupta, H. V., S. Sorooshian, and P. O. Yapo (1999), Status of automatic calibration for hydrological models: comparison with multilevel expert calibration, *Journal of Hydrologic Engineering*, *4*(2), 135-143.

Gupta, H. V., H. Kling, K. K. Yilmaz, and G. F. Martinez (2009), Decomposition of the mean squared error and NSE performance criteria: Implications for improving hydrological modelling, *Journal of Hydrology*, *377*(1-2), 80-91.

Gupta, V. K., and S. Sorooshian (1985), The relationship between data and the precision of parameter estimates of hydrologic-models, *Journal of Hydrology*, *81*(1-2), 57-77.

Gutmann, H. M. (2001), A radial basis function method for global optimization, *Journal of Global Optimization*, *19*(3), 201-227.

Ha, K., D. C. Koh, B. W. Yum, and K. K. Lee (2007), Estimation of layered aquifer diffusivity and river resistance using flood wave response model, *Journal of Hydrology*, *337*(3-4), 284-293.

Hagan, M. T., and M. B. Menhaj (1994), Training feedforward networks with Marquardt algorithm, *IEEE Transactions on Neural Networks*, *5*(6), 989-993.

Hamm, L., B. W. Brorsen, and M. T. Hagan (2007), Comparison of stochastic global optimization methods to estimate neural network weights, *Neural Processing Letters*, *26*, 145-158.

Hansen, N., and A. Ostermeier (2001), Completely derandomized self-adaptation in evolution strategies, *Evolutionary Computation*, *9*(2), 159-195.

Hansen, N., S. D. Muller, and P. Koumoutsakos (2003), Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES), *Evolutionary Computation*, *11*(1), 1-18.

Haupt, R. L., and S. E. Haupt (1998), *Practical Genetic Algorithms*, Wiley, New York.

He, Y., and C. W. Hui (2007), Genetic algorithm based on heuristic rules for high-constrained large-size single-stage multi-product scheduling with parallel units, *Chemical Engineering and Processing*, *46*(11), 1175-1191.

Hemker, T., K. R. Fowler, M. W. Farthing, and O. von Stryk (2008), A mixed-integer simulation-based optimization approach with surrogate functions in water resources management, *Optimization and Engineering*, *9*(4), 341-360.

Hornberger, G. M., and R. C. Spear (1981), An approach to the preliminary-analysis of environmental systems, *Journal of Environmental Management*, *12*(1), 7-18.

Hornik, K., M. Stinchcombe, and H. White (1989), Multilayer feedforward networks are universal approximators, *Neural Networks*, *2*(5), 359-366.

Hsu, K. L., H. V. Gupta, and S. Sorooshian (1995), Artificial neural network modeling of rainfall-runoff process, *water resources research*, *31*(10), 2517-2530.

Huang, D., T. T. Allen, W. I. Notz, and R. A. Miller (2006), Sequential kriging optimization using multiple-fidelity evaluations, *Structural and Multidisciplinary Optimization*, *32*(5), 369-382.

Hussain, M. F., R. R. Barton, and S. B. Joshi (2002), Metamodeling: Radial basis functions, versus polynomials, *European Journal of Operational Research*, *138*(1), 142-154.

Jeong, S., S. Obayashi, and Ieee (2005), *Efficient global optimization (EGO) for multi-objective problem and data mining*, 2138-2145 pp.

Jin, R., W. Chen, and T. W. Simpson (2001), Comparative studies of metamodelling techniques under multiple modelling criteria, *Structural and Multidisciplinary Optimization*, *23*(1), 1-13.

Jin, Y. (2005), A comprehensive survey of fitness approximation in evolutionary computation, *Soft Computing*, *9*(1), 3-12.

Jin, Y. C., M. Olhofer, and B. Sendhoff (2002), A framework for evolutionary optimization with approximate fitness functions, *IEEE Transactions on Evolutionary Computation*, *6*(5), 481-494.

Johnson, V. M., and L. L. Rogers (2000), Accuracy of neural network approximators in simulation-optimization, *Journal of Water Resources Planning and Management-Asce*, *126*(2), 48-56.

Jones, D. R. (2001), A taxonomy of global optimization methods based on response surfaces, *Journal of Global Optimization*, *21*(4), 345-383.

Jones, D. R., M. Schonlau, and W. J. Welch (1998), Efficient global optimization of expensive black-box functions, *Journal of Global Optimization*, *13*(4), 455-492.

Joslin, D., J. Dragovich, H. Vo, and J. Terada (2006), Opportunistic fitness evaluation in a genetic algorithm for civil engineering design optimization, paper presented at IEEE Congress on Evolutionary Computation.

Jourdan, L., D. W. Corne, D. A. Savic, and G. Walters (2006), LEMMO: hybridising rule induction and NSGA II for multi-objective water systems design. In: Proc. of the Eighth International Conference on Computing and Control for the Water Industry, vol. 2, pp. 45-50.

Juston, J., J. Seibert, and P.-O. Johansson (2009), Temporal sampling strategies and uncertainty in calibrating a conceptual hydrological model for a small boreal catchment, *Hydrological Processes*, *23*(21), 3093-3109.

Karamouz, M., S. Razavi, and S. Araghinejad (2008), Long-lead seasonal rainfall forecasting using time-delay recurrent neural networks: a case study, *Hydrological Processes*, *22*(2), 229-241.

