ABSTRACT

MIRGHANI, BAHAELDIN YOUSIF AHMED. Evolutionary Algorithms-based Parallel Simulation-Optimization Framework for Solving Inverse Problems. (Under the direction of G. Mahinthakumar and S. Ranji Ranjithan.)

Inverse problems are computationally challenging to solve due to inherent ill-posedness and computational intractability. In this dissertation the use of a simulation-optimization approach that couples a numerical simulation model with evolutionary algorithms for solution of the inverse problem is adopted. In this approach, the simulation model is solved iteratively during the evolutionary search, which in general can be computationally intensive since a large number of forward model evaluations are typically required for solution. Parallel simulation models and surrogate models are investigated with the aim of improving the simulation model efficiency. Numerical search methods such as parallel hybrid methods and noisy genetic algorithms are investigated for optimization algorithm improvement. In addition, high performance computing and grid computing are explored as a means to facilitate computationally tractable solution of such problems. In this dissertation, the solution of a groundwater inverse problem is explored to validate and illustrate the methods, an array of illustrative groundwater inverse problem applications are demonstrated. The results tend to confirm the effectiveness of the parallel simulation and surrogate modeling for improving the simulation model executing time. Also the results support and illustrate the advantage of using the newly developed EA-based parallel hybrid and noisy genetic algorithms that enhance the efficiency of solving the inverse problem. Finally, the high performance computational experiments performed on the National Science Foundation's TeraGrid resources demonstrate the effectiveness of the grid-enabled simulation-optimization approach in improving solution time.

Evolutionary Algorithms-based Parallel Simulation-Optimization Framework for Solving Inverse Problems

by

BahaEldin Yousif Ahmed Mirghani

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Civil, Construction and Environmental Engineering

Raleigh, North Carolina

2007

APPROVED BY:

Prof. E. Downey Brill, Jr.	Prof. John W. Baugh, Jr.
Dr. S. Ranji Ranjithan (Co-chair of Advisory Committee)	Dr. G. Mahinthakumar (Chair of Advisory Committee)

DEDICATION

This dissertation is dedicated to my parents, Mrs. Thoriya A. Idrees and Dr. Yousif A. Mirghani; and to my children, Mohamed B. Mirghani and Lena B. Mirghani with great respect and love.

BIOGRAPHY

BahaEldin Yousif Ahmed Mirghani was born in Medani, Sudan. He was raised within a typical extended Sudanese family with his siblings. By 1987 he graduated in Wad Medani High School. He obtained a Bachelor of Science degree in Civil Engineering from Mansoura University, Egypt in July 1992. Mr. Mirghani started his career as a civil engineer in a government sector in Sudan. In October 1996 he joined the International Institute for Infrastructural Hydraulic and Environmental Engineering (UNESCO-IHE), in Delft, the Netherlands for his Masters of Science in Civil Engineering, Hydrology and Water Resources. He completed his degree in July 1998 under the direction of Prof. M. J. Hall, the theme of his study was investigating applications of rainfall-runoff computer models to simulate arid and semi-arid zones. His interest in civil engineering researches in combination with information technology leaded him to University of Detroit Mercy to complete his second Masters of Science in Computer Science in December 2001. Mr. Mirghani started his doctoral program in Computer-Aided Engineering in Civil Engineering Department at North Carolina State University in August 2003. He worked as a graduate research assistant under Dr. G. Mahinthakumar and Dr. S. Ranji Ranjithan while working on his doctoral program. His main research interests are inverse problems modeling, design and development of evolutionary algorithms for engineering problems, high performance computing, grid computing, and software development.

ACKNOWLEDGMENTS

First of all, I would like to thank God, Allah, for providing me with faith, guidance, strength and patience to complete this work.

I would like to thank Dr. G. Kumar Mahinthakumar, my adviser and committee chairperson, for his fruitful advice and encouragement throughout this process. His tremendous support and availability for discussion contribute to the completion of this research. I would like to express my profound thanks to Dr. S. Ranji Ranjithan, my co-advisor and committee member, not only for his valuable insights, effort, and enthusiasm but also for his encouragement at critical moments during my research. I would like to express my gratitude to my other committee members, Professors E. Downey Brill Jr. and John W. Baugh Jr., for sharing their knowledge, effort, time and invaluable suggestions.

I would like to acknowledge my friends and/or officemates for their help, fellowship, and constructive research discussion including Kaite Wang, Mansour Malik, Matthew Clayton, Praveen Vankayala, Xin Jin, and Yong Jung. I would like also to acknowledge my friends and colleagues Michael Tryby for his research collaborations and discussions in the design of a computational framework for the implementation of part of this research, and Emily Zechman for her contribution and input in the development of the surrogate models.

I would like to acknowledge the TeraGrid Sites (NCSA, SDSC and UC/ANL) for providing the supercomputer resources necessary for this research. This work was partly supported by National Science Foundation (NSF) under Grant Number BES-0238623.

My family in Sudan has been essential sources of power, support and inspiration to me including my parents and siblings. They always believed in me and went through the longing of being apart. Finally, my heartfelt gratitude goes to my dear friend and wife, Eiman M. Abbas, for her continual encouragement, patience, understanding and taking care of our children throughout all phases of this process.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

System characterization using limited observational data collected at measurement stations is generally classified as an inverse problem. Solving inverse problems is relatively complex due to the ill-posedness present in the problem (Sun, 1994). Among several available methods, the simulation-optimization approach is a technique utilized to solve inverse problems by formulating them as an optimization model and solving them by coupling search procedures with the simulation models (Atmadja and Bagtzoglou, 2001). Some simulation models, usually referred to as forward models, are a system of partial differential equations (PDEs) that describes the governing processes of a system and defines the relationships between system inputs and outputs. The inverse problem is solved using a simulation-optimization approach via search algorithms to identify the best system characteristics that minimize the error between the model predictions and the system observations. While this approach is generic and robust, it is computationally expensive as it requires iterative executions of the forward model (Andrad´ottir, 1998).

The complexity of solving such problems is directly proportional to the number of system inputs to be determined. An optimization search paired with a numerical simulation model poses high computational burdens since each simulation model execution can be expensive. In this study, evolutionary algorithms-based search procedures are employed to solve the inverse problems using a simulation-optimization approach. The focus of this dissertation is to investigate novel approaches to address the algorithmic and computational issues associated with the proposed solution method. This consists of three major

components: improving the execution efficiency of the simulation model; improving the effectiveness and efficiency of the search algorithms; and facilitating a distributed computational framework to support the simulation-optimization approach.

While evolutionary algorithms (EAs) are good global search procedures, they deem inefficient, when compared with local search procedures, in refining the solutions beyond a certain degree of convergence (Mahinthakumar and Sayeed, 2005). This dissertation research explores new techniques for integrating the good features of the global and local search procedures such that the overall search efficiency is improved. In real-world application, inverse problems occur under noisy environments. Thus, another methodological investigation of this dissertation research is to develop a new algorithm for search under noisy conditions. In addition, recent sciences and technologies have become available through parallel computing and soft computing approaches to expedite the executing time of simulation models. This research investigates techniques to improve the computational efficiency of the simulation model. A fine-grained parallel implementation of the simulation model and a fast surrogate model to approximate the simulation model were also explored.

Finally, given the computational resources demands and the inherent parallelism present in the proposed simulation-optimization approach, an appropriate parallel/distributed computing framework can be utilized to improve the computational efficiency. Recent investments in national high-speed network infrastructure have allowed the aggregation of geographically distributed high-performance computing resources into computational grids. Because computational grids promote reliable and economical access to high-end computing resources, they have emerged as a new paradigm in scientific and engineering computation (Catlett, 2002; 2005). Computational grids have the potential to enable the solution of inverse

problems that previously would not have been feasible. Investigations and results based on computational experiments performed on the National Science Foundation's (NSF's) TeraGrid demonstrate the efficiency of the grid-enabled simulation-optimization approach in terms of solving accurately the unknown system inputs and improving the computational performance.

Many instances of both research as well as real-world applications are posed as inverse problems. For example, in groundwater contamination characterization problems, the unknown contamination location and the release profile are estimated based on observation data. Similarly, in drinking water distribution system security management, monitoring data from a contamination sensor network is used to determine the contamination potential locations and characteristics so that appropriate evasive actions could be expeditiously determined and executed (Ostfeld and Salomons, 2005). Another application is in medical imaging where the exact properties of internal organs are assessed via non-invasive screening and diagnosis.

Throughout this study, the following three groundwater inverse problems were used to test and illustrate the methodologies: (1) the source history reconstruction (SHR) problem; (2) the contaminant source identification (CSI) problem; and (3) the combined problem, in which both the release history and the source location (SHR-CSI) are identified. Also, several instances of these problems were utilized to test the performance of the computational framework that was developed as part of this dissertation research. The SHR, CSI and SHR-CSI problems are important in environmental forensics and contaminant characterization for many reasons, including for the purposes of regulatory enforcement and liability assessment (Poeter and Hill, 1997). In this problem context, source locations and/or

contaminant release concentrations are unknown model inputs, which are resolved from a set of spatially and temporally distributed observational data collected at monitoring wells. This dissertation discusses the solution of the groundwater inverse problems in two and three-dimensional groundwater domains.

The research investigations and their results to achieve the objectives of this research as described above are described herein as a collection of journal articles that report the solution approaches, methodological developments and results of their applications. High performance computing approach for efficiency improvements in simulation models are discussed in Chapter 2. This chapter utilizes an illustrative application using three-dimensional groundwater inverse problem (Mirghani et al., 2007a). A surrogate modeling approach for efficiency improvements in simulation models is discussed in Chapter 3. An artificial neural network-based procedure is used as a surrogate model to expedite the execution time of simulation models. Instances of both two and three-dimensional groundwater source identification problems are used as case studies to illustrate the approach (Mirghani et al., 2007b). Chapter 4 presents the development of the new parallelizable, thus computationally efficient, hybrid search methods that were built upon evolutionary computation algorithms. The new methods combine the power of local search into a global search method to maintain the inherent parallelism, and were demonstrated through two illustrative application problems. Chapter 5 demonstrates the development of a new noisy genetic algorithm-based procedure to efficiently address problems under uncertainty conditions. The new procedure is tested and applied to groundwater inverse applications with uncertain parameters. A grid-based computational framework development and its performance evaluation are described in Chapter 6. The framework was tested using single

and cross site runs utilizing the TeraGrid sites (Mirghani et al., 2007c). Chapter 7 summarizes the overall conclusions, final findings and the major research contributions.

CHAPTER 2

COMPUTATIONAL SIMULATION-OPTIMIZATION APPROACH FOR SOLVING INVERSE PROBLEMS

Abstract

Groundwater environmental characterization involves the resolution of unknown system characteristics from observation data, and is classified as an inverse problem. Inverse problems are relatively challenging to solve due to natural ill-posedness and computational intractability. Here we adopt the use of a simulation-optimization approach that couples a numerical pollutant-transport simulation model with evolution computation algorithms for solution of the inverse problem. In this approach, the numerical transport model is solved iteratively during the evolutionary search, which in general can be computationally intensive since several hundreds to thousands of forward model evaluations are typically required for solution. Given the potential computational intractability of such a simulation-optimization approach, parallel computation is employed to ease and enable the solution of such problems. In this paper, the solution of two groundwater inverse problems will be explored. The computational experiments were performed on the TeraGrid available on the National Center for Supercomputing Applications. The results demonstrate the performance of the parallel simulation-optimization approach in terms of solution quality and computational performance.

2.1 Introduction

Contaminant source identification problems are important in environmental forensics and characterization of contamination for the purposes of regulatory enforcement and liability assessment. In this problem context, source locations and historical contaminant release schedules are unknowns, and are resolved from the spatially and temporally distributed observations collected at contaminant monitoring wells. Such problems, where system characteristics are estimated from sparse observational data, are classified as inverse problems. A forward model, usually represented by a system of partial differential equations (PDEs), describes the transport processes of the groundwater system and defines the relationship between system inputs and outputs. Comprehensive overviews of these types of problems are described by Sun (1994) and Yeh (1986).

In general, inverse problems could be described as problems where the response of the system is known, but not the conditions that led to it. In the context of numerical modeling, finding model input parameter values given output data can be categorized as a problem of time inversion. This means that for time dependent systems, we have to solve the governing equations backward in time through "inverse modeling" (Atmadja and Bagtzoglou, 2001). Inverse problems are generally difficult to solve due to ill-posedness arising from: (1) non-existence; (2) non-uniqueness; and (3) instability of the solution (Sun, 1994). The solution complexity of such problems depends on the amount of observations available and the number of system inputs that must be determined.

Simulation-optimization approach is among several approaches used to solve inverse problems (Andradóttir, 1998; Azadivar, 1999). In a simulation-optimization approach, the simulation model is coupled loosely/tightly with an optimization technique to determine the

model inputs that best represent the observed data (Figure 2-1). This paper describes the investigation of a simulation-optimization solution approach that employs an evolutionary algorithm-based search method. A simulation-optimization approach to solve an inverse problem is several orders of magnitude more computationally challenging than the solution of the corresponding forward model, since the optimization procedure is based on an iterative process that requires many forward model evaluations.

The computational tractability of such a simulation-optimization approach could be enhanced by improving the efficiency of both the simulation model and the optimization method. Several approaches, such as surrogate modeling and code parallelization, are available to improve the simulation model efficiency by reducing the forward model solution time. In the research presented in this paper, the simulation model parallelism approach is investigated to improve the execution time of the simulation model. This approach is based on high performance computing technologies that are used to expedite the execution time by harnessing the fine and coarse gained parallelism exhibited by the simulation model (Sayeed and Mahinthakumar, 2002).

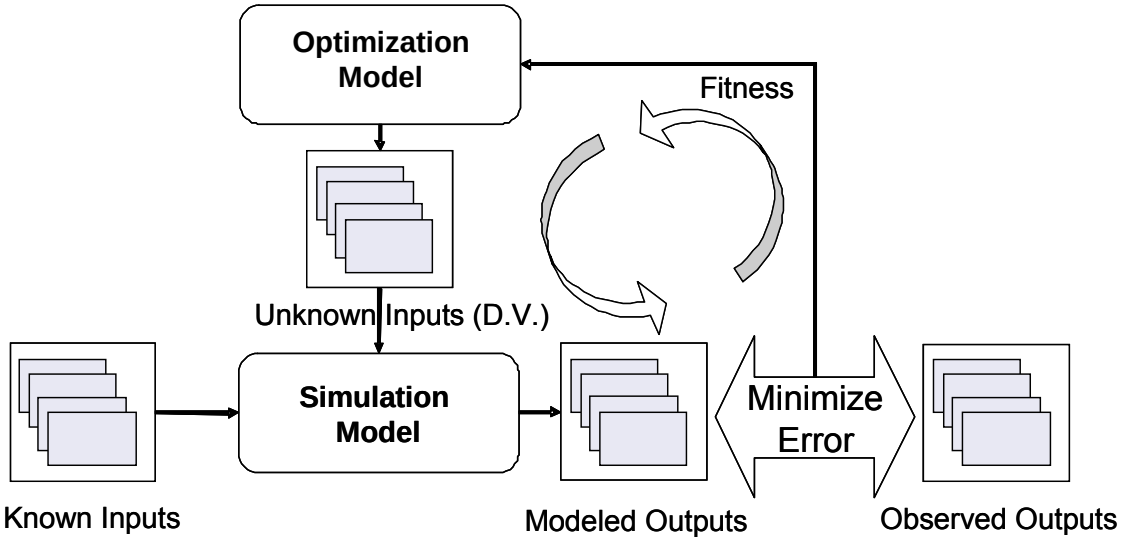

Figure 2-1. A Schematic of a Simulation-Optimization Approach

The main goal of this research is to enhance the efficiency of the simulation-optimization approach utilizing high performance technologies to reduce the overall computation time for solving inverse problems. This approach is illustrated by applying it to solve instances of a groundwater inverse problem. Many approaches have been considered to solve this problem with varying degree of success. These include regularization and stabilization (e.g., Neupauer et al., 2000), linear programming and multiple regression (e.g., Gorelick et al., 1983), non-linear maximum likelihood estimation (e.g., Wagner, 1992), integer programming (e.g., Mahar and Datta, 1997; 2000; 2001), stochastic differential equations backward in time (e.g., Neupauer and Wilson, 2001), correlation coefficient optimization method (e.g., Sidauruk et al., 1998); and heuristic methods (e.g., Mahinthakumar and Sayeed, 2005; Mirghani et al., 2006). An overview of mathematical methods for groundwater pollution source identification is provided by Atmadja and Bagzogolu (2001).

Singh et al. (2006) report a simulation-optimization approach to solve a groundwater contaminant source identification problem. In contrast to their work, the research presented in this paper considers three-dimensional heterogeneous and homogeneous flow field problems with increased problem complexity (four to seven parameters to be estimated) where the source could be located anywhere in the aquifer (instead of at a set of predefined potential source locations), and the solution method uses a loosely coupled simulation-optimization approach as opposed to a tightly coupled simulation-optimization approach. In addition, our research aims to improve the efficiency of a generic simulation-optimization approach, and is not limited to groundwater inverse problems.

The following section describes the overall source characterization inverse problem, followed by a description of the solution approach. The subsequent section presents the problem scenarios that were investigated and reported in this paper. The results section presents the quality of the solutions obtained for the different scenarios and computational performance comparisons. The last section provides a summary and presents some observations.

2.2 Problem Overview

In this study a three-dimensional groundwater contaminant source characterization problem is considered, where a three-dimensional groundwater transport model is employed to simulate the fate and transport of a contaminant plume. The governing equations describing the groundwater transport are fully explained in Section 2.3.1. This problem represents an unknown contaminant release at a single contaminant source (or multiple contaminant sources) that is to be resolved from spatially and temporally distributed concentration observations collected at prespecified monitoring wells. The problem assumes that the contaminant source locations and the contaminant release at the sources are unknown. Concentration observations at the monitoring wells are compiled to generate a concentration time-series at each monitoring location. For a possible contamination source characteristics (defined by the source location, size and concentration of the contaminant), the resulting concentrations at the monitoring wells are estimated using a groundwater transport model (forward model). The optimization method searches to identify the source characteristics that yield concentration estimates that best match the observations. Figure 2-2

illustrates the groundwater source identification problem in a three-dimensional groundwater domain.

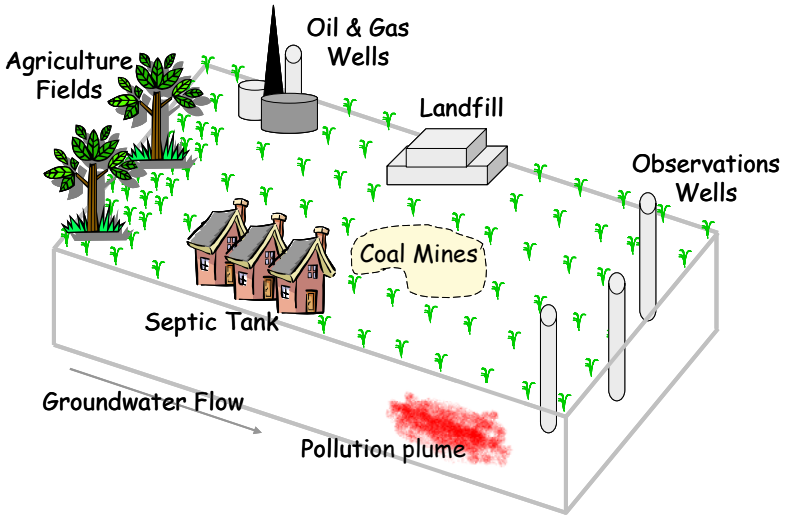

Figure 2-2. Hypothetical Three-Dimensional Domain Illustrating the Groundwater Inverse Problem

The inverse problem could be posed as a constrained optimization model where the prediction error between the observed and simulated concentrations is minimized. Equation 2-1 illustrates a typical error minimization objective function. Numerous constraints are typically included to enforce decision variable bounds and feasibility.

$$\underset{DV}{\text{minimize}} \sum_{j=1}^{nw} \sum_{i=1}^{n} \left| C_{ij}^{obs} - C_{ij}^{sim} \right| \qquad (2\text{-}1)$$

where $\mathbf{C}_{ij}^{obs}$ is the observed concentration estimated using the groundwater forward model for time step i at monitoring well j, n is the number of contamination simulation time steps, nw is the number of monitoring wells, $\mathbf{C}_{ij}^{sim} = f$ *(contaminant source location, size, concentration)* is the simulated contaminant concentration at monitoring well j and time step i, and DV is the set of decision variables used to represent the source characteristics. Depending on the

number of contaminant sources, the number of decision variables is equal to the product of the number of contaminant sources times the number of unknowns used to describe each source.

2.3 Solution Approach

The overall simulation-optimization solution framework is developed upon an evolutionary algorithm-based search procedure that is coupled with a parallel groundwater simulation model (i.e., the simulation model in Figure 2-1 is substituted with a parallel groundwater simulation model). Each of the key components of this framework is described in the following subsections.

2.3.1 Simulation Model -- Groundwater Transport Model

The simulation model used as an illustration in solving a groundwater inverse problem in this study is the Parallel Groundwater transport and REMediation code (PGREM3D). This is a suite of massively parallel codes applied for numerical simulation, using the finite element method, of three-dimensional groundwater transport and remediation problems, originally developed by Mahinthakumar (1999). The flow module solves the steady state groundwater flow equation (Bear, 1979) as described below:

$$\nabla(K\nabla h) = q \qquad (2\text{-}2)$$

where h is the computed head field from the groundwater flow equation, K is the 3x3 hydraulic conductivity tensor, θ is the porosity, and q represents the source/sink terms coming from injection/pumping wells. The hydraulic head field h is computed from Equation 2-2, and then the corresponding velocity field v is computed using Darcy's law

$$\theta v = -K\nabla h \qquad (2\text{-}3)$$

2.3.1.1 The Transport Governing Equations

The system of equations describing the transport of each dissolved contaminant *i (i = 1, 2, ..., nc*, where *nc* is the total number of contaminants) undergoing reactions in saturated porous media is defined by

$$\frac{\partial C_i}{\partial t} = \nabla\cdot(D\cdot\nabla C_i) - \nabla\cdot(C_i v) + \frac{q}{\theta}(C_i - C_{0i}) - R_i \qquad i = 1,\ 2,\ ...,\ nc \qquad (2\text{-}4)$$

where v is the 3x1 velocity field vector, D is the 3x3 dispersion tensor dependent on v, and C_i is the dissolved concentration of contaminant i. The term $q(C_i\text{-}C_{0i})/\theta$ represents the source term with volumetric flux q, medium porosity θ, and injected concentration C_{0i} . R_i is the rate of mass loss of contaminant i due to reactions. More detailed description is provided by Mahinthakumar (1999).

2.3.1.2 Numerical Methodology

The numerical methodology used in the modules, i.e., the flow and transport codes, is the Galerkin finite element method with eight-node linear hexahedral elements. A logically rectangular grid structure is assumed but irregular geometries are supported using distorted elements. Crank-Nicolson approximation is employed for the time derivative terms. A lumped mass formulation (Huyakorn and Pinder, 1983) is used for all time-derivative and non-derivative (zeroth spatial derivative) terms. The coupled non-linear system is solved using a modified form of the Sequential Iterative Algorithm (SIA). Several Krylov subspace iterative solvers are used for the matrix solution (Mahinthakumar et al., 1997).

2.3.1.3 Parallel Implementation

The transport module simulator codes are written in FORTRAN using double-precision arithmetic. The codes are parallelized using a two-dimensional domain decomposition (in the *x* and *y* directions). Explicit message passing interface library (MPI) was utilized to exchange information between these domains. (More details on the use of MPI can be found in many references, such as Gropp et al., 1999). The MPI provides portability to a wide range of parallel architectures, where the use of the MPI communicators is a convenient way to couple processes. The simulation model is implemented to accommodate fine, semi-coarse and coarse grained parallelism. Fine grained parallelism is applied by executing the simulation model on multiple computer processors via the MPI communication library. The semi-coarse grained parallelism is implemented using the MPI Group library, in which multiple instances of the simulation model are executed simultaneously in groups. The coarse grained parallelism is implemented by using multiple servers executed simultaneously; each server consists of a number of groups. This technique is illustrated below considering that a given total number of processors are available. Each server consists of a number of processors equal to [total number of processors/number of servers]. In each server one processor P_0^{SM} acts as a server master to establish communication between the server and the simulator. The remaining number of processors [number of processors in a server-1] will be divided into number of groups. Thus, in each server, each group consists of a number of processors equal to [(number of processors in a server-1) /(number of servers)]. Within each group one processor acts as a group master P_0^{GM} to establish interprocessor communication with P_0^{SM}. A schematic of the simulation model parallelization with three levels of parallelization is shown in Figure 2-3.

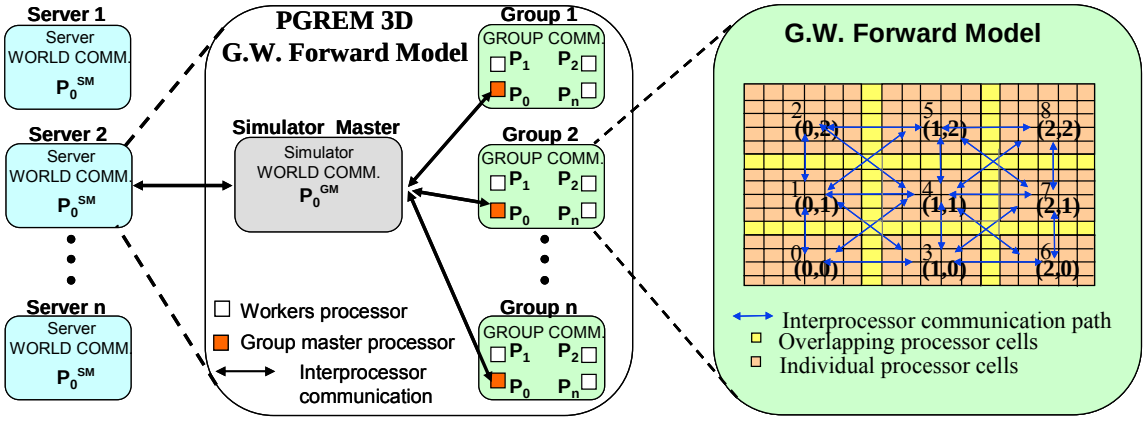

Figure 2-3. A Schematic of the Simulation Model Parallelization

2.3.2 Optimization Approach -- Evolutionary Algorithms

An evolutionary algorithm (EA) procedure is used as an optimization component in the simulation-optimization approach to solve the inverse problem. In general, EAs work on a population of solutions using several operators that evaluate the fitness of each solution, select better solutions, and create new sets of solutions by combing the features of relatively more successful solutions. These operators are in general applied probabilistically to explore the solution space and to conduct a stochastic search. Among several evolutionary computation approaches that are available, we investigated an approach based on evolution strategies (ES) in this paper. Figure 2-4 shows the key steps in an ES-based search procedure.

ES was developed in 1963 by Ingo Rechenberg and Hans-Paul Schwefel at the Technical University of Berlin (TUB) while solving an engineering optimization problem. ES is a stochastic search heuristic based on ideas of adaptation and evolution and is conceptually similar to natural evolution (Back, 1997). ES is similar to genetic algorithms (GAs) in many ways, but differs in the mechanics of some of the operators.

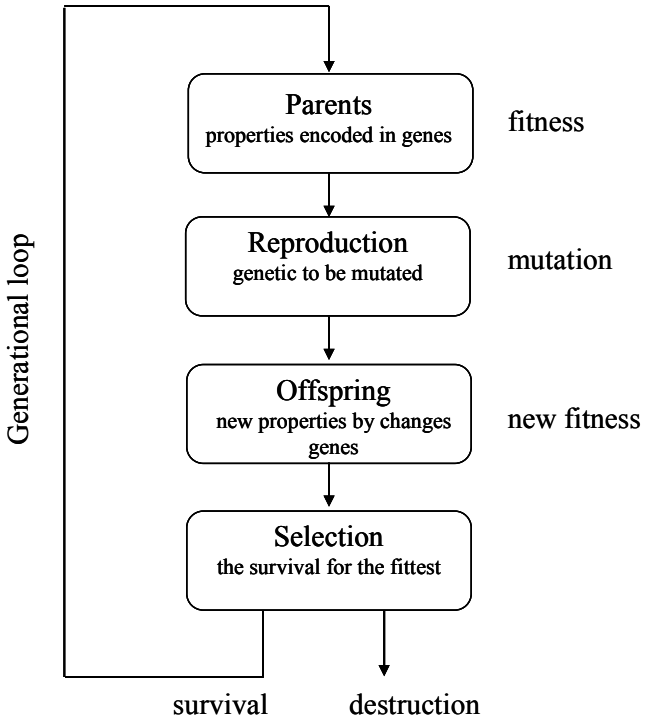

Figure 2-4. A Schematic of the Evolutionary Strategies Procedure, Modified from Beyer (2001)

In an ES-based procedure, each potential solution (i.e., an "individual") is encoded by a vector of real-valued "decision variables." ES uses primarily mutation and selection as search operators, which are applied iteratively (where each iteration is called a generation) as shown is Figure 2-4. The sequence of generations is continued until a specified set of termination criteria are met. Mutation is normally performed by perturbing a decision vector component by a normally distributed random value. The step size or mutation strength (i.e., the standard deviation of the normal distribution) is often determined via a self-adaptation mechanism. The selection is carried out deterministically based on fitness rankings. In the context of inverse problems, fitness is inversely proportional to the absolute difference between simulated and observed values, i.e., the prediction error. Calculating fitness for each solution requires a forward solve.

The simplest form of ES, called (1+1)-ES, operates on a population of size two: the current solution ("parent") and the offspring of its mutation ("mutant"). If the mutant has a higher fitness than the parent, it becomes the parent in the next generation, otherwise the mutant is discarded. Generally, λ mutants are generated and compete with the parent, this form of ES called (1+λ)-ES. Another form of ES are also available, known as (1,λ)-ES, where the best mutant becomes the parent of the next generation while the current parent is always discarded. The forth form of ES, called (μ/ρ, λ)-ES, usually use a population of μ parents and also recombination as an additional operator. The inherently parallel structure of an ES enables it be used in a parallel and/or distributed computing environment since the fitness evaluation for each individual in a generation can proceed independently, (Schwefel, 1995; Baeck, 1995).

2.3.3 Computational Framework Overview

The simulation-optimization approach is solved using the LAarge Scale Simulation-Optimization framework (LASSO); a description of the framework is available in Mirghani et al. (2005b). LASSO consists of a centralized optimization application that utilizes a master-worker task distribution strategy. The optimization, master, and worker processes are executed on parallel computational resources. The worker processes interface with instances of the forward model for distributed task execution. Results are returned to the master for processing by the centralized optimization application. In the following sections, key components of the application architecture are described in greater detail.

2.3.3.1 Framework Architecture

Several different search procedures have been implemented, making up the centralized optimization application. Here, the optimization model representation of the inverse problem is solved using evolution strategies. The ES-based procedure encodes, within an individual, the decision variables that describe a potential solution to the inverse problem. The objective function (representing prediction error) of the optimization model is used to quantify a fitness value indicating how well an individual solves the inverse problem. Fitness values are calculated using the results of forward model evaluations. The application of an ES-based search to inverse problems is advantageous because of their robustness and global search characteristics. Some drawbacks, however, include the computational intensity of a typical ES search and the slow final convergence prior to termination.

2.3.3.2 Framework Parallelism

The optimization application is coupled with the forward model (PGREM3D) for fitness evaluation. Each forward model evaluation is handled by a set of processors (typically 1-8, and is referred to as the number of processors per group). MPI is used to group processors, associate processors to computational domains, and for fine grained message passing within each of these groups. As mentioned earlier, the solution procedure adopted here involves multiple levels of parallelism: one level exhibited by the search procedure and the other two levels exhibited by the forward model. Each iteration of the search procedure exhibits a coarse grained parallel structure that requires an uncoupled forward model evaluation for each individual in the population. The optimization application acts as the master process in the master-worker task distribution strategy. The master, worker, and task

pool used in the framework were designed and implemented as part of Vitri (Baugh, 2003). The master maintains a pool of remote tasks − a bundle of individuals requiring evaluation. Aggregating individuals in this manner reduces communications overhead. Worker processes running on distributed computing resources, having established a TCP-IP socket connection with the master, signal their readiness and draw tasks from the task pool (Mirghani et al., 2005b). The worker process transfers the remote task to the MPI's zeroth processor. From this point forward, standard MPI group communications are utilized. The forward model manages multiple MPI groups, and each group evaluates an individual in the task bundle. The results of these simulations are then aggregated into a result bundle and returned to the optimization application for processing by the search algorithm. Finally, the subsequent generation of the search is initiated.

2.4 Illustrative Application Setting

Two sets of scenarios were conducted to illustrate the applicability and efficiency of the proposed simulation-optimization approach in terms of finding accurate solutions for a groundwater contaminant source identification problem and reducing the computational needs of the simulation-optimization approach. The first analysis demonstrates the solution quality of a three-dimensional contaminant source identification problem, where a set of scenarios with different levels of problem complexity were studied. The second analysis provides a timing study to show the performance of the proposed simulation-optimization approach, where the executing time was recorded while an increasing number of computational resources were used to solve the inverse problem. Intuitively, with increasing number of groups to solve the inverse problem the solution quality will remain unchanged.

In this section, the application problem scenarios along with the optimization formulation for each scenario are explained, and the groundwater fields and the simulation model parameters are described. Also, the optimization formulations for each problem instance are discussed. In addition, the computational framework settings are presented.

2.4.1 Application Scenarios

Three different scenarios were introduced based on how the contaminant source location is characterized. The scenarios were designed to illustrate the problem complexity in terms of the number of unknowns to be estimated and the effect of the groundwater velocity field on estimating these unknowns. In each scenario two cases were considered, namely, homogeneous and heterogeneous hydraulic conductivity fields with the appropriate steady-state hydraulic head and velocity distributions.

For the first scenario (referred to as 3D-1), the contaminant source is characterized by its centroid and contaminant initial concentration, and the simulated contaminant concentration for scenario 1 at a monitoring well is represented as $C_{ij}^{sim1} = f\ (x_c,\ y_c,\ z_c,\ C_0)$ where x_c, y_c and z_c are the coordinates for the centroid of the contaminant source, and C_0 is the initial source concentration.

In the second scenario (referred to as 3D-2), the contaminant source is characterized by the contaminant source location and concentration assuming that the source is cube shaped. Thus, the simulated contaminant concentration for scenario 2 at a monitoring well is represented as $C_{ij}^{sim2} = f\ (x_c,\ y_c,\ z_c,\ s,\ C_0)$ where s is the length of a side of the cube-shaped contaminant source.

Table 2-1. Design Variables for Each Scenario

Scenario	Design Variables
3D-1	(x_c, y_c, z_c, C_0)
3D-2	(x_c, y_c, z_c, s, C_0)
3D-3	$(x_1, y_1, z_1, x_2, y_2, z_2, C_0)$

For the third scenario (referred to as 3D-3), the source is characterized by the contaminant concentration and the coordinates of two diagonally opposite corners of a hexahedron. Thus, the simulated contaminant concentration for scenario 3 at a monitoring well is given by $C_{ij}^{sim3} = f(x_1, y_1, z_1, x_2, y_2, z_2, C_0)$ where x_k, y_k and z_k $(k = 1, 2)$ are the coordinates of the vertices of diagonally opposite corners of a hexahedron-shaped source. Table 2-1 summarizes the number of unknowns to be determined for each scenario.

2.4.2 Simulation Setup

A hypothetical three-dimensional (3D) groundwater domain, one with a homogeneous velocity field and another with a heterogeneous velocity field, was considered. The 3D domain is of dimensions 500m x 300m x 10m that was modeled using 51 x 31 x 11 finite element nodes. The simulation duration of 8000 days was modeled using 400 time steps, and the concentrations were recorded at 18 monitoring wells distributed evenly (9 wells each) in the middle and farthest end of the aquifer (Figure 2-5). For the homogeneous field, a mean flow velocity of 0.1 m/day is used. For the heterogeneous field, the steady state flow is computed using a randomly heterogeneous hydraulic conductivity field that was generated using a turning band code (Thompson et al., 1999). The homogeneous simulation model execution time on a single node at the TeraGrid-NCSA cluster is approximately 8 seconds while the heterogeneous model simulation requires approximately 20 seconds.

Detailed geometrical and hydraulic parameters are shown in Table 2-2, and in Figure 2-5. Representative simulated concentration profiles at monitoring wells 5 and 14 are shown in Figure 2-6.

Table 2-2. Hypothetical Three-Dimensional Domain Parameters

Parameter	Parameter Values for Homogeneous Field	Parameter Values for Heterogeneous Field
Problem Size	500m x 300 m x 10 m	
Grid Size	51x31x11 grid	
Number of Time Steps	400	
Time Step Size (dt)	20 day	
Grid Spacing	dx=10m, dy=10m, dz=1 m	
Dispersion Parameters	α_L=10m, α_{TH}=5m, α_{TV}=1m	
Flow Field	Homogeneous	Heterogeneous
Velocity	0.1 m/day	Spatially Variable
Executing Time	7.77 sec	19.19 sec
Source Location (3D-1)	(x_c= 9, y_c=10, z_c=4, C_0=70)	
Source Location (3D-2)	(x_c= 9, y_c=10, z_c=4, s= 2, C_0=70)	
Source Location (3D-3)	(x_1=8, y_1=9, z_1=3, x_2=10, y_2=11, z_2=5, C_0=70)	

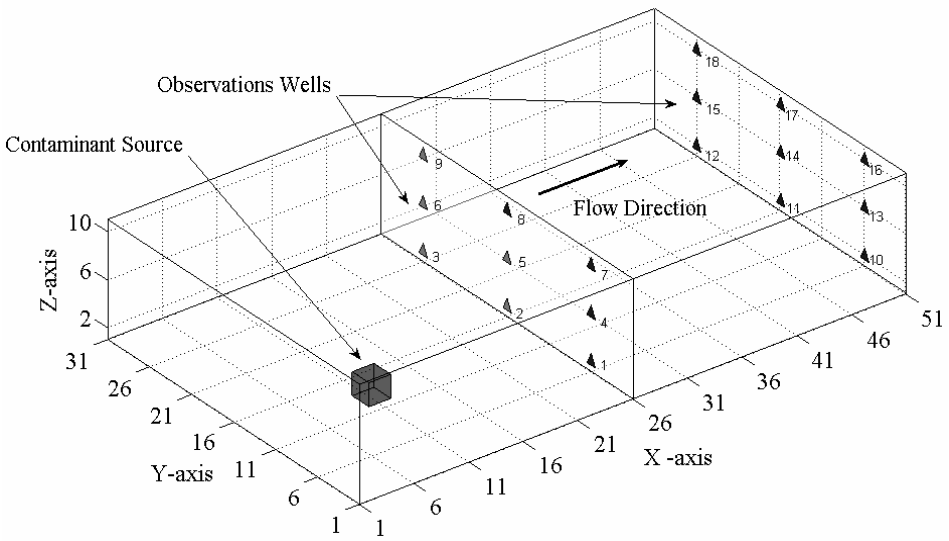

Figure 2-5. Hypothetical Three-Dimensional Domain

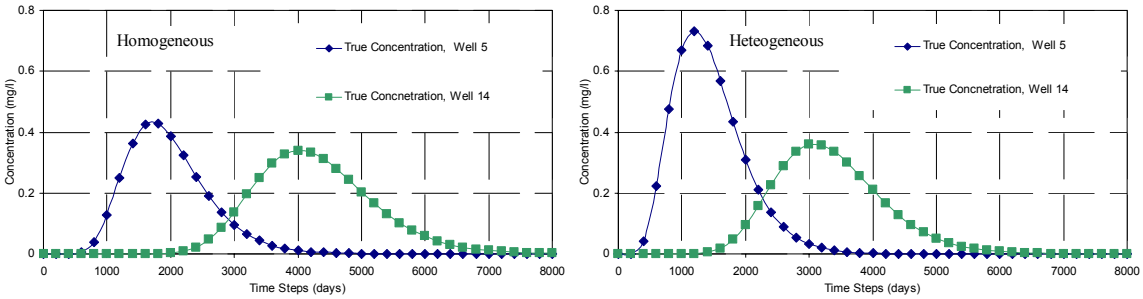

Figure 2-6. Concentration Profiles Monitored at Observation Wells 5 and 14 for Homogeneous and Heterogeneous Fields, respectively