Kavetski, D., and G. Kuczera (2007), Model smoothing strategies to remove microscale discontinuities and spurious secondary optima in objective functions in hydrological calibration, *Water Resources Research*, *43*(3), 9.

Kavetski, D., G. Kuczera, and S. W. Franks (2006), Bayesian analysis of input uncertainty in hydrological modeling: 1. Theory, *water resources research*, *42*(3).

Keane, A. J. (2006), Statistical improvement criteria for use in multiobjective design optimization, *Aiaa Journal*, *44*(4), 879-891.

Keating, E. H., J. Doherty, J. A. Vrugt, and Q. J. Kang (2010), Optimization and uncertainty assessment of strongly nonlinear groundwater models with high parameter dimensionality, *water resources research*, *46*.

Kennedy, J., and R. C. Eberhart (1995), Particle Swarm Optimization, paper presented at IEEE International Conference on Neural Networks, IEEE Service Center, Piscataway, NJ.

Kennedy, J., and R. C. Eberhart (2001), *Swarm Intelligence*, Academic Press, London.

Kennedy, M. C., and A. O'Hagan (2000), Predicting the output from a complex computer code when fast approximations are available, *Biometrika*, *87*(1), 1-13.

Kennedy, M. C., and A. O'Hagan (2001), Bayesian calibration of computer models, *Journal of the Royal Statistical Society Series B-Statistical Methodology*, *63*, 425-450.

Khan, M. S., and P. Coulibaly (2006), Bayesian neural network for rainfall-runoff modeling, *water resources research*, *42*(7).

Khu, S. T., and M. G. F. Werner (2003), Reduction of Monte-Carlo simulation runs for uncertainty estimation in hydrological modelling, *Hydrology and Earth System Sciences*, *7*(5), 680-692.

Khu, S. T., D. Savic, Y. Liu, and H. Madsen (2004), A fast evolutionary-based metamodelling approach for the calibration of a rainfall-runoff model, paper presented at The First Biennial Meeting of the International Environmental Modelling and Software Society, Osnabruck, Germany.

Kim, C., S. Wang, and K. K. Choi (2005), Efficient response surface modeling by using moving least-squares method and sensitivity, *Aiaa Journal*, *43*(11), 2404-2411.

Kim, D. W., and J. Lee (2010), An improvement of Kriging based sequential approximate optimization method via extended use of design of experiments, *Engineering Optimization*, *42*(12), 1133-1149.

Kingston, G. B., M. F. Lambert, and H. R. Maier (2005), Bayesian training of artificial neural networks used for water resources modeling, *water resources research*, *41*(12).

Kingston, G. B., H. R. Maier, and M. F. Lambert (2008), Bayesian model selection applied to artificial neural networks used for water resources modeling, *water resources research*, *44*(4).

Kleijnen, J. P. C. (2009), Kriging metamodeling in simulation: A review, *European Journal of Operational Research*, *192*(3), 707-716.

Knowles, J. (2006), ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems, *IEEE Transactions on Evolutionary Computation*, *10*(1), 50-66.

Koch, P. N., T. W. Simpson, J. K. Allen, and F. Mistree (1999), Statistical approximations for multidisciplinary design optimization: The problem of size, *Journal of Aircraft*, *36*(1), 275-286.

Kollat, J. B., P. M. Reed, and T. Wagener (2012), When are multiobjective calibration trade-offs in hydrologic models meaningful?, *Water Resources Research*, *48*.

Kourakos, G., and A. Mantoglou (2009), Pumping optimization of coastal aquifers based on evolutionary algorithms and surrogate modular neural network models, *Advances in Water Resources*, *32*(4), 507-521.

Kouwen, N., E. D. Soulis, A. Pietroniro, J. Donald, and R. A. Harrington (1993), Grouped Response Units for Distributed Hydrologic Modeling, *Journal of Water Resources Planning and Management-Asce*, *119*(3), 289-305.

Kuczera, G., and E. Parent (1998), Monte Carlo assessment of parameter uncertainty in conceptual catchment models: the Metropolis algorithm, *Journal of Hydrology*, *211*(1-4), 69-85.

Kuzmin, V., D. J. Seo, and V. Koren (2008), Fast and efficient optimization of hydrologic model parameters using a priori estimates and stepwise line search, *Journal of Hydrology*, *353*(1-2), 109-128.

Leary, S. J., A. Bhaskar, and A. J. Keane (2003), A knowledge-based approach to response surface modelling in multifidelity optimization, *Journal of Global Optimization*, *26*(3), 297-319.

Lee, J., H. Jeong, D.-H. Choi, V. Volovoi, and D. Mavris (2007), An enhancement of constraint feasibility in BPN based approximate optimization, *Computer Methods in Applied Mechanics and Engineering*, *196*(17-20), 2147-2160.

Leshno, M., V. Y. Lin, A. Pinkus, and S. Schocken (1993), Multilayer feedforward networks with a non-polynomial activation function can approximate any function, *Neural Networks*, *6*(6), 861-867.

Li, M., G. Li, and S. Azarm (2008), A kriging metamodel assisted multi-objective genetic algorithm for design optimization, *Journal of Mechanical Design*, *130*(3).

Liong, S. Y., S. T. Khu, and W. T. Chan (2001), Derivation of Pareto front with genetic algorithm and neural network, *Journal of Hydrologic Engineering*, *6*(1), 52-61.

Liu, W. (2003), Development of gradient-enhanced kriging approximations for multidisciplinary design optimization, 177 pp, University of Notre Dame, Notre Dame.

Locatelli, M. (2003), A note on the Griewank test function, *Journal of Global Optimization*, *25*(2), 169-174.

Lophaven, S. N., H. B. Nielsen, and J. Sondergaard (2002), DACE: A MATLAB Kriging toolbox version 2.0,Technical Report IMM-TR-2002-12, Technical University of Denmark, Copenhagen.*Rep.*

Luk, K. C., J. E. Ball, and A. Sharma (2000), A study of optimal model lag and spatial inputs to artificial neural network for rainfall forecasting, *Journal of Hydrology*, *227*(1-4), 56-65.

Mackay, D. J. C. (1992), Bayesin interpolation, *Neural Computation*, *4*(3), 415-447.

MacLean, A. (2009), Calibration and analysis of the MESH hydrological model applied to cold regions, MASc Thesis, Civil and Environmental Engineering, University of Waterloo, Waterloo, ON.

Madsen, H., G. Wilson, and H. C. Ammentrop (2002), Comparison of different automated strategies for calibration of rainfall-runoff models, *Journal of Hydrology*, *261*(1-4), 48-59.

Madsen, J. I., and M. Langthjem (2001), Multifidelity response surface approximations for the optimum design of diffuser flows, *Optimization and Engineering, 2,* 453-468.

Maier, H. R., and G. C. Dandy (2000), Neural networks for the prediction and forecasting of water resources variables: a review of modelling issues and applications, *Environmental Modelling & Software*, *15*(1), 101-124.

Maier, H. R., A. Jain, G. C. Dandy, and K. P. Sudheer (2010), Methods used for the development of neural networks for the prediction of water resource variables in river systems: Current status and future directions, *Environmental Modelling & Software*, *25*(8), 891-909.

Maier, H. R., A. R. Simpson, A. C. Zecchin, W. K. Foong, K. Y. Phang, H. Y. Seah, and C. L. Tan (2003), Ant colony optimization distribution for design of water systems, *Journal of Water Resources Planning and Management-Asce*, *129*(3), 200-209.

Maschler, T., and D. A. Savic (1999), Simplification of Water Supply Network Models through Linearisation, Centre for Water Systems, Report No.99/01, School of Engineering, University of Exeter, Exeter, United Kingdom, p.119.*Rep.*

MathWorks (2010), MATLAB global optimization toolbox 3 edited.

Matott, L. S. (2005), OSTRICH: an optimization software tool: documentation and user's guide*Rep.*, University at Buffalo, Buffalo, NY.

Matott, L. S., and A. J. Rabideau (2008), Calibration of complex subsurface reaction models using a surrogate-model approach, *Advances in Water Resources*, *31*(12), 1697-1707.

McKay, M. D., R. J. Beckman, and W. J. Conover (1979), A comparison of three methods for selecting values of input variables in the analysis of output from a computer code, *Technometrics*, *21*(2), 239-245.

McKinney, D. C., and M. D. Lin (1994), Genetic algorithm solution of groundwater-management models, *water resources research*, *30*(6), 1897-1906.

McPhee, J., and W. W. G. Yeh (2008), Groundwater management using model reduction via empirical orthogonal functions, *Journal of Water Resources Planning and Management-Asce*, *134*(2), 161-170.

Melching, C. S., B. C. Yen, and H. G. Wenzel (1990), A reliability estimation in modeling watershed runoff with uncertainties, *water resources research*, *26*(10), 2275-2286.

Michalski, R. S. (2000), Learnable Evolution ModeL: Evolutionary processes guided by machine learning, *Machine Learning*, *38*(1-2), 9-40.

Mondal, A., Y. Efendiev, B. Mallick, and A. Datta-Gupta (2010), Bayesian uncertainty quantification for flows in heterogeneous porous media using reversible jump Markov chain Monte Carlo methods, *Advances in Water Resources*, *33*(3), 241-256.

Montgomery, D. C. (2008), *Design and analysis of experiments*, 7th ed., Wiley.

Mugunthan, P., and C. A. Shoemaker (2006), Assessing the impacts of parameter uncertainty for computationally expensive groundwater models, *Water Resources Research*, *42*(10).

Mugunthan, P., C. A. Shoemaker, and R. G. Regis (2005), Comparison of function approximation, heuristic, and derivative-based methods for automatic calibration of computationally expensive groundwater bioremediation models, *Water Resources Research*, *41*(11), 1-17.

Mullur, A. A., and A. Messac (2006), Metamodeling using extended radial basis functions: a comparative approach, *Engineering with Computers*, *21*(3), 203-217.

Myers, R. H., and D. C. Montgomery (2002), *Response surface methodology*, 2nd ed., Wiley-Interscience, New York.

Nakayama, H., M. Arakawa, and R. Sasaki (2002), Simulation-Based Optimization Using Computational Intelligence, *Optimization and Engineering*, *3*(2), 201–214.

Nash, J. E., and J. V. Sutcliffe (1970a), River flow forecasting through conceptual models, part I. A discussion of principles, *J. Hydrology*, *10*, 282– 290.

Nash, J. E., and J. V. Sutcliffe (1970b), River flow forecasting through conceptual models part I — A discussion of principles, *Journal of Hydrology*, *10*(3), 282-290.