2.4.3 Optimization Models

The contaminant source identification problem is posed as an optimization model where the prediction error denoted by the root squared error (RSE) between the observed and modeled concentrations at a set of monitoring wells is minimized. Equation 2-5 shows the prediction error function used for all scenarios. The constraints represented by Equations (2-6)-(2-11) are included to enforce decision variable bounds and feasibility. Equation 2-6 is applicable to all scenarios, Equation 2-7 is applicable to scenarios 3D-1 and 3D-2, Equations 2-8 and 2-9 are only applicable to scenario 3D-2, and Equation 2-10 and 2-11 are only applicable to scenario 3D-3.

$$\underset{DV^{sc}}{\text{minimize}} \sqrt{\sum_{j=1}^{nw}\sum_{i=1}^{n}\left(C_{ij}^{obs} - C_{ij}^{sim_{sc}}\right)^2} \qquad (2\text{-}5)$$

subject to:

$$0 \le C_0 \le C_{max} \qquad (2\text{-}6)$$

$$x_{min} \le x_c \le x_{max}; \qquad y_{min} \le y_c \le y_{max}; \qquad z_{min} \le z_c \le z_{max} \qquad (2\text{-}7)$$

$$s_{min} \le s \le s_{max} \qquad (2\text{-}8)$$

$$x_{min} \le x_c \pm s/2 \le x_{max}; \; y_{min} \le y_c \pm s/2 \le y_{max}; \; z_{min} \le z_c \pm s/2 \le z_{max} \qquad (2\text{-}9)$$

$$x_1 \le x_2; \qquad y_1 \le y_2; \qquad z_1 \le z_2 \qquad (2\text{-}10)$$

$$x_{min} \leq x_k \leq x_{max};\ y_{min} \leq y_k \leq y_{max};\ z_{min} \leq z_k \leq z_{max}, \quad k = 1,2 \tag{2-11}$$

where

DV^{sc} is the set of parameters to be estimated for scenario *sc*, where,

$$DV^1 = (x_c, y_c, z_c, C_0), \tag{2-12}$$

$$DV^2 = (x_c, y_c, z_c, s, C_0), \tag{2-13}$$

$$DV^3 = (x_1, y_1, z_1, x_2, y_2, z_2, C_0), \tag{2-14}$$

C_i^{obs} is the observed concentration at a number of monitoring wells downstream,

$C_i^{sim_{sc}}$ is the time-series of modeled monitoring concentrations,

x_{min}, x_{max}, y_{min}, y_{max}, z_{min} *and* z_{max} are the groundwater domain boundaries,

s_{min} *and* s_{max} are the maximum and minimum source size, respectively,

n is the total number of time steps,

nw is the total number of observations wells.

2.4.4 Computational Resources

The computational experiments presented here were performed on the NSF TeraGrid site. The TeraGrid is a heterogeneous computational resources distributed across the United States connected through a specialized interconnection network designed for high-band width data transfer (Catlett, 2002; 2005). The study was conducted on the IA-64 Intel Itanium2 cluster (1.5 Ghz, 256 computing nodes) available at National Center for Supercomputing Applications (TeraGrid-NCSA) at the University of Illinois at Urbana-Champaign.

2.5 Results and Discussion

This section presents the results in the context of: 1) the quality of solutions obtained using the ES-based simulation-optimization approach; and 2) computational performance

when the distributed computational framework is utilized to execute the new solution approach.

2.5.1 Contaminant Source Identification Problem Solution Quality

The results presented here for illustrative purposes are based on synthetically generated observations at the 18 monitoring wells. A "true" conservative contaminant source with a fixed continuous contaminant concentration was placed in the aquifer at a specific location. The groundwater contaminant transport model was executed to generate the concentration profiles at the 18 monitoring wells. These "observed" concentration profiles are used as the observations that should be matched by the source characteristics identified by the simulation-optimization method that minimizes the prediction error function in Eqn. 5. In this study the simulation model is assumed to perfectly generate the concentrations (i.e., no model error), and the measurements have no error.

Since the true source characteristics for these synthetically generated contamination problem scenarios are known, the quality of the identified solutions can be evaluated by comparing them to the known true source characteristics. A solution error metric (Eqn. 2-15) is introduced to evaluate in a post-processing step the quality of the obtained solution.

$$Solution\ Error\ (\%) = \left(\sqrt{\sum_{i=1}^{n} \left(\frac{p_i^{true} - p_i^{pre}}{p_i^{true}} \right)^2} \right) \times \frac{100}{n} \qquad (2\text{-}15)$$

where p_i^{true} is the true value of unknown (i.e., decision variable) i, p_i^{pre} is the predicted value of unknown i, and n is the number of unknowns.

Table 2-3. Common ES Settings Used for the Three Problem Scenarios

ES Parameter	Parameter Setting		
	Scenario 3D-1	Scenario 3D-2	Scenario 3D-3
ES Type	ES($\mu + \lambda$)		
μ	100	100	200
λ	100	100	200
Variables Encoding	Real		
Number of Generation	49	99	199
Number of Total Evaluations	5,000	10,000	40,000

Several trials were first conducted for each scenario to tweak and fine-tune the ES parameters, such as population size, number of generations, and mutation rate. The common ES parameter settings and the bounds on the decision variables used for the different problem scenarios are shown in Tables 2-3 and 2-4, respectively. As the ES-based search is a probabilistic method, a set of 30 random trials were conducted to evaluate the robustness of the procedure.

Table 2-4. Allowable Range of Decision Variable Values for all Scenarios

Variables	Ranges [Minimum, Maximum]
x_c, x_k	[1, 51]
y_c, y_k	[1, 31]
z_c, z_k	[1, 11]
s	[1, 10]
C_0	[0, 100]

The scenario 3D-1 was solved using the mutation rate of 1.0 and 0.1 for homogeneous and heterogeneous cases, respectively. Table 2-5 shows the best (among the 30 trials) predicted values of the decision variables, which are compared with the corresponding

true values. This table also lists the corresponding prediction errors (Eqn. 2-5) and the solution errors (Eqn. 2-7), which indicate good performance (nearly zero prediction error and less than 3% deviation from the true source characteristics) in characterizing the source. Figure 2-7 shows the true and the predicted source locations in the 3D domain for both cases. Figure 2-8 illustrates typical examples of the predicted concentration profiles in comparison to the true concentration profiles at two representative monitoring well locations.

Table 2-5. True and Predicted Source Characteristics for Scenario 3D-1

	Homogeneous Case		Heterogeneous Case	
	True Value	Predicted Value	True Value	Predicted Value
Variable x_c	9	8.72	9	9.45
Variable y_c	10	9.94	10	9.68
Variable z_c	4	3.92	4	4.33
Variable C_0	70	70.00	70	70.12
Prediction Error	-	0.000157	-	0.00654
Solution Error (%)	-	0.943%	-	2.554%

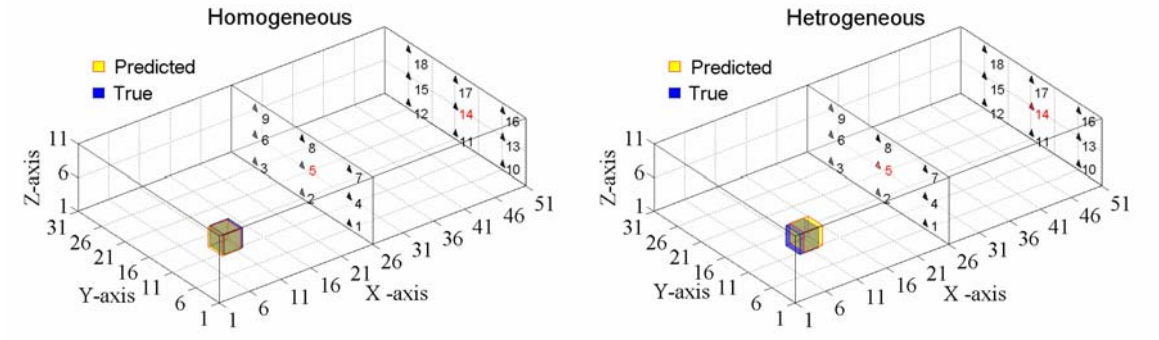

Figure 2-7. True and Predicted Source Locations for the Homogeneous and Heterogeneous Cases for Scenario 3D-1

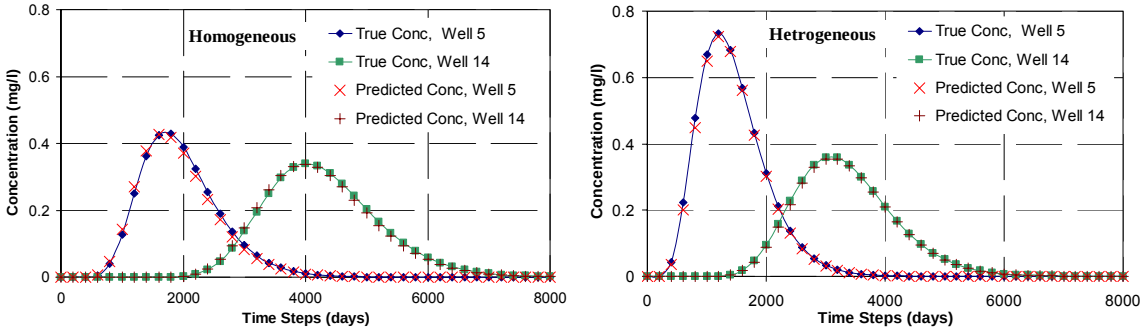

Figure 2-8. True and Predicted Concentration Profiles at Wells 5 and 14 for the Homogeneous and Heterogeneous Case for Scenario 3D-1

In scenario 3D-2, the mutation rate was set to 2.0 and 1.0 for the homogeneous case and heterogeneous case, respectively. Table 2-6 summarizes for both cases the predicted values of the decision variables, which are compared with the corresponding true values, and lists the corresponding prediction and solution errors. The estimated source locations and sizes are shown in Figure 2-9, and a comparison of predicted versus observed concentration values at two representative monitoring wells are shown in Figure 2-10. These results indicate that while the source characterization procedure is able to identify solutions with relatively low prediction errors, the solution error is relatively high (~12%). Most of the error is associated with the estimates for the z_c and s parameter values.

Table 2-6. True and Predicted Source Characteristics for Scenario 3D-2

	Homogeneous Case		Heterogeneous Case	
	True Value	Predicted Value	True Value	Predicted Value
Variable x_c	9	9.01	9	9.35
Variable y_c	10	9.63	10	9.59
Variable z_c	4	2.58	4	3.11
Variable s	2	3.15	2	3.17
Variable C_0	70	72.27	70	69.52
Prediction Error	-	0.130	-	0.02761
Solution Error (%)	-	13.55%	-	12.526%

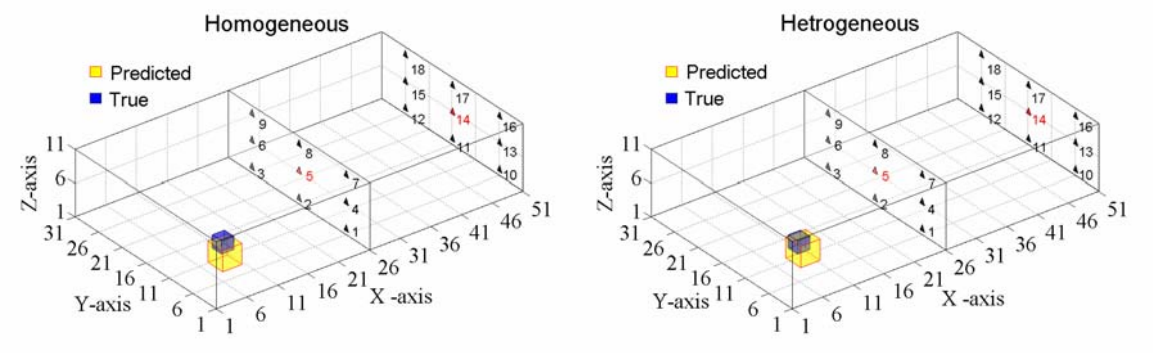

Figure 2-9. True and Predicted Source Locations for the Homogeneous and Heterogeneous Cases for Scenario 3D-2

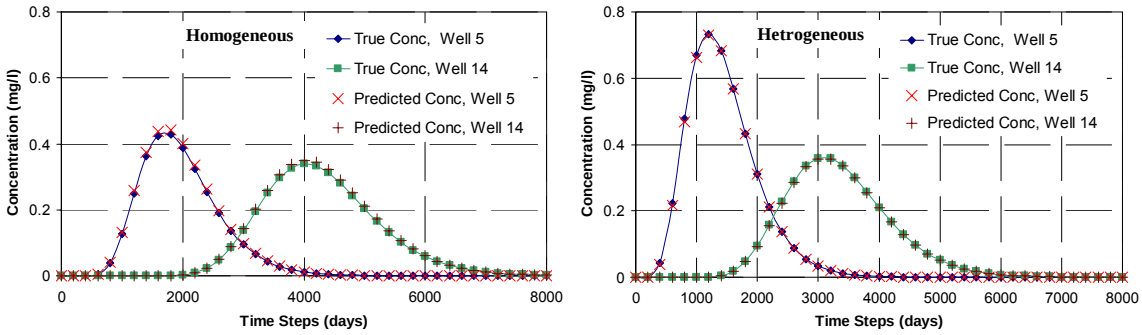

Figure 2-10. True and Predicted Concentration Profiles at Wells 5 and 14 for the Homogeneous and Heterogeneous Case for Scenario 3D-2

Similar to the previous two scenarios, both the heterogeneous and homogeneous cases were investigated for this scenario 3D-3, and the mutation rate was set to 0.1 for both cases. The predicted values for the seven decision variables, the prediction error and solution error are summarized in Table 2-7. The predicted source location characteristics and the corresponding concentrations at monitoring wells 5 and 14 are shown in Figure 2-11 and Figure 2-12, respectively. As in the previous scenario, the estimated source characteristics yield relatively low prediction error values while the solution error values remain higher compared to those for the previous two scenarios. Although the objective function error is not as low as in the other scenarios, the predicted source location is in close vicinity of the actual source. In general, the contaminant source identification problem is complex due to

the presence of non-uniqueness. The 3D-3 scenario in comparison to the 3D-2 and 3D-1 scenarios was computationally more challenging due to the higher degree of non-uniqueness posed by the larger number of unknowns (decision variables).

Table 2-7. True and Predicted Source Characteristics for Scenario 3D-3

	Homogeneous Case		Heterogeneous Case	
	True Value	Predicted Value	True Value	Predicted Value
Variable x_1	8	8.41	8	9.61
Variable y_1	9	8.10	9	7.02
Variable z_1	3	5.98	3	3.42
Variable x_2	10	10.76	10	10.58
Variable y_2	11	11.76	11	12.6
Variable z_2	5	9.61	5	8.91
Variable C_0	70	77.81	70	49.9
Fitness Error	-	0.202	-	0.157
Solution Error (%)	-	19.53%	-	12.99%

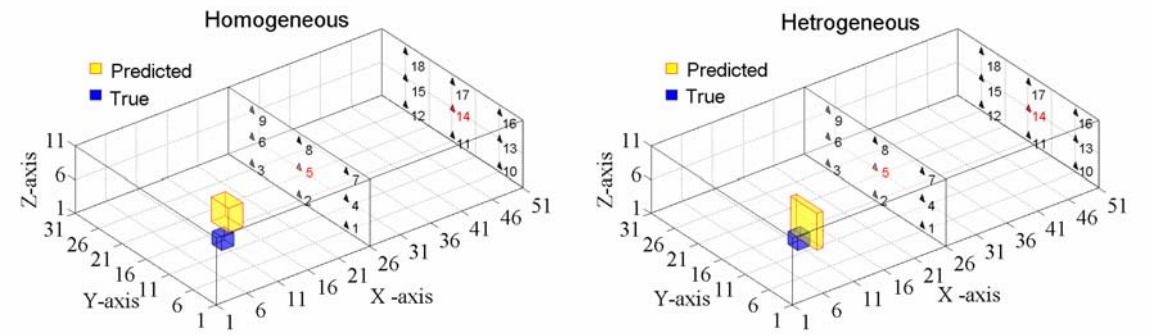

Figure 2-11. True and Predicted Source Locations for the Homogeneous and Heterogeneous Cases for Scenario 3D-3

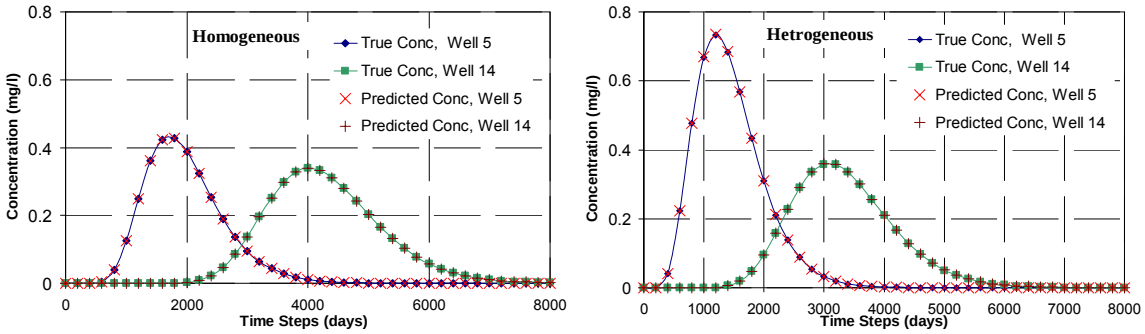

Figure 2-12. True and Predicted Concentration Profiles at Wells 5 and 14 for the Homogeneous and Heterogeneous Case for Scenario 3D-3

The typical convergence behavior of ES in these different scenarios/cases is summarized in Figure 2-13. The results from the 30 random trials, each for the homogeneous and heterogeneous cases, conducted for scenario 3D-2 is used here as a sample for illustrative purposes. This graph shows the variation of the best and the worst objective function values (i.e., prediction errors) obtained at each generation over the 30 trials. This implies that all solutions, among the 30 random trials, at a particular generation number would have had objective function values within the best and worst values at that generation number. Also, a plot (not shown) of the variation of the best objective function value with generation number for any one of the random trials falls between the plots for the best and worst values shown in Figure 2-13. The rate of change of the best values indicates the convergence of the search procedure over iterations (i.e., generations in ES). The difference between the best and the worst values at each generation number indicates the degree of robustness of the search method; the smaller the difference, higher the level of robustness. This figure indicates that the prediction error rapidly improves until around generation number 50, and shows only marginal improvements thereafter, which is characteristic of most evolutionary algorithms. Although the ES-based search procedure was run for 100 generations, the best solutions are found in approximately 50 generations.