Ndiritu, J. G., and T. M. Daniell (2001), An improved genetic algorithm for rainfall-runoff model calibration and function optimization, *Mathematical and Computer Modelling*, *33*(6-7), 695-706.

Neitsch, S. L., J. G. Arnold, J. R. Kiniry, and J. R. Williams (2001), Soil and Water Assessment Tool theoretical documentation version 2000: Draft–April 2001, 506 pp., Agric. Res. Serv., U.S. Dep. of Agric., Temple, Tex.*Rep.*

Neitsch, S. L., J. G. Arnold, J. R. Kiniry, and J. R. Williams (2001), Soil and Water Assessment Tool Theoretical Documentation Version 2000: Draft-April 2001*Rep.*, 506 pp, US Department of Agriculture - Agricultural Research Service, Temple, Texas.

Nelder, J. A., and R. Mead (1965), A simplex method for function minimization, *Computer Journal*, *7*(4), 308-313.

Nguyen, D., and B. Widrow (1990), Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights, *Ijcnn International Joint Conference on Neural Networks*, *Vols 1-3*, C21-C26.

Nicklow, J., et al. (2010), State of the Art for Genetic Algorithms and Beyond in Water Resources Planning and Management, *Journal of Water Resources Planning and Management-Asce*, *136*(4), 412-432.

O'Hagan, A. (2006), Bayesian analysis of computer code outputs: A tutorial, *Reliability Engineering & System Safety*, *91*(10-11), 1290-1300.

Ostfeld, A., and S. Salomons (2005), A hybrid genetic - instance based learning algorithm for CE-QUAL-W2 calibration, *Journal of Hydrology*, *310*(1-4), 122-142.

Papadrakakis, M., N. D. Lagaros, and Y. Tsompanakis (1998), Structural optimization using evolution strategies and neural networks, *Computer Methods in Applied Mechanics and Engineering*, *156*(1-4), 309-333.

Pechlivanidis, I. G., B. M. Jackson, N. R. McIntyre, and H. S. Wheater (2011), Catchment scale hydrological modelling: a review of model types, calibration approaches and uncertainty analysis methods in eth context of recent developments in technology and applications, *Global Nest Journal*, *13*(3), 193-214.

Pedersen, M. E. H. (2008), *SwarmOps: Black-Box Optimization in ANSI C*, Hvass Laboratories.

Perrin, C., L. Oudin, V. Andreassian, C. Rojas-Serna, C. Michel, and T. Mathevet (2007), Impact of limited streamflow data on the efficiency and the parameters of rainfall-runoff models, *Hydrological Sciences Journal-Journal Des Sciences Hydrologiques*, *52*(1), 131-151.

Pianosi, F., and R. Soncini-Sessa (2009), Real-time management of a multipurpose water reservoir with a heteroscedastic inflow model, *water resources research*, *45*.

Picheny, V., N.-H. Kim, R. T. Haftka, and N. V. Queipo (2008), Conservative predictions using surrogate modeling, in *49th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Schaumburg, IL, AIAA Paper 2008-1716, April 2008.*, edited.

Pierson, F. B., C. W. Slaughter, and Z. K. Cram (2000), Monitoring discharge and suspended sediment, Reynolds Creek Experimental Watershed, Idaho, USA, ARS Technical Bulletin, NWRC-2000-8*Rep.*

Pietroniro, A., et al. (2007), Development of the MESH modelling system for hydrological ensemble forecasting of the Laurentian Great Lakes at the regional scale, *Hydrology and Earth System Sciences*, *11*(4), 1279–1294.

Poeter, E. P., and M. C. Hill (1998), Documentation of UCODE: a computer code for universal inverse modeling, *Water-Resources Investigations Report Rep.*, 121 pp pp, US Geological Survey.

Ponweiser, W., T. Wagner, D. Biermann, and M. Vincze (2008), Multiobjective Optimization on a Limited Budget of Evaluations Using Model-Assisted S-Metric Selection, in *Parallel Problem Solving from Nature - Ppsn X, Proceedings*, edited by G. J. T. L. S. P. C. B. N. Rudolph, pp. 784-794.

Preis, A., A. J. Whittle, A. Ostfeld, and L. Perelman (2011), Efficient Hydraulic State Estimation Technique Using Reduced Models of Urban Water Networks, *Journal of Water Resources Planning and Management-Asce*, *137*(4), 343-351.

Rabitz, H., O. F. Alis, J. Shorter, and K. Shim (1999), Efficient input-output model representations, *Computer Physics Communications*, *117*(1-2), 11-20.

Ratto, M., and A. Pagano (2010), Using recursive algorithms for the efficient identification of smoothing spline ANOVA models, *Asta-Advances in Statistical Analysis*, *94*(4), 367-388.

Ratto, M., A. Pagano, and P. Young (2007), State dependent parameter metamodelling and sensitivity analysis, *Computer Physics Communications*, *177*(11), 863-876.

Ratto, M., A. Castelletti, and A. Pagano (2012), Editorial - Emulation techniques for the reduction and sensitivity analysis of complex environmental models, *Environmental Modelling & Software*, *34*(0), 1-4.

Razavi, S. and B. A. Tolson (2013), An efficient framework for hydrologic model calibration on a long data periods, *Water Resources Research* (submitted in Dec 2012).

Razavi, S., M. Asadzadeh, B. A. Tolson, D. Fay, S. Moin, J. Bruxer, Y. Fan (2013), Evaluation of new control structures for regulating the Great Lakes system: a multi-scenario, multi-reservoir optimization approach, *ASCE Journal of Water Resources Planning and Management (*submitted in April 2012).

Razavi, S., and M. Karamouz (2007), Adaptive neural networks for flood routing in river systems, *Water International*, *32*(3), 360-375.

Razavi, S., and S. Araghinejad (2009), Reservoir Inflow Modeling Using Temporal Neural Networks with Forgetting Factor Approach, *Water Resources Management*, *23*(1), 39-55.

Razavi, S., and B. A. Tolson (2011), A new formulation for feedforward neural networks, *IEEE Transactions on Neural Networks*, *22*(10), 1588-1598, DOI: 1510.1109/TNN.2011.2163169.

Razavi, S., B. A. Tolson, and D. H. Burn (2012a), Numerical Assessment of Metamodelling Strategies in Computationally Intensive Optimization, *Environmental Modelling and Software, in press.*

Razavi, S., B. A. Tolson, and D. H. Burn (2012b), Review of surrogate modeling in water resources, *Water Resources Research*, *48*.

Razavi, S., B. A. Tolson, L. S. Matott, N. R. Thomson, A. MacLean, and F. R. Seglenieks (2010), Reducing the Computational Cost of Automatic Calibration through Model Pre-Emption, *water resources research*.

Reed, P., B. S. Minsker, and D. E. Goldberg (2003), Simplifying multiobjective optimization: An automated design methodology for the nondominated sorted genetic algorithm-II, *Water Resour. Res.*, *39*(7).

Reed, R. (1993), Pruning algorithms - a survey, *IEEE Transactions on Neural Networks*, *4*(5), 740-747.

Regis, R. G., and C. A. Shoemaker (2004), Local function approximation in evolutionary algorithms for the optimization of costly functions, *IEEE Transactions on Evolutionary Computation*, *8*(5), 490-505.

Regis, R. G., and C. A. Shoemaker (2007a), Improved strategies for radial basis function methods for global optimization, *Journal of Global Optimization*, *37*(1), 113-135.

Regis, R. G., and C. A. Shoemaker (2007b), A stochastic radial basis function method for the global optimization of expensive functions, *Informs Journal on Computing*, *19*(4), 497-509.

Regis, R. G., and C. A. Shoemaker (2009), Parallel Stochastic Global Optimization Using Radial Basis Functions, *Informs Journal on Computing*, *21*(3), 411-426.

Reichard, E. G. (1995), Groundwater-surface water management with stochastic surface-water supplies - a simulation optimization approach, *water resources research*, *31*(11), 2845-2865.

Rewienski, M., and J. White (2006), Model order reduction for nonlinear dynamical systems based on trajectory piecewise-linear approximations, *Linear Algebra and Its Applications*, *415*(2-3), 426-454.

Robinson, T. D. (2007), Surrogate-based Optimization Using Multifidelity Models with Variable Parameterization, MIT, Cambridge.

Robinson, T. D., M. S. Eldred, K. E. Willcox, and R. Haimes (2006), Strategies for Multifidelity Optimization with Variable Dimensional Hierarchical Models, in *In: 47th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, AIAA Paper 2006-1819*, edited, Newport, Rhode Island, May 1-4

Robinson, T. D., M. S. Eldred, K. E. Willcox, and R. Haimes (2008), Surrogate-Based Optimization Using Multifidelity Models with Variable Parameterization and Corrected Space Mapping, *Aiaa Journal*, *46*(11), 2814-2822.

Roos, G. N. (2009), Development of the Dipole Flow and Reactive Tracer Test (DFRTT) for aquifer parameter estimation, MASc Thesis, Civil and Environmental Engineering, University of Waterloo, Waterloo, ON.

Ryu, J. S., M. S. Kim, K. J. Cha, T. H. Lee, and D. H. Choi (2002), Kriging interpolation methods in geostatistics and DACE model, *Ksme International Journal*, *16*(5), 619-632.

Sacks, J., W. J. Welch, T. J. Mitchell, and H. P. Wynn (1989), Design and analysis of computer experiments *Statistical Science*, *4*(4), 409-435.

Sakata, S., F. Ashida, and M. Zako (2003), Structural optimization using Kriging approximation, *Computer Methods in Applied Mechanics and Engineering*, *192*(7-8), 923-939.

Saltelli, A., S. Tarantola, and F. Campolongo (2000), Sensitivity analysis as an ingredient of modeling, *Statistical Science*, *15*(4), 377-395.

Sasena, M. J., P. Papalambros, and P. Goovaerts (2002), Exploration of metamodeling sampling criteria for constrained global optimization, *Engineering Optimization*, *34*(3), 263-278.

Savic, D. A., and G. A. Walters (1997), Genetic algorithms for least-cost design of water distribution networks, *Journal of Water Resources Planning and Management-Asce*, *123*(2), 67-77.

Schonlau, M. (1997a), Computer experiments and global optimization, PhD thesis, University of Waterloo.

Schonlau, M. (1997b), Computer Experiments and Global Optimization University of Waterloo, Waterloo.

Schultz, M. T., M. J. Small, R. S. Farrow, and P. S. Fischbeck (2004), State water pollution control policy insights from a reduced-form model, *Journal of Water Resources Planning and Management-Asce*, *130*(2), 150-159.