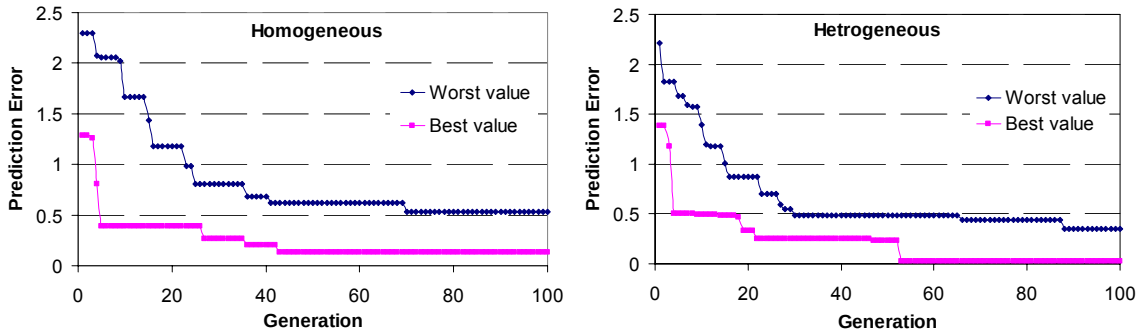

Figure 2-13. Variation of the Best and the Worst Values for Prediction Error over the 30 Random Trials for the Homogeneous and Heterogeneous Cases Corresponding to Scenario 3D-2

2.5.2 Computational Performance Results

For this comparison, the goal is to measure the overall timing of the simulation-optimization approach for solving the application problem. Different sets of runs were conducted to investigate the performance in terms of fine grained parallelism, semi-coarse grained parallelism and coarse grained parallelism. Since the objective is to evaluate the computational performance, one instance, namely scenario 3D-3 for the heterogeneous case, of the CSI problem was selected for illustrative purposes. A population size of 128 was used for the ES-based procedure, which was executed for 10 generations, to estimate the computing time.

The first set of runs was used to investigate fine grained parallelism within the FEM simulator. Time was determined based on the wall clock time corresponding to increasing number of processors per group while the number of groups and number of tasks remained constant. Figure 2-14 illustrates that evaluation time decreases until four processors per group, and then increases thereafter. Improvement in fine grained parallelism is associated with the problem size, and since the size of the problem used in this study is relatively small,

the improvement is limited. Using eight processors or more for such a small problem increases the wall time as a result of excessive communication between processors.

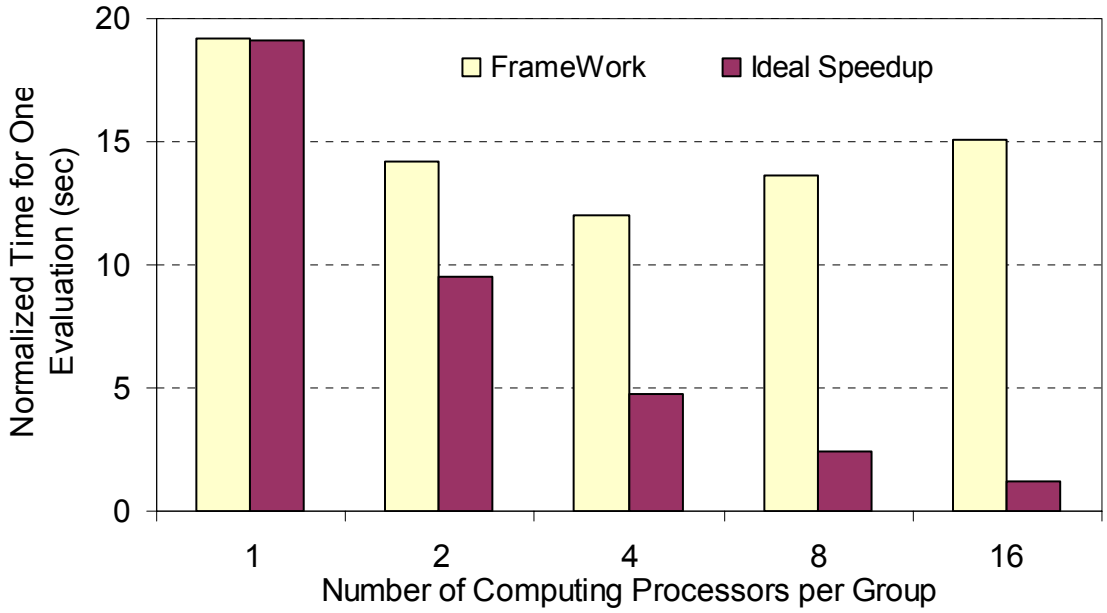

Figure 2-14. Computation Time Comparison due to Different Degree of Fine Grained Parallelism (Number of Groups =1, Task/Group=1:1)

The second set of runs was used to investigate the timing issues associated with semi-coarse grained parallelism. Here the wall time was determined when the number of groups increases. Other parameters such as number of evaluations, number of processors per group and number of tasks per group were kept unchanged. Theoretically, the computation time should scale linearly with the number of processors used. Speed-up results shown in Figure 2-15 indicate that the framework scales almost identical to the theoretical scale until 32 groups, and then scales slightly sub-linearly when the number of group exceeds 64. This could be due to the effect of communication time that plays an important role in increasing the wall time when more than 32 groups are used.

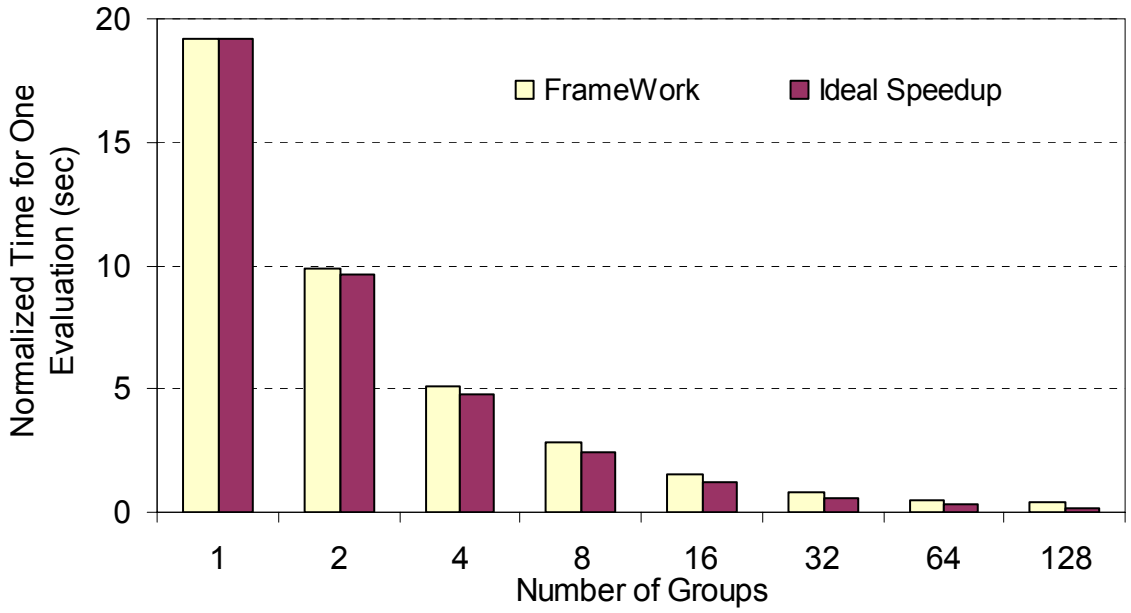

Figure 2-15. Computation Time Comparison due to Different Degree of Semi-Coarse Grained Parallelism (Procs/Group =1:1, Tasks/Group= 1:1)

The third set of runs was used to investigate the computational time associated with coarse grained parallelism. Here the computation time was determined when the number of servers increases. Other parameters such as number of evaluations, number of processors per group, number of groups per server and number of tasks per group were kept unchanged. Theoretically, the computation time should scale linearly with the number of servers used. Speed-up results shown in Figure 2-16 indicate that the framework scales almost identical to the theoretical scale until 16 servers, and then scales slightly sub-linearly when the number of group exceeds 32. This could be due to the effect of communication that plays an important role in increasing the wall time when more than 32 servers are used.

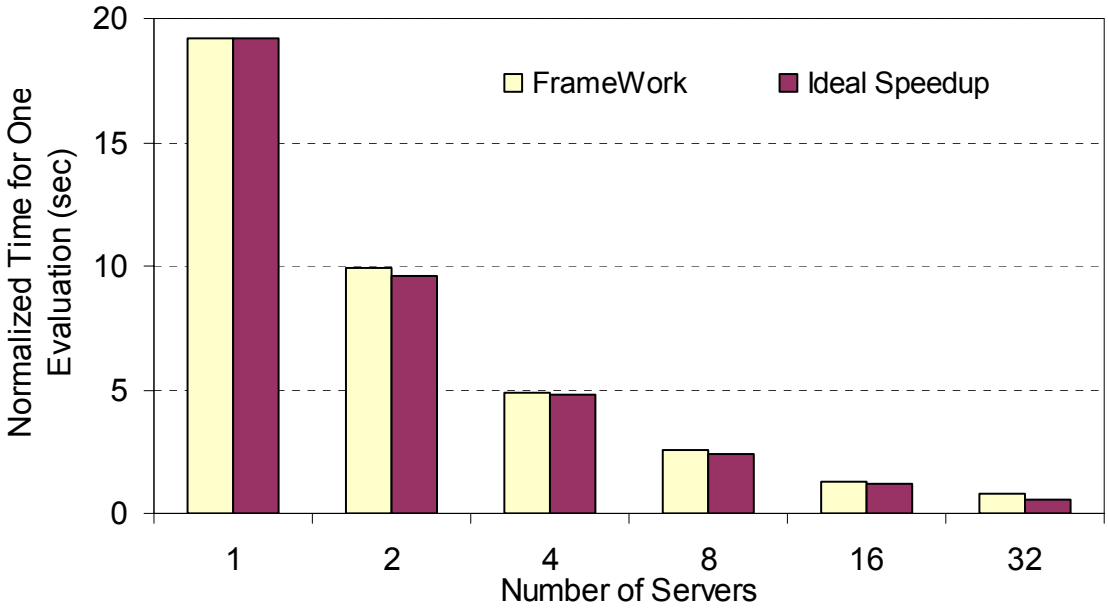

Figure 2-16. Computation Time Comparison due to Different Degree of Coarse Grained Parallelism (Groups/Server = 1:1, Tasks/Group = 1:1, Population size = 128, Procs/Group = 1:1)

Table 2-8. Computation Resources Utilized for All Scenarios

Parameters	Values
Number of Clients (Optimizer)	1
Number of Servers (Coarse Grained)	4
Number of Groups (Semi-Coarse Grained)	32
Number of Processors per Group (Fine Grained)	1
Total Number of Nodes (Processors)	73 (146)

The computational resources settings used to conduct the CSI problem were based on the framework performance results described above and availability of resources at TeraGrid NCSA (i.e., amount of resource that is easy to access without considerably long wait time (matter of hours or days) in the scheduler queue). For the problem scenarios considered in this paper, the number of forward model evaluations performed during a search is a function of the population size multiplied by the number of generations. The total number of evaluations was 5,000, 10,000, and 40,000 for scenario 3D-1, 3D-2, and 3D-3, respectively.

For each scenario, these evaluations were distributed among four servers, each utilizing a 32 processor parallel forward simulation model groups with a single processor per group. Thus, each study utilized a total of 73 computing nodes, i.e., a single node is assigned for the optimization engine, four nodes for the servers, and 68 nodes (134 processors) for the groups with masters. Table 2-8 summarizes the allocation of computation resources, and Table 2-9 shows the total and the normalized computation time utilizing the framework in comparison to the theoretical total and the normalized computation time using a single node for each study. In addition, a speedup analysis is shown for each study. For instance, in the 3D-1 heterogeneous case, an evaluation using the framework took approximately 0.1867 second compared to 19.19 second on a single processor, i.e., solving the problem took approximately 125 minutes (7,467 seconds) instead of approximately nine days (767,600 seconds). The speedup results show that the performance of the framework for all studies are approximately 100 times faster compared to the single processor results.

2.6 Final Remarks

In this paper we have addressed an instance of a groundwater inverse problem using a parallel simulation-optimization framework implemented on the TeraGrid site at NCSA, where the solution procedure employs an ES-based search procedure. A set of scenarios were introduced to investigate the groundwater contaminant source identification problem. The results indicate that the solution for the problem performs adequately for all scenarios, especially for scenarios 3D-1 and 3D-2. In addition, the results show that the problem scenarios perform slightly better for heterogeneous cases than for homogeneous cases. That could be because the effect of non-uniqueness is more pronounced and significant in the complex scenario (3D-3) than in simple ones (3D-1 and 3D-2), and in homogeneous cases than in heterogeneous ones. Additionally, the decision space for the 3D-3 problem is more

challenging and contains multiple local minima, which negatively affects the performance of the search algorithms. The parallelized framework along three levels, coarse and semi-coarse grained at the search algorithm level and fine grained at the forward model level, reduced the evaluation time drastically, to minutes instead of days. For future work, different solution procedures would be attempted for better solution performance such as modeling to generate alternatives and hybrid algorithms for the 3D-3 scenarios. Also the potential sources zone could be constrained further to simplify the search and to reduce the non-uniqueness issue.

Table 2-9. Evaluation Time for Each Scenario

Scenario (No. of Evaluations)	Executing Time Using the Framework Setting (sec)		Executing Time Using a Single Node (sec)		Speed up
	Total	Normalized	Theoretical Total	Normalized	
3D-1 Homogeneous (5,000)	340.45	0.0681	38,850	7.77	114.11
3D-2 Homogeneous (10,000)	7,048.25	0.0705	777,000	7.77	110.24
3D-3 Homogeneous (40,000)	3,006.44	0.0752	310,800	7.77	103.38
3D-1 Heterogeneous (5,000)	849.53	0.1699	95,950	19.9	112.94
3D-2 Heterogeneous (10,000)	1,744.41	0.1744	191,900	19.9	110.01
3D-3 Heterogeneous (40,000)	7,466.87	0.1867	767,600	19.9	102.80

CHAPTER 3

ENHANCED SIMULATION-OPTIMIZATION APPROACH USING SURROGATE MODELING FOR SOLVING INVERSE PROBLEMS

Abstract

This paper investigates and discusses groundwater system characterization problem utilizing surrogate modeling. In this inverse problem, the contaminant signals at monitoring wells are recorded to recreate the pollution profiles. In this study, simulation-optimization approach is a technique utilized to solve inverse problems by formulating them as an optimization model, where evolutionary computation algorithms are used to perform the search. In this approach, the PDE groundwater transport simulation model is solved iteratively during the evolutionary search, which in general can be computationally expensive since thousands of simulation model evaluations will be evaluated. To overcome this limitation, the simulation model is replaced by a surrogate model which is computationally much faster than the simulation model and yet is relatively accurate. Artificial Neural Networks (ANN) is used to construct surrogate models that provide acceptable accuracy performances. The ANN surrogate model, which replaces the PDE groundwater transport simulation model, is then coupled with an evolutionary computation search procedure to solve the source identification problem. In this study, two and three-dimensional problem settings with several experiments scenarios are investigated. The results will present the quality solution of the ANN surrogate model versus the groundwater simulation model, the solution of the inverse problem for different experiment scenarios and finally a timing study analysis conducted to measure the surrogate model performance.

3.1 Introduction

Groundwater system characterization is enormously important in real-time forecasting of groundwater pollutant plumes for making effective decisions to protect environmental resources, conduct liability assessment, estimate risk and design disaster mitigation. The characterization of a source of groundwater pollution from sparse observational data collected at measurement stations is generally classified as an inverse problem. Solving inverse problems is relatively complex due to the ill-posed nature of the problem. Solution of an inverse problem may be limited by non-uniqueness, where more than one solution may exist; instability, where the solutions may be overly sensitive to observation errors; or non-existence, where a solution may not exist (Sun, 1994; Atmadja and Bagtzoglou, 2001).

Among several available methods, the simulation-optimization approach is a technique utilized to solve inverse problems through the coupling of a simulation model with an optimization model (Atmadja and Bagtzoglou, 2001). The inverse problem is formulated and solved as an optimization model. Simulation models, usually referred to as forward models, describe the governing processes of a system and define the relationship between system inputs and outputs. A simulation-optimization approach uses search algorithms to identify the best system characteristics that minimize the error between the model predictions and the system observations. While this approach is generic and robust, it is computationally expensive because the optimization model uses an iterative process and the corresponding simulation model is typically a complex system of processes requiring a relatively long time to solve.

Surrogate modeling is being investigated to address the excessive computational cost resulting from iterative evaluation of the simulation model. Efficiency of the search could be improved by constructing a surrogate model that is able to estimate quickly the simulation model output (Ong et al., 2003). As the level of complexity of a simulation model increases, the effort and resources needed to construct and train a surrogate model increase as well. This increase, however, occurs prior to the iterative search procedure and does not contribute to the computational burden during the simulation-optimization procedure.

The main objective of this research is to construct surrogate models that ease the computational overhead of the simulation-optimization approach to address an illustrative instance of a groundwater system characterization problem. As previous studies (e.g., Sun, 1994) reported that identifying a unique solution for groundwater system characterization problems can be challenging, a second objective of this research is to investigate the degrees of non-uniqueness in such problem for scenarios with increasing complexity. In this article, artificial neural networks (ANN) are used for surrogate modeling of two and three-dimensional numerical simulation groundwater transport and remediation models. While using ANN applications in predicting and forecasting of water resources variability is increasingly popular (Ranjithan et al., 1993), its applications in groundwater and groundwater inverse problems are limited. Examples of such applications and reviews are as follows: The inverse problem of parameter estimation in groundwater models is addressed by means of an ANN (Zio, 1997). Sun (1997) effectively applied an ANN and a genetic algorithm to identify distributed parameter zones for two-dimensional groundwater problems. Simunek et al. (1998) used an ANN to estimate the soil hydraulic characteristics of a two layered soil system through an inverse procedure. Rajanayaka et al. (2002) successfully used

an ANN-based approach to estimate hydraulic conductivity and longitudinal dispersion coefficient in stochastic groundwater modeling. Shigidi and Garcia (2003) used ANN approaches to estimate the hydraulic head values in groundwater domains. Singh et al. (2004) presented some preliminary results for identification of unknown pollution sources using an ANN.

The groundwater contaminant source identification problem (CSI) is utilized in this study. The application problem and the solution approach (ANN-GA) used in this paper resembles the work by Datta and Chakrabarty (2003) in which contaminant source identification problem is solved using an ANN-GA approach. While the previous study modeled a two-dimensional groundwater fields, this work tests both two and three-dimensional heterogeneous flow field problems with increasing problem complexity. Additionally, previous work considered only a few sites as potential source locations, and this study allows 80% of the domain as potential source locations. This study demonstrates the accuracy of the ANN surrogate model in predicting pollutant concentration profiles compared with those produced from the original groundwater transport simulation model. The results also illustrate the quality solution of the inverse problem under alternate scenarios. Finally, a comparative timing study analysis will be demonstrated using the surrogate model versus the simulation model.

3.2 Problem Complexity and Solution Approach

3.2.1 Problem Description

The groundwater contaminant source identification problem can be described as an unknown contaminant release at a single source location (or multiple source locations) that is

resolved from spatially and temporally distributed concentration observations collected at monitoring wells (Fig. 3-1). The contaminant source locations and the contaminant release at the sources are unknown. Concentration observations at monitoring wells are collected to generate a concentration time-series at each monitoring location. Furthermore, the signature of the source embedded in the monitoring data is a function of the source characteristics.