Schultz, M. T., M. J. Small, P. S. Fischbeck, and R. S. Farrow (2006), Evaluating response surface designs for uncertainty analysis and prescriptive applications of a large-scale water quality model, *Environmental Modeling & Assessment*, *11*(4), 345-359.

Schutte, J. F., J. A. Reinbolt, B. J. Fregly, R. T. Haftka, and A. D. George (2004), Parallel global optimization with the particle swarm algorithm, *International Journal for Numerical Methods in Engineering*, *61*(13), 2296-2315.

Sexton, R. S., R. E. Dorsey, and J. D. Johnson (1998), Toward global optimization of neural networks: A comparison of the genetic algorithm and backpropagation, *Decision Support Systems*, *22*(2), 171-185.

Shamir, U., and E. Salomons (2008), Optimal real-time operation of urban water distribution systems using reduced models, *Journal of Water Resources Planning and Management-Asce*, *134*(2), 181-185.

Shamseldin, A. Y. (1997), Application of a neural network technique to rainfall-runoff modelling, *Journal of Hydrology*, *199*(3-4), 272-294.

Shan, S., and G. G. Wang (2004), Space exploration and global optimization for computationally intensive design problems: a rough set based approach, *Structural and Multidisciplinary Optimization*, *28*(6), 427-441.

Shan, S., and G. G. Wang (2010a), Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions, *Structural and Multidisciplinary Optimization*, *41*(2), 219-241.

Shan, S., and G. G. Wang (2010b), Metamodeling for High Dimensional Simulation-Based Design Problems, *Journal of Mechanical Design*, *132*(5).

Shin, Y. S., and R. V. Grandhi (2001), A global structural optimization technique using an interval method, *Structural and Multidisciplinary Optimization*, *22*(5), 351-363.

Shoemaker, C. A., R. G. Regis, and R. C. Fleming (2007), Watershed calibration using multistart local optimization and evolutionary optimization with radial basis function approximation, *Hydrological Sciences Journal-Journal Des Sciences Hydrologiques*, *52*(3), 450-465.

Shrestha, D. L., N. Kayastha, and D. P. Solomatine (2009), A novel approach to parameter uncertainty analysis of hydrological models using neural networks, *Hydrology and Earth System Sciences*, *13*(7), 1235-1248.

Siade, A. J., M. Putti, and W. W. G. Yeh (2010), Snapshot selection for groundwater model reduction using proper orthogonal decomposition, *water resources research*, *46*.

Simpson, T. W., and F. Mistree (2001), Kriging models for global approximation in simulation-based multidisciplinary design optimization, *Aiaa Journal*, *39*(12), 2233-2241.

Simpson, T. W., J. D. Peplinski, P. N. Koch, and J. K. Allen (2001), Metamodels for computer-based engineering design: survey and recommendations, *Engineering with Computers*, *17*(2), 129-150.

Simpson, T. W., V. Toropov, B. V., and F. A. C. Viana (2008), Design and analysis of computer experiments in multidisciplinary design optimization: a review of how far we have come - or not paper presented at 12th AIAA/ISSMO multidisciplinary analysis and optimization conference, Victoria, British Columbia, Canada.

Simpson, T. W., A. J. Booker, D. Ghosh, A. A. Giunta, P. N. Koch, and R. J. Yang (2004), Approximation methods in multidisciplinary analysis and optimization: a panel discussion, *Structural and Multidisciplinary Optimization*, *27*(5), 302-313.

Singh, S. K., and A. Bardossy (2012), Calibration of hydrological models on hydrologically unusual events, *Advances in Water Resources*, *38*, 81-91.

Skaggs, T. H., and D. A. Barry (1997), The first-order reliability method of predicting cumulative mass flux in heterogeneous porous formations, *water resources research*, *33*(6), 1485-1494.

Sobester, A., S. J. Leary, and A. J. Keane (2005), On the design of optimization strategies based on global response surface approximation models, *Journal of Global Optimization*, *33*(1), 31-59.

Sobol, I. M. (2003), Theorems and examples on high dimensional model representation, *Reliability Engineering & System Safety*, *79*(2), 187-193.

Sreekanth, J., and B. Datta (2011), Coupled simulation-optimization model for coastal aquifer management using genetic programming-based ensemble surrogate models and multiple-realization optimization, *water resources research*, *47*.

Storlie, C. B., L. P. Swiler, J. C. Helton, and C. J. Sallaberry (2009), Implementation and evaluation of nonparametric regression procedures for sensitivity analysis of computationally demanding models, *Reliability Engineering & System Safety*, *94*(11), 1735-1763.

Storlie, C. B., H. D. Bondell, B. J. Reich, and H. H. Zhang (2011), Surface estimation, variable selection, and the nonparametric oracle property, *Statistica Sinica*, *21*(2), 679-705.

Sun, G., G. Li, M. Stone, and Q. Li (2010), A two-stage multi-fidelity optimization procedure for honeycomb-type cellular materials, *Computational Materials Science*, *49*(3), 500-511.

Svenson, J. D. (2011), Computer Experiments: Multiobjective Optimization and Sensitivity Analysis, Ph.D. thesis, Ohio State University.

Tamura, S., and M. Tateishi (1997), Capabilities of a four-layered feedforward neural network: Four layers versus three, *IEEE Transactions on Neural Networks*, *8*(2), 251-255.

Tan, C. C., C. P. Tung, C. H. Chen, and W. W. G. Yeh (2008), An integrated optimization algorithm for parameter structure identification in groundwater modeling, *Advances in Water Resources*, *31*(3), 545-560.

Teoh, E. J., K. C. Tan, and C. Xiang (2006), Estimating the number of hidden neurons in a feedforward network using the singular value decomposition, *IEEE Transactions on Neural Networks*, *17*(6), 1623-1629.

Thokala, P., and J. R. R. A. Martins (2007), Variable-complexity optimization applied to airfoil design, *Engineering Optimization*, *39*(3), 271-286.

Thomson, N. R., S. F. Thornton, R. D. Wilson, S. S. Banwart, and B. Rehia (2010), Dipole flow and reactive tracer test, *submitted to J. Contam. Hydrol.*

Tolson, B. A., and C. A. Shoemaker (2007a), Cannonsville Reservoir Watershed SWAT2000 model development, calibration and validation, *Journal of Hydrology*, *337*(1-2), 68-86.

Tolson, B. A., and C. A. Shoemaker (2007b), Dynamically dimensioned search algorithm for computationally efficient watershed model calibration, *Water Resources Research*, *43*(1).

Tolson, B. A., and C. A. Shoemaker (2008), Efficient prediction uncertainty approximation in the calibration of environmental simulation models, *Water Resources Research*, *44*(4).

Tolson, B. A., M. Asadzadeh, H. R. Maier, and A. Zecchin (2009), Hybrid discrete dynamically dimensioned search (HD-DDS) algorithm for water distribution system design optimization, *water resources research*, *45*.

Tonkin, M., and J. Doherty (2009), Calibration-constrained Monte Carlo analysis of highly parameterized models using subspace techniques, *Water Resources Research*, *45*.

Torczon, V. (1997), On the convergence of pattern search algorithms, *Siam Journal on Optimization*, *7*(1), 1-25.

Ulanicki, B., A. Zehnpfund, and F. Martinez (1996), Simplification of water distribution network models Proc., 2nd Int. Conf. on Hydroinformatics,

, in *Proc., 2nd Int. Conf. on Hydroinformatics*, edited, Balkema, Rotterdam, Netherlands, 493–500.

van Keulen, F., and K. Vervenne (2004), Gradient-enhanced response surface building, *Structural and Multidisciplinary Optimization*, *27*(5), 337-351.

Vermeulen, P. T. M., A. W. Heemink, and C. Stroet (2004), Reduced models for linear groundwater flow models using empirical orthogonal functions, *Advances in Water Resources*, *27*(1), 57-69.

Vermeulen, P. T. M., A. W. Heemink, and J. R. Valstar (2005), Inverse modeling of groundwater flow using model reduction, *water resources research*, *41*(6).

Vermeulen, P. T. M., C. B. M. t. Stroet, and A. W. Heemink (2006), Model inversion of transient nonlinear groundwater flow models using model reduction, *water resources research*, *42*(9).

Verseghy, D. L. (1991), CLASS – a Canadian land surface scheme for GCMs, I. Soil model, *International Journal of Climatology*, *11*(2), 111-133.

Verseghy, D. L., N. A. McFarlane, and M. Lazare (1993), CLASS – a Canadian land surface scheme for GCMs, II. Vegetation model and coupled runs, *International Journal of Climatology*, *13*(4), 347-370.

Viana, F. A. C., and R. T. Haftka (2008), Using multiple surrogates for metamodeling, Proceedings of the 7th ASMOUKISSMO International Conference on Engineering Design Optimization, Pages: 1-18, edited.

Viana, F. A. C., V. Picheny, and R. T. Haftka (2010), Using Cross Validation to Design Conservative Surrogates, *Aiaa Journal*, *48*(10), 2286-2298.

Viana, F. A. C., V. Steffen, Jr., S. Butkewitsch, and M. d. F. Leal (2009), Optimization of aircraft structural components by using nature-inspired algorithms and multi-fidelity approximations, *Journal of Global Optimization*, *45*(3), 427-449.

Vila, J. P., V. Wagner, and P. Neveu (2000), Bayesian nonlinear model selection and neural networks: A conjugate prior approach, *IEEE Transactions on Neural Networks*, *11*(2), 265-278.

Villa-Vialaneix, N., M. Follador, M. Ratto, and A. Leip (2011), A comparison of eight metamodeling techniques for the simulation of N2O fluxes and N leaching from corn crops, *Environmental Modelling & Software (to appear: this Thematic Issue)*.

Vitali, R., R. T. Haftka, and B. V. Sankar (2002), Multi-fidelity design of stiffened composite panel with a crack, *Structural and Multidisciplinary Optimization*, *23*(5), 347-356.

Vrugt, J. A., B. A. Robinson, and J. M. Hyman (2009a), Self-Adaptive Multimethod Search for Global Optimization in Real-Parameter Spaces, *IEEE Transactions on Evolutionary Computation*, *13*(2), 243-259.

Vrugt, J. A., B. O Nuallain, B. A. Robinson, W. Bouten, S. C. Dekker, and P. M. A. Sloot (2006a), Application of parallel computing to stochastic parameter estimation in environmental models, *Computers & Geosciences*, *32*(8), 1139-1155.

Vrugt, J. A., H. V. Gupta, S. C. Dekker, S. Sorooshian, T. Wagener, and W. Bouten (2006b), Application of stochastic parameter optimization to the Sacramento Soil Moisture Accounting Model, *Journal of Hydrology*, *325*(1-4), 288-307.