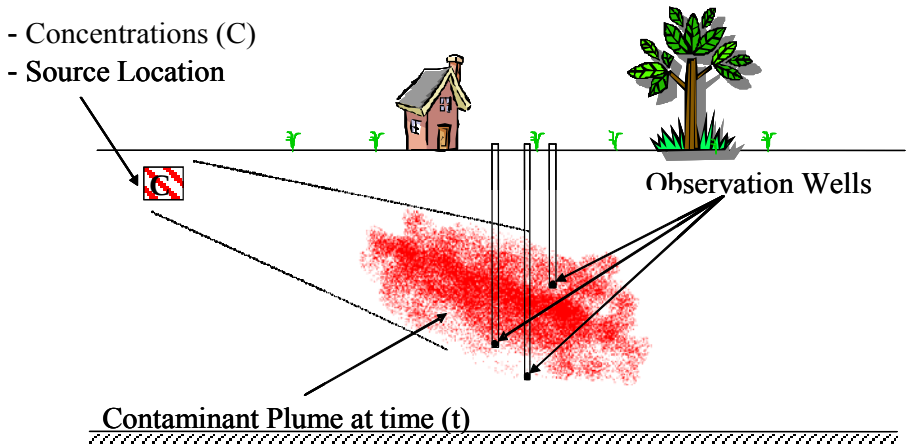

Figure 3-1. A Schematic Explaining the Contaminant Source Identification Problem

The inverse problem can be posed as a constrained optimization model where the error between the observed and simulated concentrations is minimized. Several error objective functions can be used to minimize the error; for example Equation 3-1 shows a normalized root square error used to minimize the objective function.

$$\underset{(Decision\ Variables)}{\text{minimize}} \sqrt{\frac{\sum_{w=1}^{nw}\sum_{i=1}^{n}\left(C_{iw}^{o} - C_{iw}^{m}\right)^2}{\sum_{w=1}^{nw}\sum_{i=1}^{n}\left(C_{iw}^{o}\right)^2}} \qquad (3\text{-}1)$$

where $\mathbf{C}_{iw}^{o}$ is the observed concentration generated using the groundwater forward model for n readings at discrete time steps recorded at (nw) monitoring wells, $\mathbf{C}_{iw}^{m}$ is the modeled

concentrations generated using decision variables as inputs for the groundwater forward model for *n* readings at discrete time steps modeled at (*nw*) monitoring wells, and *Decision Variables* are the contaminant source location(s) and contaminant concentration(s).

3.2.2 Simulation-Optimization Methodologies

Several solution methods for inverse problems, including simulation-optimization techniques, probabilistic procedures, analytical methods, and direct inversion approaches are available (Johnson and Rogers, 2000). This research explores the simulation-optimization approach. This is a general term used to describe optimization techniques that utilize simulation models for the evaluation of objective and constraint functions. In this approach, the simulation model is coupled with optimization techniques to determine the model inputs that best approximate the observed data. This process is applied iteratively until certain stopping criteria are met.

A simulation-optimization approach to solve the groundwater inverse problem is several orders of magnitude more computationally challenging than the solution of the corresponding groundwater transport forward model, since several thousands of forward model evaluations, requiring solution to a system of PDEs, are required, (Andradóttir, 1998). Given the potential computational intractability of such a simulation-optimization approach, improving the efficiency of simulation model is required in this research.

A typical groundwater simulation model is used as an illustration in solving a groundwater inverse problem. PGREM3D, Parallel Groundwater transport and REMediation codes, a suite of codes used for numerical simulation of groundwater transport and

remediation problems. The code is based on the finite element methods and a detailed description of the code is available (Mahinthakumar, 1999).

While several different search procedures can be explored to conduct the optimization search, the emphasis of this study is on a Genetic Algorithm-based heuristic search approaches. The Genetic Algorithm (GA) was developed in the 1960s by John Holland (Holland, 1975) and has been used to solve an array of optimization problems. Although a GA-based approach has the potential to find globally optimal solutions, it has been known as a computationally demanding approach (Goldberg, 1989; Davis, 1991).

Several approaches could be attempted to decrease the computational overhead that exists in the simulation-optimization approach. Surrogate modeling is explored to improve the simulation model efficiency and reduce the solution time. Surrogate models are built to approximate computationally expensive simulation codes, and surrogate models are orders of magnitude cheaper to execute. Several surrogate methods, such as genetic programming (GP), artificial neural network (ANN), response surfaces method (RSM) and statistical regression methods, are available to replace a simulation model (Fig. 3-2). In this study, ANN surrogate models are coupled with the GA-based optimization techniques (GAs) to improve the efficiency of the simulation model by expediting the execution. The ANN-GA is similar to the procedure used by Morshed and Kaluarachchi (1998) to estimate groundwater hydraulic conductivity and by Tsai et al. (2003) to identify parameter structure in groundwater modeling.

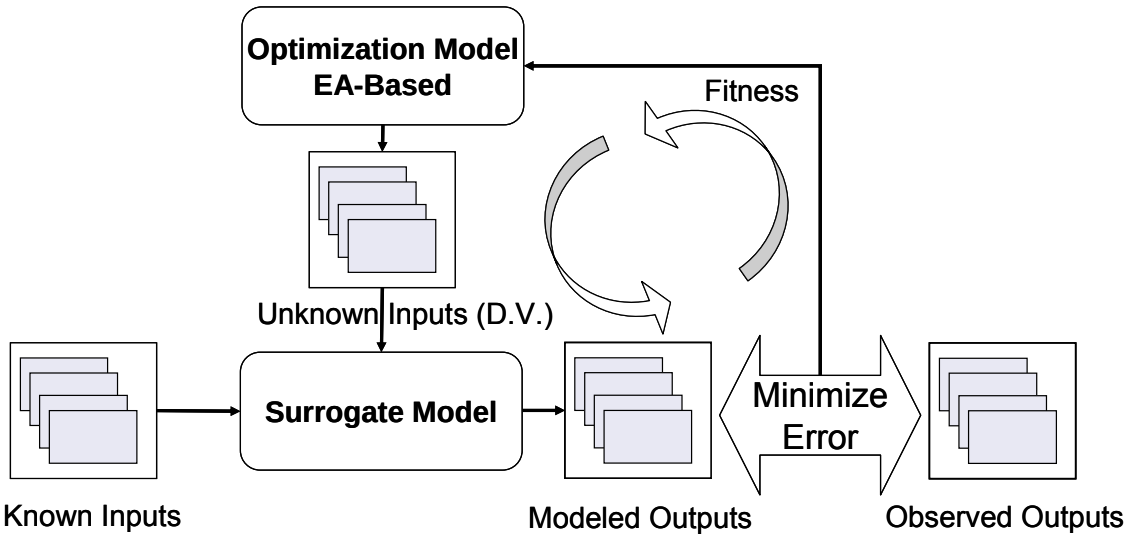

Figure 3-2. Simulation-Optimization Approach Utilizing Surrogate Models

3.3 Surrogate Modeling -- Artificial Neural Network

ANNs mimic human brains to learn the relationship between certain inputs and outputs from experience, and do not require specific knowledge of the fundamental processes of the system. ANNs may be defined as universal function estimators, which are capable of finding relationships between input and output system parameters through the work of trained neurons (Bishop, 1995). A general schematic of a three-layer feed-forward backpropagation ANN model consists of an input layer, a hidden layer and an output layer (Figure 3-3). The input layer consists of a number of processing elements equal to the number of input parameters, and the output layer consists of a number of processing elements equal to the number of output parameters. An artificial neuron or processing elements consists of three components: input links, a processing unit and an output link. The processing unit usually consists of two functions and a threshold. The threshold is a negative resident weight of the neuron (θ). The two functions are a summation function (see Eqn. 3-2) and an activation function (see Eqn. 3-3). The summation function sums up the inputs (w_{ji}, x_i)

and the threshold (θ_j), and the sum is fed into the activation function (f) to produce an output stimulus. More detailed descriptions of ANNs are given in many references (e.g., Flood and Kartam, 1994).

$$I_j = \sum w_{ji} x_i + \theta_j \tag{3-2}$$

$$y_j = f(I_j) \tag{3-3}$$

Previous studies (e.g., El-Din et al., 2004) reported that ANNs avoid some drawbacks of the other surrogate modeling techniques. The ANN is based on learning from historical data without comprehensive understanding of the model's mathematical and physical processes. Therefore ANN does not require a developed model or human expert to construct a surrogate model. The resulting ANN models are able to mimic both linear and non-linear systems and usually are robust against process noise or instrumentation bias.

The general procedure used to develop backpropagation ANN models consists of the following steps.

Step 1. Producing data: Generate data using the simulation model, including inputs and outputs data.

Step 2. Selecting input parameters: Select input variables based on engineering/scientific decision variables of the problem scenarios.

Step 3. Choosing ANN architecture and internal parameters: Determine number of nodes, hidden layers, connection weights, and layers size.

Step 4. Training the ANN: The known input parameter values and the corresponding known output parameter values are used to train the ANN. By observing a large enough number of

patterns, the ANN is able to capture the relationship between the input variables and the output variable. This training process enables the ANN to find a set of connection weights that will produce the best possible input/output mapping to predict output variable values when only given the input variable values.

Step 5. Evaluating ANN model performance: The performance of the trained ANN models is assessed by comparing the predicted output signal produced by the ANN model with the observed output values. An error function can be used to evaluate the ANN model performance.

Step 6. Final ANN model: Steps 4 and 5 are repeated until an acceptable error is met. ANN model development is an iterative process involving training and performance evaluation until a stopping criterion is achieved. The final prediction output can usually be condensed into a simple mathematical formula.

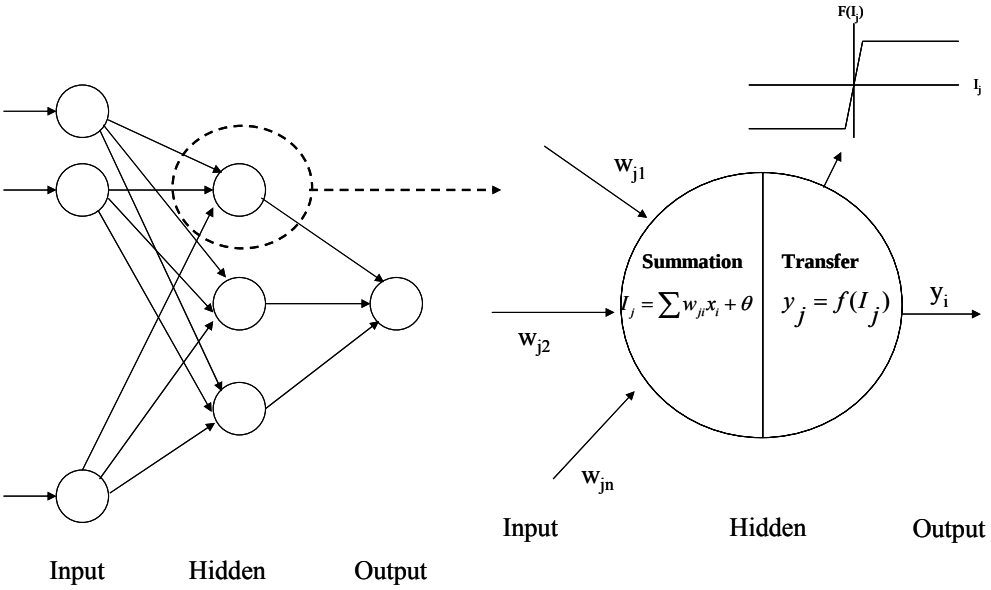

Figure 3-3. Overview of a Typical Feed-Forward Backpropagation ANN

3.4 Illustrative Application

To demonstrate the applicability of the solution approach for a contaminant source identification problem, a number of scenarios with different degrees of complexity were developed. Synthetic data were generated using the groundwater simulation model (PGREM3D), and a description of the groundwater fields and the simulation model parameters are given below, including a description of the ANN architecture used to construct the application scenarios and the optimization formulations for each application scenario.

3.4.1 Application Scenarios

In this study, ANN surrogate models were constructed for both two and three-dimensional groundwater fields, and three scenarios of varying levels of complexity were introduced for each field (a total of six scenarios). The scenarios differ in the number of unknown parameters used to characterize the contaminant source. In simple scenarios, few parameters are used to characterize the source, where parameters that are not estimated are assumed. The assumptions are made as realistic as possible based on the size of the groundwater domain and practical experiences.

2D-1, in a two-dimensional field the contaminant source is characterized by the source centroid and contaminant concentration. Three parameters are estimated: the contaminant source centroid (x_c, y_c) and the initial concentration (C_0). In this scenario the source size is assumed (or known).

2D-2, in a two-dimensional field the contaminant source is described by the location, concentration and size assuming the source is square (i.e., the source lengths L_1, L_2 are

equal). Four parameters are estimated: contaminant source centroid (x_c, y_c), contaminant source size (s) and the contaminant initial concentration (C_0).

2D-3, in a two-dimensional field the contaminant source is characterized using contaminant source location, concentration and size of each side. Five parameters are estimated: contaminant source centroid (x_c, y_c), contaminant source size (s_x, s_y) and the contaminant initial concentration (C_0).

3D-1, in a three-dimensional field the contaminant source is predicted using centroid and contaminant initial concentration. Four parameters are estimated: the contaminant source centroid (x_c, y_c, z_c) and the initial concentration (C_0). In this scenario the source size is assumed.

3D-2, in a three-dimensional field the contaminant source is characterized using the contaminant source location, concentration and size assuming the source has a cubical geometry (i.e., the source lengths L_1, L_2, L_3 are equal). Five parameters are estimated: contaminant source centroid (x_c, y_c, z_c), contaminant source size (s) and the contaminant initial concentration (C_0).

3D-3, in a three-dimensional field the contaminant source is characterized using the contaminant source location, concentration and size assuming the source has a cuboidal geometry (i.e., the source lengths L_1, L_2 ,L_3 can be unequal). Seven parameters are estimated: contaminant source centroid (x_c, y_c, z_c), contaminant source size (s_x, s_y, s_z) and the contaminant initial concentration (C_0).

Table 3-1. Hypothetical Domain Parameters

Parameter	Values for 2D problem	Values for 3D problem
Problem Size	51x31 grid	51x31x11 grid
Number of Time Steps	20	20
Time Step Size (dt)	100 day	100 day
Grid Spacing (dx, dy, dz)	10m x 10m	10m x 10m x 1m
Dispersion Parameters (α_L, α_{TH}, α_{TV}, D_m)	10m, 5m, 1m, 0.001m^2/day	10m, 5m, 1m, 0.001m^2/day
Flow Field	Heterogeneous, Isotropic	Heterogeneous, Isotropic
Observation Wells	5	18

3.4.2 Simulation Model Settings

In this study, hypothetical heterogeneous two and three-dimensional fields were considered. Detailed geometrical and hydraulic parameters are shown in Tables 3-1 and 3-2, and Figures 3-4 and 3-5. These parameters were used in the groundwater simulation model to generate data used to train the ANN surrogate models. The flow fields for both two and three–dimensional scenarios are heterogeneous; the steady state flow is computed using a randomly heterogeneous hydraulic conductivity field that was generated using a turning band code (Thompson et al., 1989). The simulations, for all problems, are performed for 1000 time steps or 2000 days. For the two-dimensional applications, the grid resolution for the simulation model resulted in 1,581 (51x31) finite element nodes within the groundwater domain, the simulation duration was 20 time steps, and a total of five monitoring wells were located close to the downstream end of the field (Fig. 3-4). For the three-dimensional applications, the grid resolution for the simulation model resulted in 17,391 (51x31x11) finite element nodes within the groundwater domain, the simulation duration was 20 time steps,

and a total of 18 monitoring wells were located in the middle and downstream end of the field (Fig. 3-5).

Table 3-2. True Source Locations for 2D and 3D Scenarios

2D Scenarios	True Source Parameters				
	x_c=8	y_c=10	s_x=2	s_y=2	C_0=70
2D-1	Est.	Est.	Assumed (square), $s_x = s_y$		Est.
2D-2	Est.	Est.	Est. (square), $s_x = s_y$		Est.
2D-3	Est.	Est.	Est. (square or rectangular), s_x, s_y may not be equal		Est.

3D Scenarios	True Source Parameters						
	x_c=8	y_c=10	z_c=4	s_x=2	s_y=2	s_z=2	C_0=70
3D-1	Est.	Est.	Est.	Assumed (cubical), $s_x = s_y = s_z$			Est.
3D-2	Est.	Est.	Est.	Est. (cubical), $s_x = s_y = s_z$			Est.
3D-3	Est.	Est.	Est.	Est. (cuboidal), s_x, s_y, s_z may not be equal			Est.

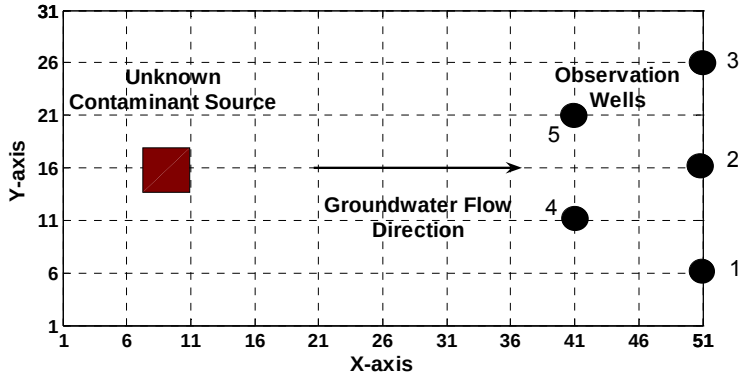

Figure 3-4. Hypothetical Two-Dimensional Domain, Axes Units are Grid Node

3.4.3 ANN Surrogate Model Settings Used to Develop the Applications Scenarios

Training an ANN model uses an optimization procedure in which the model parameters such as connection weights and bias values are adjusted to minimize the error between the outputs predicted by the model and the actual outputs. An ANN surrogate model is constructed for each experiment scenario, for a total of six surrogate models. All ANN

models were developed using the MATLAB Neural Network Toolbox (Demuth and Beale, 2005; MATLAB, 2006). The procedure used to develop the ANN models follows the steps explained in Section 3.3 and is customized for the groundwater problem scenarios.

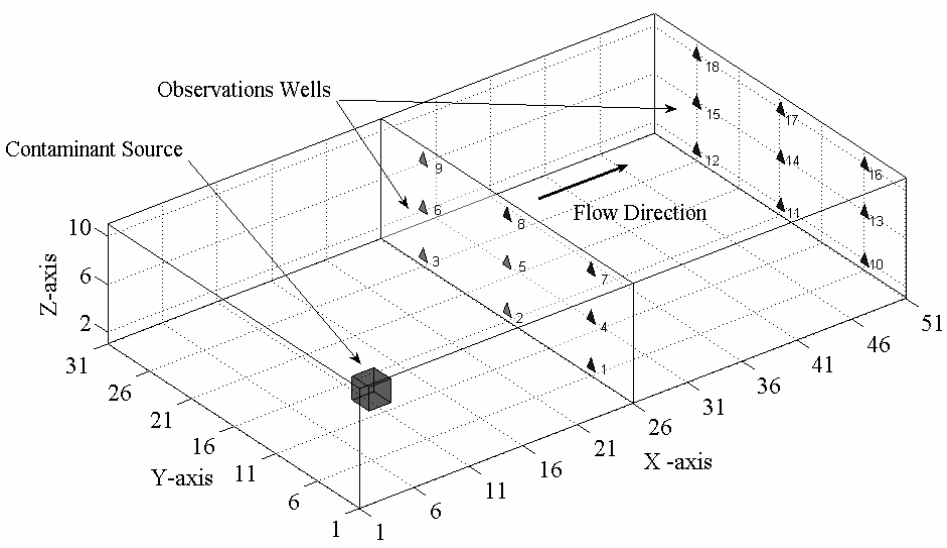

Figure 3-5. Hypothetical Three-Dimensional Domain, Axes Units are Grid Node

Table 3-3. Number and Sizes of Potential Sources for Each Scenario

Scenario	Parameters Need to be Estimated	Size of Potential Source (x, y) Grid Unit	No. of Data Points Used to Train ANN
2D-1	x_c, y_c, C_0	1x1	1,200
2D-2	x_c, y_c, s, C_0	1x1, 2x2, 3x3, 4x4	1,700
2D-3	x_c, y_c, s_x, s_y, C_0	1x1, 1x2, 1x3, 1x4, 2x1, 2x2, 2x3, 2x4 3x1, 3x2, 3x3, 3x4, 4x1, 4x2, 4x3, 4x4	5,146
Scenario	**Parameters Need to be Estimated**	**Size of Potential Source (x, y, z) Grid Unit**	**No. of Data Points Used to Train ANN**
3D-1	x_c, y_c, z_c, C_0	2x2x2	1,500
3D-2	x_c, y_c, z_c, s, C_0	1x1x1, 2x2x2, 3x3x3, 4x4x4	14,030
3D-3	x_c,y_c,z_c,s_x,s_y,s_z,C_0	1x1x1, 1x1x2, 1x1x3, 1x1x4,…,1x4x4 2x1x1, 2x1x2, 2x1x3, 2x1x4,…,2x4x4 3x1x1, 3x1x2, 3x1x3, 3x1x4,…,3x4x4 4x1x1, 4x1x2, 4x1x3, 4x1x4,…,4x4x4	102,920

3.4.3.1 Data

The two and three-dimensional groundwater transport models are utilized to produce concentration profiles for potential contaminant sources at five and eighteen monitoring wells respectively (Table 3-3). Each concentration profile consists of 20 time steps. Eighty percent of the domain was used to generate the potential sources for both the 2D and 3D domain. The potential sources area is considered from the upstream of the domain to 80% of the x direction, thus the potential sources domains are 1200 m^2 (400m x 300m) and 12000 m^3 (400m x 300m x 10m) for 2D problem and 3D problem, respectively.

3.4.3.2 Model Structures

In this study, a three-layer backpropagation ANN model training function and weigh/bias learning function were used. In each scenario the set of unknowns (or parameters to be estimated) were used as input parameters, except for the contaminant concentration parameter (C_0) since it has a linear response. Other internal parameters were determined by a trial-and-error method. Table 3-4 shows the internal parameters used in this study.

3.4.3.3 Training

The purpose of the model is to capture adequately the underlying relationship between input parameters (unknown parameters) and the output parameters (concentration profile). The training process identifies a set of neural network vectors (weights and bias values) that produces the best possible input/output mapping (unknown parameters/concentration profiles) mapping. An ANN model is constructed for each time step for each observation well; therefore each scenario consists of 20 ANN models for each

well. For each 3D scenario, a total of 360 (20 time steps, 18 wells) ANN models were constructed, while for each 2D scenario, a total of 100 (20 time steps, 5 wells) ANN models were constructed.

3.4.3.4 Validation

The performance of the trained ANN models was assessed by comparing the predicted simulated concentration profiles produced by the ANN model with the actual concentration profile produced by the simulation model. The prediction performance and generalization ability of ANN models are evaluated using a maximum absolute error function shown below:

$$Error = \max(|Pr_i - Ob_i|) \quad (3\text{-}4)$$

where Pr_i is the predicted output by the trained ANN model of the i^{th} sample and Ob_i is the observed output value.

The training and the evaluation process will be repeated until the performance goal is met or total number of epochs is reached, as specified in Table 3-3. The final formula of a surrogate model is shown below:

$$C(t) = (LW_{3,2} \times (\exp\ (-(LW_{2,1} \times (\exp\ (-(IW_1 \times IP + b_1)^2)) + b_2)^2)) + b_3) \quad (3\text{-}5)$$

where $C(t)$ is the concentration at time step (t), b_L is a bias vector ($S_L, 1$) where (S_L) is number of neurons in layer (L), IP is the input vector ($R, 1$) where (R) is the number of element in the input vector, $\boldsymbol{LW_{L, L-1}}$ is the layer weight matrix (S_L, S_{L-1}) and $\boldsymbol{IW_L}$ is the input weight matrix (S_L, R).

Table 3-4. ANN Model Architecture and Input/Output Parameters

ANN Parameters	Scenario 2D-1	Scenario 2D-2	Scenario 2D-3
ANN Model	Feed-forward Backpropagation		
Number of Layers	3 Layers		
Number of Neurons per Layer	30, 14, 1	50, 25, 1	50, 25, 1
Epochs	3000		
Transfer Function per Layer	Radial Basis, Radial Basis, Linear Transfer Function		
Network Training Function	Resilient Backpropagation		
Weight/bias Learning Function	Gradient Descent with Momentum, 0.5		
Performance Function	Mean Square Error		
Performance Goal	1.0E-8		
Input Parameters	x_c, y_c	x_c, y_c, s	x_c, y_c, s_x, s_y
Output Parameters	Concentration Profiles (Time step)		
ANN Parameters	Scenario 3D-1	Scenario 3D-2	Scenario 3D-3
ANN Model	Feed-forward Backpropagation		
Number of Layers	3 Layers		
Number of Neurons per Layer	30, 14, 1	50, 25, 1	50, 25, 1
Epochs	3000		
Transfer Function per Layer	Radial Basis, Radial Basis, Linear Transfer Function		
Network Training Function	Resilient Backpropagation		
Weight/bias Learning Function	Gradient Descent with Momentum, 0.5		
Performance Function	Mean Square Error		
Performance Goal	1.0E-8		
Input Parameters	x_c, y_c, z_c	x_c, y_c, z_c, s	x_c, y_c, z_c, s_x, s_y, s_z
Output Parameters	Concentration Profiles (Time step)		

3.4.4 Optimization Models for the Application Scenarios

The groundwater inverse problem is posed as an optimization model where the normalized root square error (RSE) between the observed and modeled concentrations is minimized. Equation 3-6 shows the objective function for each scenario respectively. The following constraints, shown in Equations (3-7)-(3-9), are included to enforce decision variable bounds and feasibility.

For this study, a single contaminant is considered. The number of decision variables is based on the number of parameters used to describe the source.

$$\underset{P_k}{\text{minimize}} \sqrt{\frac{\sum_{w=1}^{nw}\sum_{i=1}^{n}\left(\mathbf{C}_{iw}^{o}-\mathbf{C}_{iw}^{m_1}\right)^2}{\sum_{w=1}^{nw}\sum_{i=1}^{n}\left(\mathbf{C}_{iw}^{o}\right)^2}} \tag{3-6}$$

subject to:

$$0 \le C_0 \le C_{max} \tag{3-7}$$

$$0 \le x_c \le x_{max};\ 0 \le y_c \le y_{max};\ 0 \le z_c \le z_{max} \tag{3-8}$$

$$s_x \le s_{x\_max};\ s_y \le s_{y\_max};\ s_z \le s_{z\_max} \tag{3-9}$$

where P_k is parameters need to be estimated for each scenario (k); as shown in Table 3-2. C_{iw}^{o} is the observed concentration for time step (i) at observation well (w) and $C_{iw}^{m_k}$ is the modeled concentration for scenario (k) for time step (i) at observation well (w) where

$$\boldsymbol{C}^{m1} = f(x_c, y_c, \boldsymbol{C}_s) \quad (3\text{-}10), \quad \boldsymbol{C}^{m4} = f(x_c, y_c, z_c, \boldsymbol{C}_s) \quad (3\text{-}13)$$

$$\boldsymbol{C}^{m2} = f(x_c, y_c, s, \boldsymbol{C}_s) \quad (3\text{-}11), \quad \boldsymbol{C}^{m5} = f(x_c, y_c, z_c, s, \boldsymbol{C}_s) \quad (3\text{-}14)$$

$$\boldsymbol{C}^{m3} = f(x_c, y_c, s_x, s_y, \boldsymbol{C}_s) \quad (3\text{-}12), \quad \boldsymbol{C}^{m6} = f(x_c, y_c, z_c, s_x, s_y, s_z, \boldsymbol{C}_s) \quad (3\text{-}15)$$

x_c, y_c *and* z_c are the centroids of the expected source location; s_x, s_y *and* s_z are the source size in x, y and z direction respectively; $\boldsymbol{C}_s$ is the source concentration; n is the total number of time steps, and nw is the total number of observations wells.

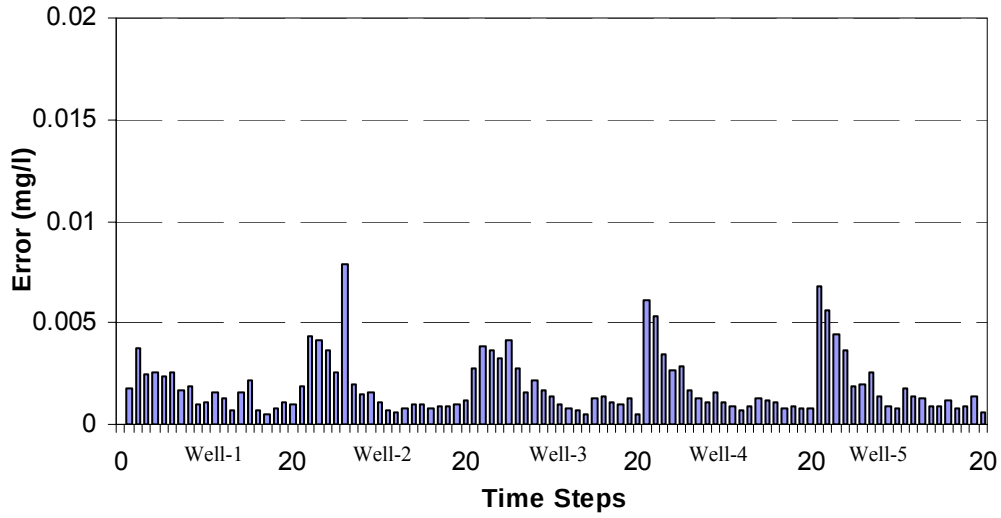

Figure 3-6. Average Error Values of ANN Models for Scenario 2D-2

3.5 Results and Discussion

3.5.1 ANN-based Surrogate Model Solution Quality

The ANN surrogate model solution quality is demonstrated for both 2D and 3D cases by calculating the maximum error obtained using Equation 3-4. The performances of both training and test data are examined; training data refers to the data used to train the ANN surrogate model while test data refers to data that was not included in the training data.

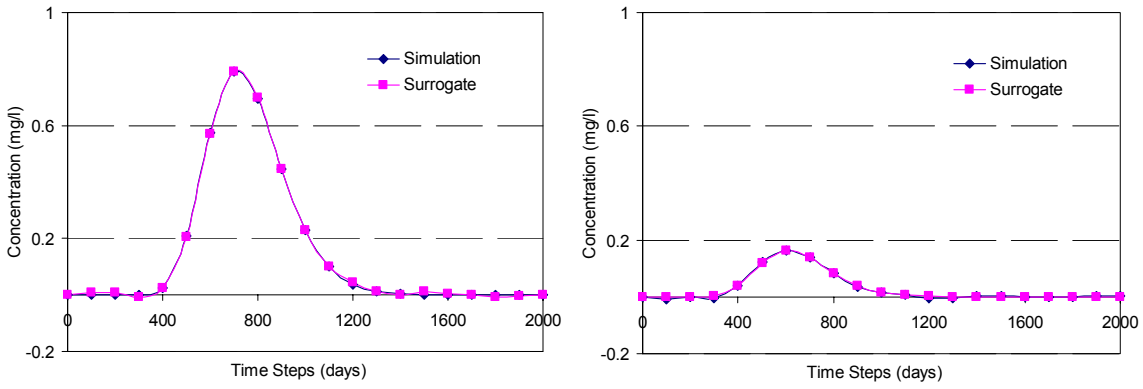

Figure 3-7. Simulation and ANN Concentration Profile for Scenario 2D-2 Using the Training Data, Profiles are for Source (8, 10, 2, 70) at Wells 1 and 5, respectively

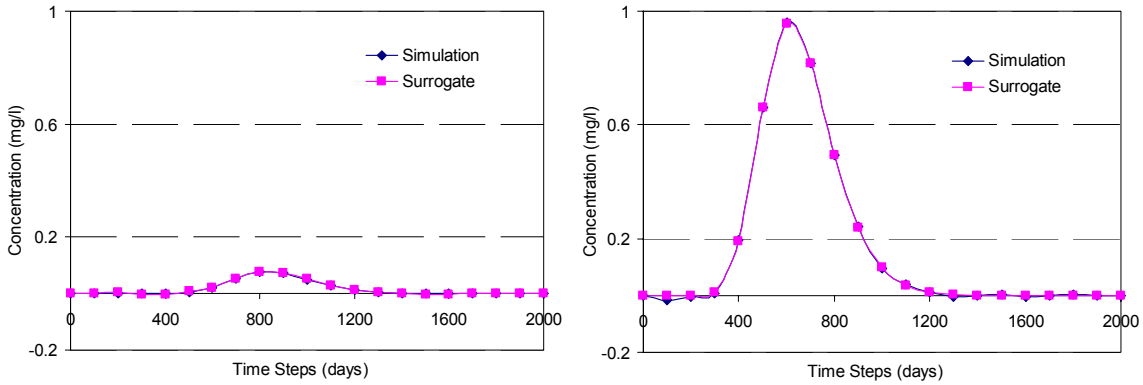

Figure 3-8. Simulation and ANN Concentration Profile for Scenario 2D-2 Using Test Data, Profiles are for Source (5, 21, 2, 70) at Wells 1 and 5, respectively

3.5.1.1 Two-Dimensional Solution Quality

The solution quality of the ANN surrogate model is demonstrated for the 2D scenarios (Scenarios 1-3). For the training data, the error values are relatively small, as example the error for 2D-2 scenario is shown in Figure 3-6. The error values increase for problems of increasing complexity. Additionally, larger error values are found at earlier time steps in the concentration profile at each well. This may be a result of large variations in the concentration profiles at those time steps. Typical examples of concentration profiles for the ANN model versus simulation model are plotted in Figure 3-7 for training data. Another set of examples of concentration profiles for ANN model versus simulation model are plotted in Figure 3-8 using test data. Observation wells 1 and 5 were used to obtain these sets of examples. The ANN models have almost identical concentration profile as those obtained by the simulation model for both test and training data.

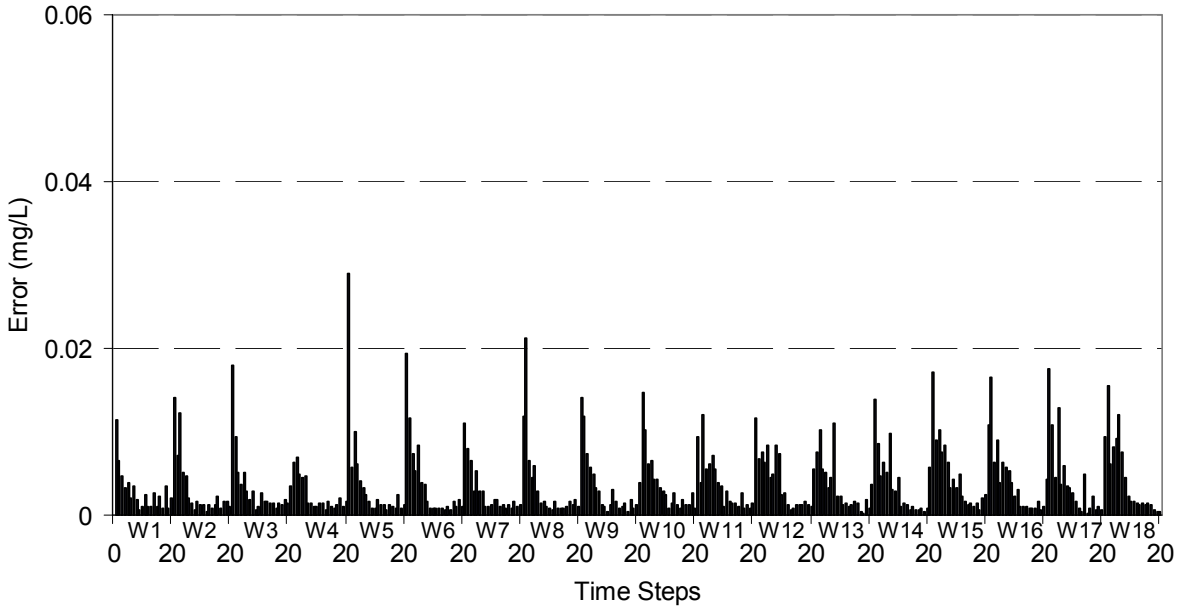

Figure 3-9. Average Error Values of ANN Models for Scenario 3D-2

3.5.1.2 Three-Dimensional Solution Quality

For the 3D problem scenarios (Scenarios 4-6), the ANN model performance is similar to the performance for the 2D scenarios. The average error values are relatively small for the set of 3D ANN models (as example the error for 3D-2 scenarios is shown in Figure 3-9); again, however, the error values increase for more complex problems, and larger error values are associated with early time steps. Additionally, larger errors usually occur from sources close to the domain boundaries. Typical examples of concentration profiles for the ANN model versus simulation model are plotted in Figure 3-10 for training data. A second set of concentration profiles are plotted in Figure 3-11 for test data at observation wells 1 and 11. The ANN models predict almost identical concentration profile to those obtained by the simulation models for both test and training data.

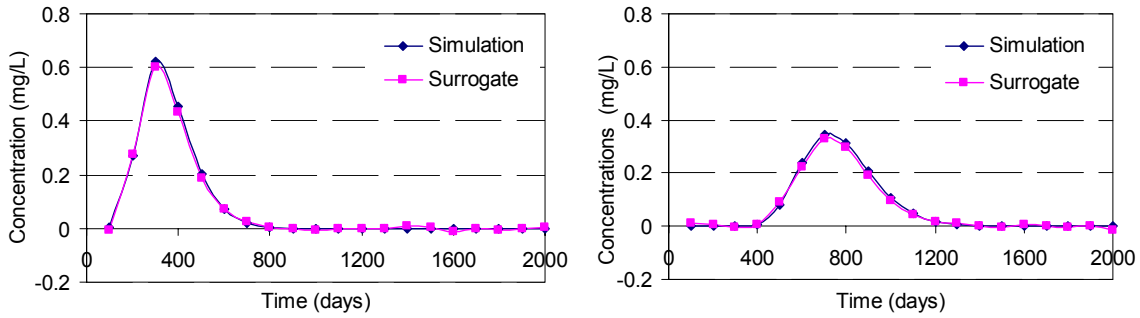

Figure 3-10. Simulation and ANN Concentration Profiles for Scenario 3D-2 Using Training Data, Profiles are for Source (8, 10, 4, 2, 70) at Wells 1 and 11, respectively

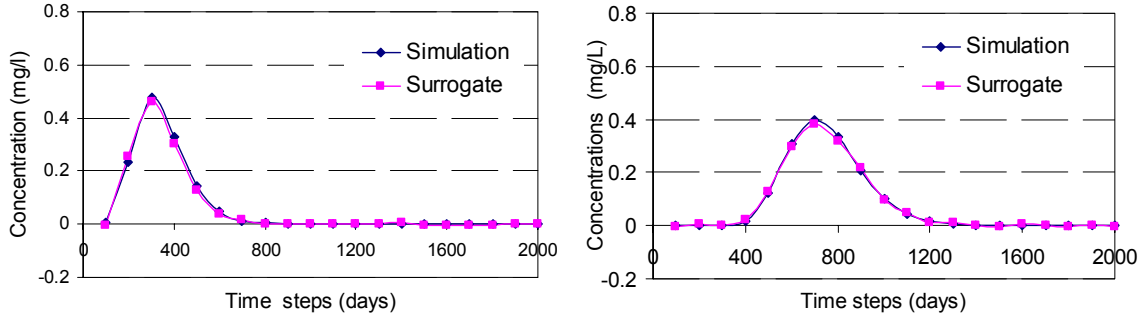

Figure 3-11. Simulation and ANN Concentration Profile for Scenario 3D-2 Using Test Data, Profiles are for Source (9, 11, 5, 2.5, 70) at Wells 1 and 11, respectively

3.5.2 Contaminant Source Identification Problem Solution

The ANN surrogate models were coupled with a standard MATLAB GA to solve the contaminant source identification problem using a set of scenarios. The GA searches for a source characterization that generates a concentration profile (modeled profile) to match a concentration profile (observed profile) that was generated using an ANN forward model. The collected observations can be obtained from up to five wells in the two-dimensional problem scenarios and up to eighteen wells in the three-dimensional problem scenarios. This study assumes that the forward model perfectly generates the concentration (i.e., no measurement error occurs at the observation wells). The objective function (Eqn. 3-6) is used to determine how well the GA identifies the source characterization.

Since the true solutions of these hypothetical CSI problem scenarios are known in advance, the quality of the solutions obtained by the GA can be evaluated based on how they are able to predict the true solution, using a solution error metric, shown in Equation 3-16 below. This metric is not part of the solution process.

$$Solution\ Error\ (\%) = \left(\sum_{j=1}^{np} \left(\frac{\left| p_j^{true} - p_j^{pred} \right|}{p_j^{true}} \right) \right) \times \frac{100}{np} \tag{3-16}$$

where p_j^{true} is the true value of parameter j (i.e., decision variables), p_j^{pred} is the predicted value of parameter j, and np is the number of parameters.