Vrugt, J. A., C. J. F. ter Braak, C. G. H. Diks, B. A. Robinson, J. M. Hyman, and D. Higdon (2009b), Accelerating Markov Chain Monte Carlo Simulation by Differential Evolution with Self-Adaptive Randomized Subspace Sampling, *International Journal of Nonlinear Sciences and Numerical Simulation*, *10*(3), 273-290.

Wagner, T., M. Emmerich, A. Deutz, and W. Ponweiser (2010), On Expected-Improvement Criteria for Model-based Multi-objective Optimization, in *Parallel Problems Solving from Nature - Ppsn Xi, Pt I*, edited by R. C. C. K. J. R. G. Schaefer, pp. 718-727.

Wahba, G. (1990), *Spline Models for Observational Data*, Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania, USA.

Wang, G. G. (2003), Adaptive Response Surface Method using inherited Latin Hypercube Design points, *Journal of Mechanical Design*, *125*(2), 210-220.

Wang, G. G., and S. Shan (2007), Review of metamodeling techniques in support of engineering design optimization, *Journal of Mechanical Design*, *129*(4), 370-380.

Wang, L. Q., S. Q. Shan, and G. G. Wang (2004), Mode-pursuing sampling method for global optimization on expensive black-box functions, *Engineering Optimization*, *36*(4), 419-438.

Wang, Q. J. (1991), The Genetic Algorithm and Its Application to Calibrating Conceptual Rainfall-Runoff Models, *Water Resour. Res.*, *27*(9), 2467–2471.

Watson, A. G., and R. J. Barnes (1995), Infill sampling criteria to lacate extremes, *Mathematical Geology*, *27*(5), 589-608.

Welch, W. J., R. J. Buck, J. Sacks, H. P. Wynn, T. J. Mitchell, and M. D. Morris (1992), Screening, predictiong, and computer experiments, *Technometrics*, *34*(1), 15-25.

Willcox, K., and A. Megretski (2005), Fourier series for accurate, stable, reduced-order models in large-scale linear applications, *Siam Journal on Scientific Computing*, *26*(3), 944-962.

Willmes, L., T. Back, Y. C. Jin, and B. Sendhoff (2003), Comparing neural networks and Kriging for fitness approximation in evolutionary optimization, in *Cec: 2003 Congress on Evolutionary Computation, Vols 1-4, Proceedings*, edited, pp. 663-670, IEEE, New York.

Xia, Y. L., Z. L. Yang, C. Jackson, P. L. Stoffa, and M. K. Sen (2004), Impacts of data length on optimal parameter and uncertainty estimation of a land surface model, *Journal of Geophysical Research-Atmospheres*, *109*(D7).

Xiang, C., S. Q. Ding, and T. H. Lee (2005), Geometrical interpretation and architecture selection of MLP, *IEEE Transactions on Neural Networks*, *16*(1), 84-96.

Xu, Y., K. W. Wong, and C. S. Leung (2006), Generalized RLS approach to the training of neural networks, *IEEE Transactions on Neural Networks*, *17*(1), 19-34.

Yan, S., and B. Minsker (2006), Optimal groundwater remediation design using an Adaptive Neural Network Genetic Algorithm, *water resources research*, *42*(5).

Yan, S., and B. Minsker (2011), Applying Dynamic Surrogate Models in Noisy Genetic Algorithms to Optimize Groundwater Remediation Designs, *Journal of Water Resources Planning and Management-Asce*, *137*(3), 284-292.

Yan, S. Q., and B. Minsker (2006), Optimal groundwater remediation design using an Adaptive Neural Network Genetic Algorithm, *water resources research*, *42*(5).

Yapo, P. O., H. V. Gupta, and S. Sorooshian (1996), Automatic calibration of conceptual rainfall-runoff models: Sensitivity to calibration data, *Journal of Hydrology*, *181*(1-4), 23-48.

Yapo, P. O., H. V. Gupta, and S. Sorooshian (1998), Multi-objective global optimization for hydrologic models, *Journal of Hydrology*, *204*(1–4), 83-97.

Ye, K. Q., W. Li, and A. Sudjianto (2000), Algorithmic construction of optimal symmetric Latin hypercube designs, *Journal of Statistical Planning and Inference*, *90*(1), 145-159.

Young, P. C., and M. Ratto (2009), A unified approach to environmental systems modeling, *Stochastic Environmental Research and Risk Assessment*, *23*(7), 1037-1057.

Yu, P. S., S. T. Chen, and I. F. Chang (2006), Support vector regression for real-time flood stage forecasting, *Journal of Hydrology*, *328*(3-4), 704-716.

Zhang, Q., W. Liu, E. Tsang, and B. Virginas (2010), Expensive Multiobjective Optimization by MOEA/D With Gaussian Process Model, *IEEE Transactions on Evolutionary Computation*, *14*(3), 456-474.

Zhang, X. S., R. Srinivasan, and M. Van Liew (2009), Approximating SWAT model using artificial neural network and support vector machine, *Journal of the American Water Resources Association*, *45*(2), 460-474.

Zou, R., W. S. Lung, and J. Wu (2007), An adaptive neural network embedded genetic algorithm approach for inverse water quality modeling, *Water Resources Research*, *43*(8).

Zou, R., W. S. Lung, and J. Wu (2009), Multiple-pattern parameter identification and uncertainty analysis approach for water quality modeling, *Ecological Modelling*, *220*(5), 621-629.