The population size for the MATLAB GA toolbox was set to 200 and the algorithm was executed for 100 generations (MATLAB, 2006). Other GA parameters and the boundary parameters used for this problem are shown in Tables 3-5 and 3-6, respectively. As a GA is a probabilistic method, a set of 30 random trials are conducted to evaluate the robustness of the procedure.

Table 3-5. GA Settings for the 2D and 3D Problems

Parameter	Setting for the MATLAB GA Toolbox
Population Size	200
Decision Variables Type	Real
Generation	100
Selection	Stochastic Uniform
Crossover	Heuristic Crossover, 0.8
Mutation	Gaussian Mutation, 0.2

Table 3-6. Allowable Range of Decision Variables Values for 2D and 3D Problems

Variables	Ranges
x axis	1-40 grid node
y axis	1-30 grid node
z axis	1-10 grid node
C_0	0-100 mg/l

Table 3-7. Best Source Characterization Solution for Scenario 2D-1 Utilizing Collected Observations from One Well

Parameters	True	Predicted (1 Well)
Variable x_c	8	8
Variable y_c	10	10
Variable C_o	70	70
Fitness	-	1.71E-08
Solution Error %	-	0 %

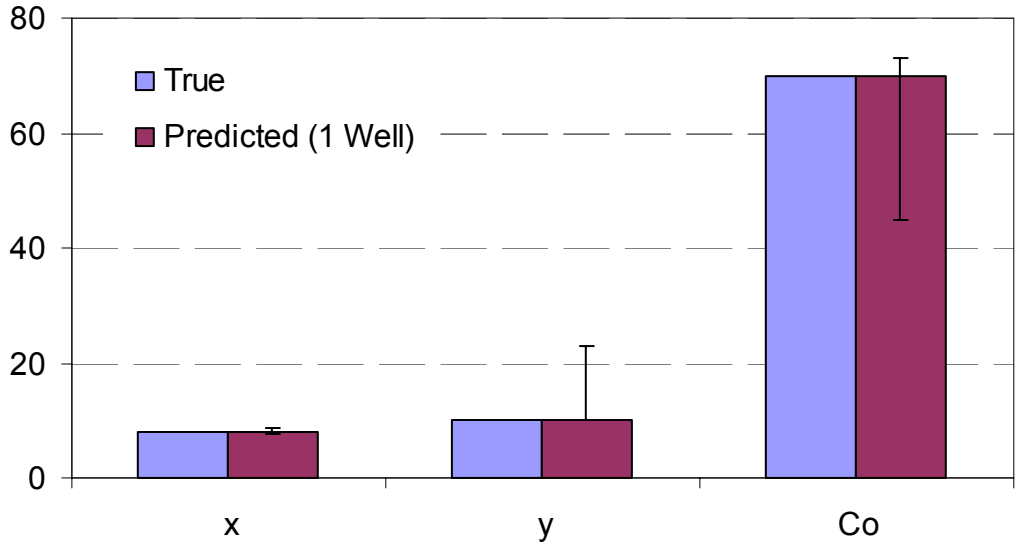

Figure 3-12. Best Source Characterization Solution for Scenario 2D-1 Utilizing Collected Observations from One Well along with Result for 30 Trials

3.5.2.1 Solution of the Two-Dimensional Problem Scenarios

In the 2D-1 scenario, the collected observations used in the study were obtained from only one observation well (well 2, see Fig. 3-4). Table 3-7 shows the best results (estimated parameters) found compared with the true parameters along with their objective function error and solution error. Figure 3-12 illustrates the best found parameters compared with the true parameters together with maximum and minimum range of parameters that obtained from 30 GA trials. Using one observation well, the best solution indicates that the GA was able to estimate the parameters perfectly with objective and solution error almost equal to zero.

Table 3-8. Best Source Characterization Solution for Scenario 2D-2 Utilizing Collected Observations from One and Three Wells

Parameters	True	Predicted (1Well)	Predicted (3 Wells)
Variable x_c	8	7.991	8
Variable y_c	10	10.045	10
Variable s	2	1.984	2
Variable C_o	70	70.281	70
Objective Function Error	-	6.87E-04	3.59E-07
Solution Error %	-	0.44%	0%

Scenario 2D-2 is more complex than scenario 2D-1, in terms of the number of parameters that should be estimated. In this scenario two investigations were attempted based on information available to conduct the search, using one well and three wells to collect observation data (wells 1-3, see Fig. 3-4). Table 3-8 shows the best results found for both investigations, compared with the true parameters along with their objective and solution error. Figure 3-13 illustrates the best found parameters for both investigations compared with

the true parameters together with maximum and minimum range of parameters that were identified for 30 GA trials. The results from both studies indicate that the GA was able to estimate the parameters almost identically to the true parameters, with objective error close to zero and solution error equal to 0.007 using one well and 0.5% using three wells.

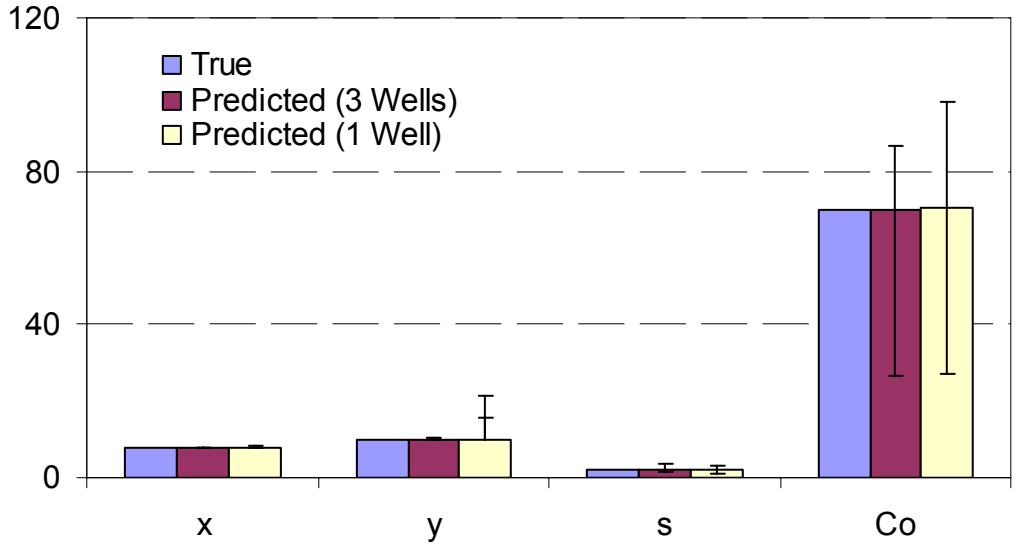

Figure 3-13. Source Characterization Solution for Scenario 2D-2 Utilizing Collected Observations from One and Three Wells along with results for 30 trials

Table 3-9. Best Source Characterization Solution for Scenario 2D-3 Utilizing Collected Observations from One, Three and Five Wells

Parameters	True	Predicted (1Well)	Predicted (3 Wells)	Predicted (5 Wells)
Variable x	8	7.9844	7.9925	7.9996
Variable y	10	10.0687	9.9966	10.0001
Variable s_x	2	1.9835	2.002	2.0353
Variable s_y	2	2.0529	1.9955	1.9579
Variable C_o	70	68.4893	66.0285	70.1737
Objective Function Error	-	1.60E-03	4.50E-03	9.19E-04
Solution Error %	-	1.302%	1.224%	0.825%

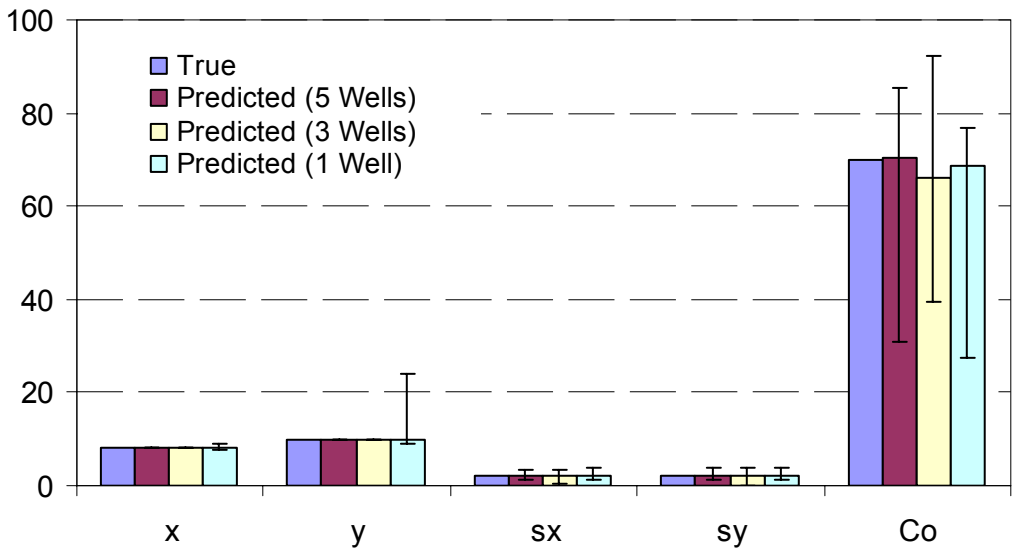

Figure 3-14. Source Characterization Solution for Scenario 2D-3 Utilizing Collected Observations from One, Three and Five Wells along with Results for 30 Trials

Scenario 2D-3 contains five parameters to be estimated which characterizes the source more precisely than scenario 2D-1 and 2D-2. Considering this complexity three investigations were completed using observation data from one well (well 1), three wells (wells 1-3), and five wells (wells 1-5), see Figure 3-4. The best results found for the three studies, compared with the true parameters along with the objective and solution error are demonstrated in Table 3-9. Figure 3-14 illustrates the best parameters compared to the true parameters together with maximum and minimum range of parameters that obtained from 30 GA trials. The solutions obtained for the three studies are each close to the true parameters. The GA was able to identify solutions with objective errors less than 0.005 and solution errors around 1%. The results from the third study (using five wells) indicate that the GA was able to predict the parameters perfectly with objective error and solution error close to zero.

Table 3-10. Best Source Characterization Solution for Scenario 3D-1 Problem Utilizing Collected Observations from Four Wells

Parameters	True	Predicted (4 Wells)
Variable z_c	4	4
Variable x_c	8	8
Variable y_c	10	10
Variable C_o	70	70
Objective Function Error	-	1.21E-07
Solution Error %	-	0 %

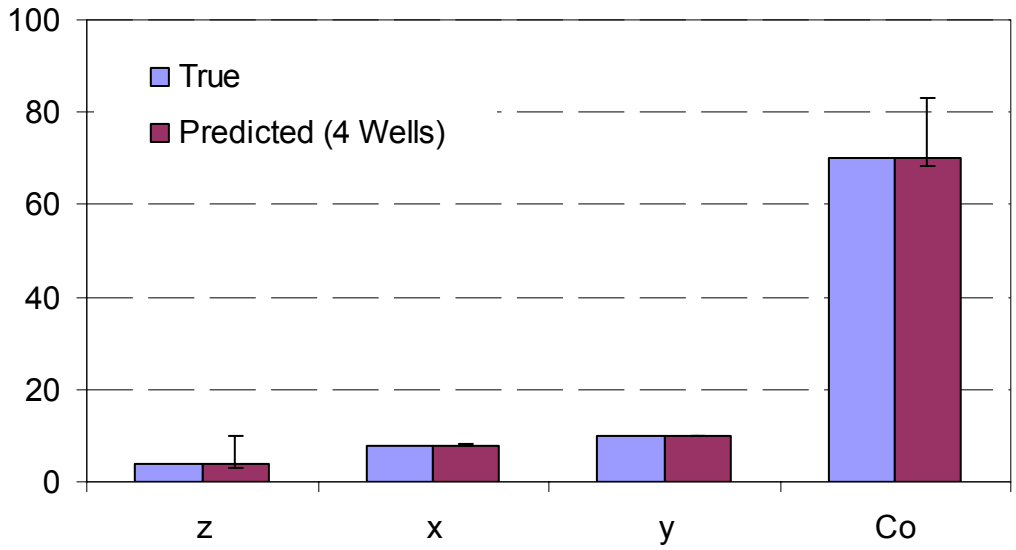

Figure 3-15. Source Characterization Solution for Scenario 3D-1 Utilizing Collected Observations from Four Wells along with Results for 30 Trials

3.5.2.2 Solution of the Three-Dimensional Problem Scenarios

The investigation procedures used to address 3D scenarios is similar to the investigation used for 2D scenarios; however more collected observations (number of wells) are used to address more complex scenarios. In the 3D-1 scenario, the collected observations used in the study were obtained from four observation wells (wells 11, 13, 15 and 17, see Fig. 3-5). Similar to previous scenarios, Table 3-10 and Figure 3-15 present the best results

found compared with the true parameters, the objective and solution error, and maximum and minimum range of parameters that obtained from 30 GA trials. The results show that the estimated parameters are identical to the observed ones with objective and solution error equal to zero.

For Scenario 3D-2, one investigation uses collected observations from four wells, and a second uses collected observations from nine wells (wells 1-9, see Fig. 3-5). Again, Table 3-11 and Figure 3-16 illustrate the best results found for both investigations compared to the true parameters, objective error and solution errors, and the solution ranges obtained for 30 GA trials. The results from the first study (one well) indicate that the GA identify solutions with low objective error (0.016), but large solution errors (44%). The results from the second study (three wells) indicate that the GA was able to predict the parameters perfectly with objective and solution error equal to zero. The solutions obtained for both investigations are almost identical to the true parameters. The GA was able to predict the parameters perfectly with objective error less than 0.008 and solution error around 0.2%.

Table 3-11. Best Source Characterization Solution for Scenario 3D-2 Problem Utilizing Collected Observations from Four Wells

Parameters	True	Predicted (4 Wells)	Predicted (9 Wells)
Variable z_c	4	4.02	4.00
Variable x_c	8	7.99	8.00
Variable y_c	10	10.00	10.00
Variable s	2	2.00	2.00
Variable C_o	70	69.73	69.72
Objective Function Error	-	7.96E-04	7.90E-04
Solution Error %	-	0.228 %	0.127 %

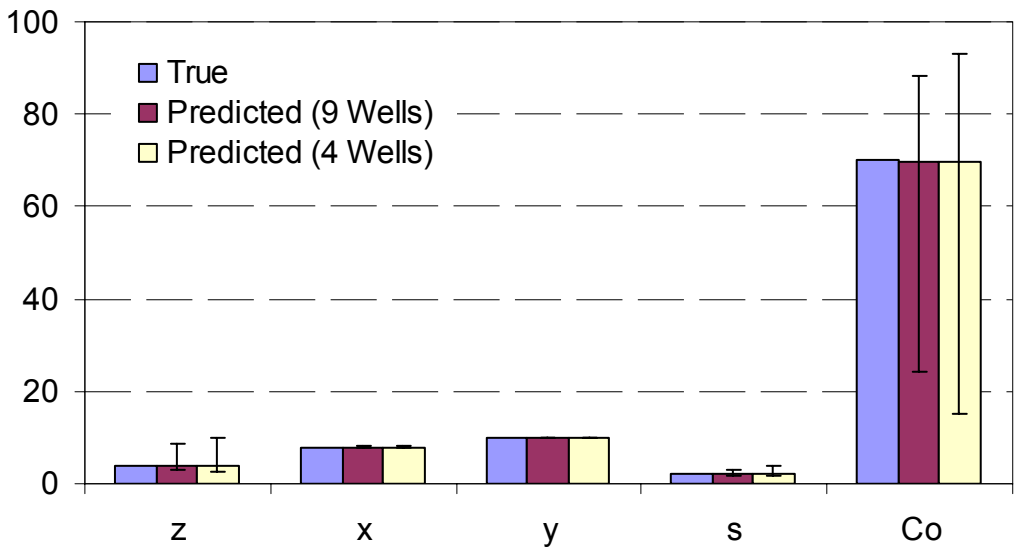

Figure 3-16. Source Characterization Solution for Scenario 3D-2 Utilizing Collected Observations from Four and Nine Wells along with Results for 30 Trials

Table 3-12. Best Source Characterization Solution for Scenario 3D-3 Problem Utilizing Collected Observations from Four Wells, Nine Wells and Eighteen Wells

Parameters	True	Predicted (4 Wells)	Predicted (9 Wells)	Predicted (18 Wells)
Variable z	4	4.21	4.03	4.05
Variable x	8	7.85	7.96	7.97
Variable y	10	9.99	9.99	10.00
Variable s_z	2	2.14	2.11	2.04
Variable s_x	2	2.61	2.00	2.05
Variable s_y	2	2.23	2.52	2.08
Variable C_o	70	51.17	56.94	66.18
Objective Function Error	-	0.019	0.026	0.073
Solution Error %	-	11.90 %	7.43 %	2.19 %

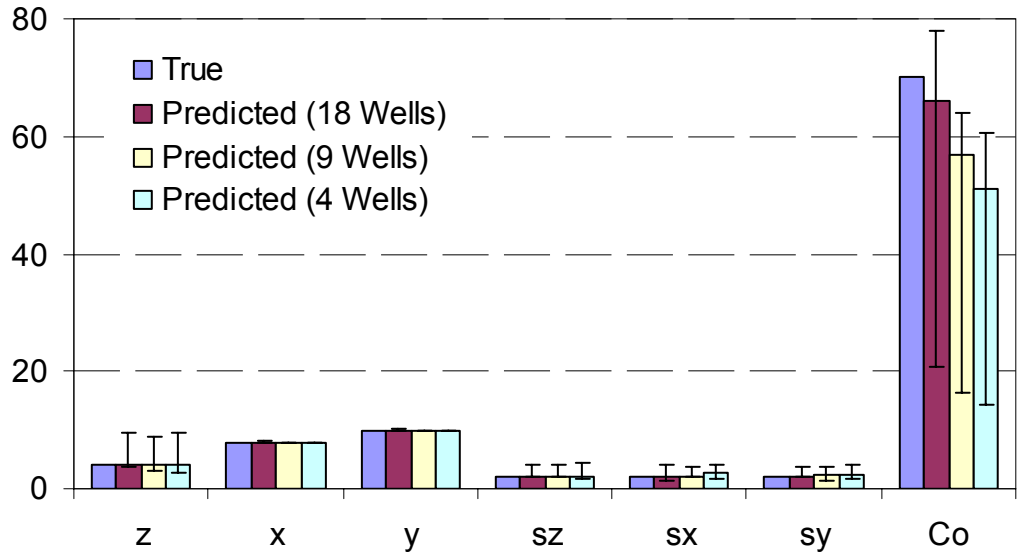

Figure 3-17. Source Characterization Solution for Scenario 3D-3 Utilizing Collected Observations from Four, Nine and Eighteen Observation Wells with Results for 30 Trials

Scenario 3D-3 is considered the most complex scenario in this research, since it represents a three-dimensional problem with seven parameters needed to characterize the source. Three investigations were completed, using information from 4 wells, 9 wells and 18 wells (wells 1-18, see Fig. 3-5). Similar to previous scenarios, Table 3-12 and Figure 3-17 show the results for the three studies. The results indicate that the GA was able to estimate the parameters for the three studies with objective error around 0.02, 0.025 and 0.075, respectively, with solution error around 12%, 8% and 2%. The quality of the solutions obtained for the third investigation is nearly identical to the observed parameters (true solution).

For each study in the 2D and 3D scenarios, the GA performance was examined. For example, the GA convergence for the 2D-3 and 3D-3 scenarios utilizing five and 18 observation wells are demonstrated in Figures 3-18 and 3-19 respectively. The best and the worst objective function error (fitness) values among 30 trials were plotted corresponding to

number of generations. The GA convergence for the two and three-dimensional scenarios is similar.

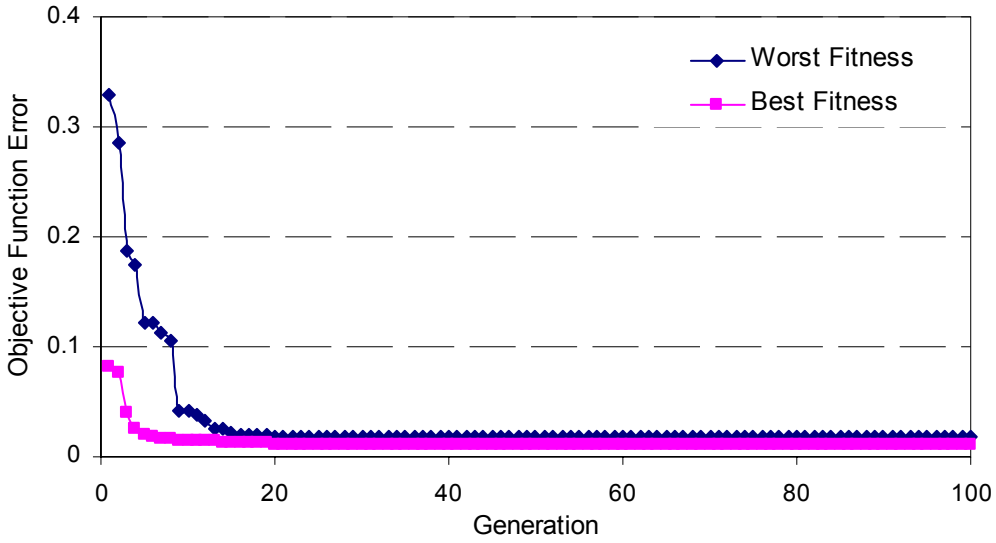

Figure 3-18. Best and Worst GA Convergence of each Generation for 30 Trials of Scenario 2D-3 Utilizing 5 Observation Wells

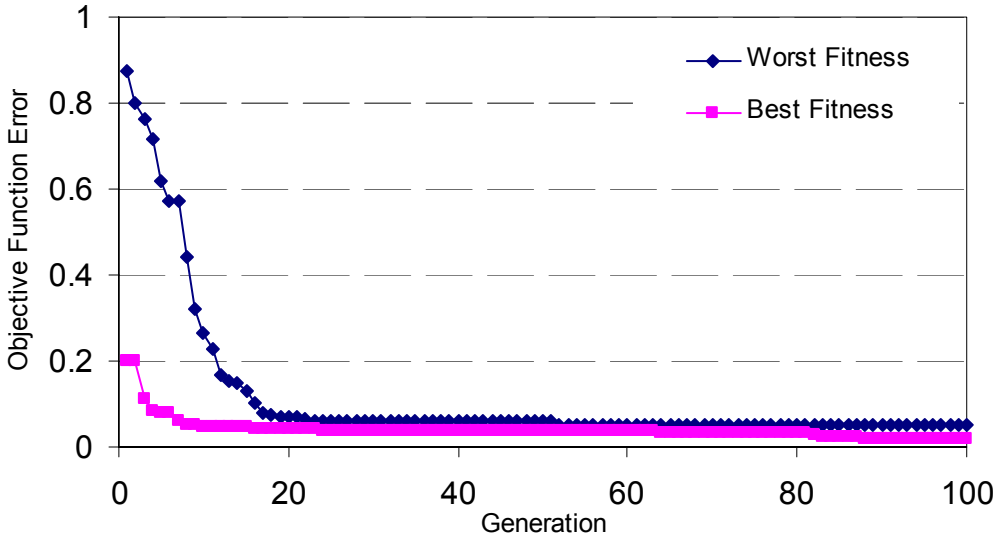

Figure 3-19. Best and Worst GA Convergence of each Generation for 30 Trials for Scenario 3D-3 Utilizing 18 Observation Wells

In the 2D scenarios (Figures 3-12, 3-13 and 3-14), the solutions ranges obtained from 30 trials are close to zero for the *x* parameter and mostly negligible for the *y* and the *s*

parameters, while the solution ranges are relatively large for the C_o parameter. For the 3D scenarios, the solution ranges (Figures 3-15, 3-16 and 3-17) are close to zero for the *x* and the *y* parameters and mostly negligible for the *s* parameters. The solution ranges are relatively large for the *z* and the C_o parameters. It is noticeable that the parameter ranges widen as the scenario complexity increases. Additionally, for both 2D and 3D scenarios, the ranges diminish in studies with more collected observations. This could be the result of non-uniqueness, where many source characterizations result in similar (or close enough) concentration profiles. For instance, a large source with a small concentration has a similar profile as a small source with a large concentration. Therefore using more information helps to reduce the non-uniqueness in the problem.

As a summary, Figure 3-20 presents the solution error for each scenario; the results show that a unique solution can be found as more information is used to solve the problem, though in some cases an adequate solution can be identified with limited information, such as the 2D-1 scenario.

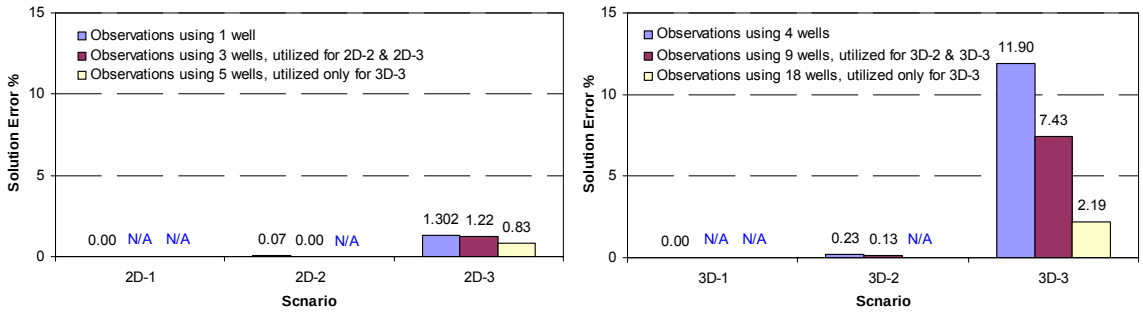

Figure 3-20. Solution Error for all Scenarios

The solution of the source identification problem assumes that the observed data recorded at observation wells are perfect without measurement errors. Typically, observed data has some error, or noise. To investigate the influence of noise on the groundwater source

identification problem, the entire investigation is repeated using noisy data. The results are shown in Appendix A-I and Appendix A-II for the 2D and 3D scenarios respectively. In this study, the ANN surrogate model predicts the simulation model very closely, but with some error, approximately 5% to 10% prediction error. The objective function (Eqn. 3-6) is calculated using simulated and modeled concentration profiles, the ANN surrogate model is still used to generate the modeled concentrations, while the simulation model is used to generate the observed concentration profile. Figure 3-21 demonstrates the solution error for each scenario. The results indicate that with noisy data, the non-uniqueness in the problem increases significantly. In this case, acceptable solutions are identified only when more information is used to solve the problem. For the most complex scenario (3D-3) the solution error remains high.

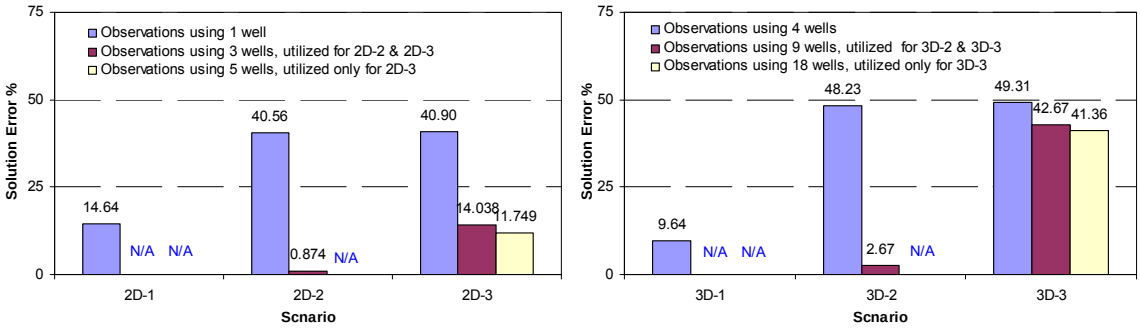

Figure 3-21. Solution Error for all Scenarios Using Noisy Observation Data

Table 3-13. Timing Study for the ANN Surrogate and Simulation Models

G.W. Forward Model	Computing Resources	Run Time (sec)
Simulation Model (FORTRAN)	TeraGrid NCSA, 1 Node, 1 CPU Intel Itanium 2, 1.5 GHz, 2 GB of RAM	40.49
ANN Surrogate Model (FORTRAN)	TeraGrid NCSA, 1 Node, 1 CPU Intel Itanium 2, 1.5 GHz, 2 GB of RAM	0.061

3.5.3 Timing Study Analysis

One of the main objectives of this research is to increase the efficiency of the search through the use of a computationally fast surrogate model. Therefore a timing study is conducted to compare the execution time for the simulation model to the time for the ANN surrogate model. The ANN surrogate model developed using MATLAB was implemented in FORTRAN to enable a fair comparison to the FORTRAN-based simulation model. Table 3-13 shows the execution time for the 3D groundwater simulation model and the ANN surrogate model along with the computation resources used to conduct the study. The results indicate that the FORTRAN-based ANN model is approximately 650 times faster than the simulation model. Though a considerable amount of time was utilized to investigate, train and develop the ANN surrogate models, this time is not integrated in this timing study.

3.6 Final Remarks

This paper investigates the use of the ANN surrogate model in reducing the computational burden in the simulation-optimization approach. The simulation-optimization approach was utilized to solve two and three-dimensional groundwater inverse problems, where the solution procedure employs an Evolutionary Computation-based search procedure (GA). The results indicate a significant success of ANN surrogate models in predicting relatively accurate concentration profiles for all experiment scenarios and for both training and test data. The results also demonstrate that the search procedure is able to estimate correctly the decision variables for most scenarios. The effect of the non-uniqueness in the solution is more significant as the problem complexity increases; this complexity includes problem domain, number of parameters to be estimated, observed data accuracy (noise)

and/or amount of information used to solve the problem. It is more difficult to estimate the z-direction parameters in the 3D scenarios, since the contaminant signatures in z-direction are less significant. Finally, the timing study analysis shows that the ANN surrogate model is able to achieve three orders of magnitude improvement in the execution time of the simulation model in comparison to the simulation model. This leverage provided by the ANN surrogate modeling allows us to investigate the CSI problem more broadly which would have been unachievable using typical simulation-optimization approach.

CHAPTER 4

EFFICIENT GA-BASED EMBEDDED HYBRID OPTIMIZATION METHODS

Abstract

The use of hybrid methods to attempt complex optimization problems has become significantly popular in the last few years. Since then there have been a wide range of hybrid optimization algorithms developed. Inverse problems are relatively complex to solve due to the complex nature of the problem that includes several local minima and a highly discontinuous decision space. The simulation-optimization approach is considered to solve this problem, which may add another level of complexity in terms of computation time since the approach is based on many iterations. Inverse problem may be addressed by using a combination of global and local search technique in a sequential order. Although sequential hybrid methods can produce high quality solutions to the inverse problem, they cannot be applied efficiently in distributed or parallel environments. In this research, parallel (embedded) hybrid procedures are developed to address inverse problems instead of sequential hybrid methods. The main idea in the developed procedures is to embed the strength of local optimization method into a powerful global optimization method. Embedded hybrid methods are evaluated and compared with global, local and sequential hybrid methods in terms of convergence characteristics, solution quality and computation time.

4.1 Introduction

Inverse problems could be described as problems where the answer is known, but not the conditions that led to it. For example, the process of determining model inputs from model outputs is known as inverse problems. Inverse problems are usually ill-posed, meaning they suffer from non-uniqueness (multiple solutions), non-existence (no solution) and solution instability (small error in the measured observations results in a large variation in the estimated parameters). As a result of this, recent research had shown that inverse problems may have several local minima and a highly discontinuous decision space (Mahinthakumar, and Sayeed, 2005). Several approaches are available to address inverse problem, in this research the focus is on simulation-optimization approach where the problem formulated as an optimization problem. Simulation-optimization approach can be computationally expensive because of two possible reasons: the iterative nature of the optimization technique where many simulation models need to be evaluated, and each simulation model evaluation can be expensive since simulation model usually are a system of partial differential equations (Atmadja and Bagtzoglou, 2001). Many examples of inverse problems are present in engineering and science applications. For instance, in groundwater contamination characterization problems, the unknown contaminant location and the release profile are estimated based on observation data. Detailed description of inverse problem generally and groundwater inverse modeling specifically are available in many references (e.g., Sun, 1994; Poeter and Hill, 1997).

Consequently, solving inverse problem using simulation-optimization approach are relatively challenging due to two factors: (1) computational intractability of the solution approach, and (2) ill-posed nature of the problem. The emergence of high performance

computing and the associated increase in available computing power can be used to address the first challenge. The new technology allows reducing the potential computational intractability of simulation-optimization approach drastically by distributing the simulation model evaluations among a grid of computing nodes. Concerning the second challenge, evolutionary based algorithms such as genetic algorithms show a significant efficiency in getting near global minimum; however, they may require fine-tuning near global minimum for better results. Several previous studies had proven that hybrid optimization methods, namely the sequential hybrid methods, show a considerable success in solving inverse problems (e.g., Espinoza and Minsker, 2006).

Hybrid methods for optimization are a practical approach for solving some complex optimization problems. Hybrid methods can be defined as a combination of two or more optimization search algorithms. This combination can be constructed in several different ways such that the best qualities of each search algorithm are incorporated into one procedure. For instance, combinations of evolutionary algorithms and local search methods yield efficient search algorithms for solving complex engineering and environmental optimization problems (Talbi, 2002).

Sequential hybrid methods, a commonly reported hybrid technique, are divided into several types based on the sequence of the optimization methods used. For example, Mahinthakumar and Sayeed (2005) developed GA-LS, a hybrid genetic algorithm (GA) and local search (LS), that first used a GA to find quickly and globally a good solution, and then fine-tuned that good solution using a local search method. While the sequential methods are effective, these methods do not necessarily maintain the parallelism to attain maximum computational efficiency. The main objective of this study is to investigate parallel

(embedded) hybrid methods that are computationally efficient while achieving same or better solution quality as sequential hybrid methods. Unlike sequential hybrid methods, embedded hybrid methods are designed such that the local search steps are integrated into the global search mechanism.

Relatively, limited number of studies has investigated embedded hybrid methods (e.g., Oh et al., 2004). The common theme among these reported methods is to develop efficient optimization procedures that function robustly in a parallel environment. Partheepan (2003) investigated a number of embedded methods where local search was integrated into GA operators such that the strength of both global and local search is embedded into the population-based procedure that is easily parallelized. One method was constructed by adding the Hooke-Jeeves exploration moves to the elitism operator to explore for locally better solutions in the neighborhood of the elite solution. Another procedure incorporated an alternative crossover operator that is designed based on the Nelder-Mead simplex method to perform local search.

Similar to the work reported by Partheepan (2003), the embedded hybrid methods developed in this research explore different ways to integrate local search steps into the highly parallelizable genetic algorithms. Key features of this research include: (1) improving the crossover operator with local search steps; (2) testing and illustrating the new methods using an instance of groundwater inverse problem; and (3) comparing the performance of the new embedded methods with that if global and local search methods, and sequential hybrid methods.

4.2 Optimization Methodologies Background

In general, a hybrid optimization method could be defined as a combination of several search procedures that are integrated to construct a more efficient method than each individual procedure. A general classification of optimization methods are given in Figure 4-1, where the methods are classified based on different criteria. Based on the solution procedure, optimization methods can be grouped into direct methods (also known as classical methods such as linear programming) and indirect methods in which the solution procedure is based on educated trial-error procedure. Indirect methods are divided into local and global methods based on the search steps in the decision space. Based on the solution procedure, the local-search methods are grouped into gradient and non-gradient based procedures, while the global-search methods can be classified to heuristic and non-heuristic methods. The emphasis of this paper is on hybrid methods constructed using local non-gradient-based methods and global heuristic methods (they appear with shaded boxes in Figure 4-1). These hybrid methods combine the commonly used local non-gradient methods (Hooke and Jeeves and Nelder Mead), and evolutionary algorithms (e.g., genetic algorithm and evolution strategies), a frequently used global heuristic method.

4.2.1 Hybrid Methods Classification

Hybrid optimization algorithms classification provides a common terminology and classification mechanisms; it also allows comparison of hybrid algorithms in a qualitative way (Talbi, 2002; Cotta et al., 2005). Few references in literature discuss hybrid method design and taxonomy; an excellent reference by Talbi (2002) provides a comprehensive study

on hybrid methods classification. Some of these classifications are customized and used in this research.

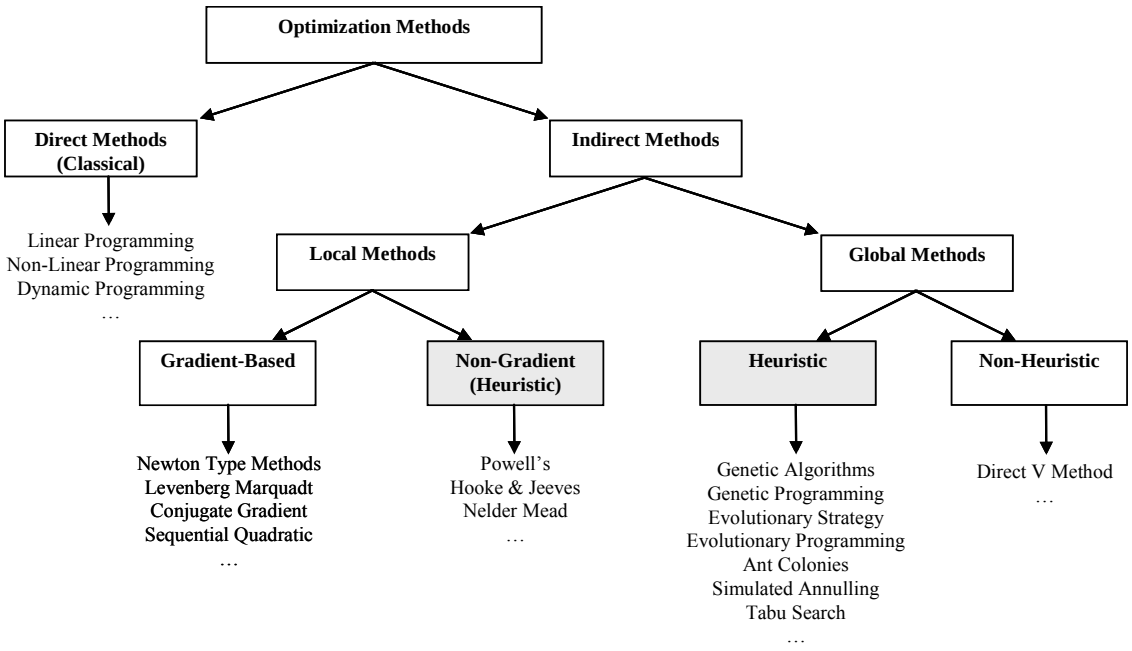

Figure 4-1. A General Classification of Optimization Methods

The hybridization of optimization methods involves several issues, which may be grouped as design and implementation issues. Furthermore design issue is classified into architecture and functionality issues, shown in Figure 4-2. The emphasis on this study is on architecture design issues where hybrid methods are classified based on their structure into:

*1. Low level versus High level***:** The low-level hybridization addresses the functional composition of a single optimization method. In this hybrid class, a given function of an optimization method is replaced by another optimization method. In high-level hybrid algorithms, the different optimization methods are self-contained. In this hybrid class, there is no direct relationship to the internal procedures of an optimization method.

*2. Sequential versus Parallel***:** In sequential hybridization, a set of optimization methods is applied one after another, each using the output of the previous as its input, acting in a pipeline fashion. Parallel hybridization represents cooperative optimization models, in which we have many cooperating agents/methods. Four classes are derived from hybrid architecture design classification:

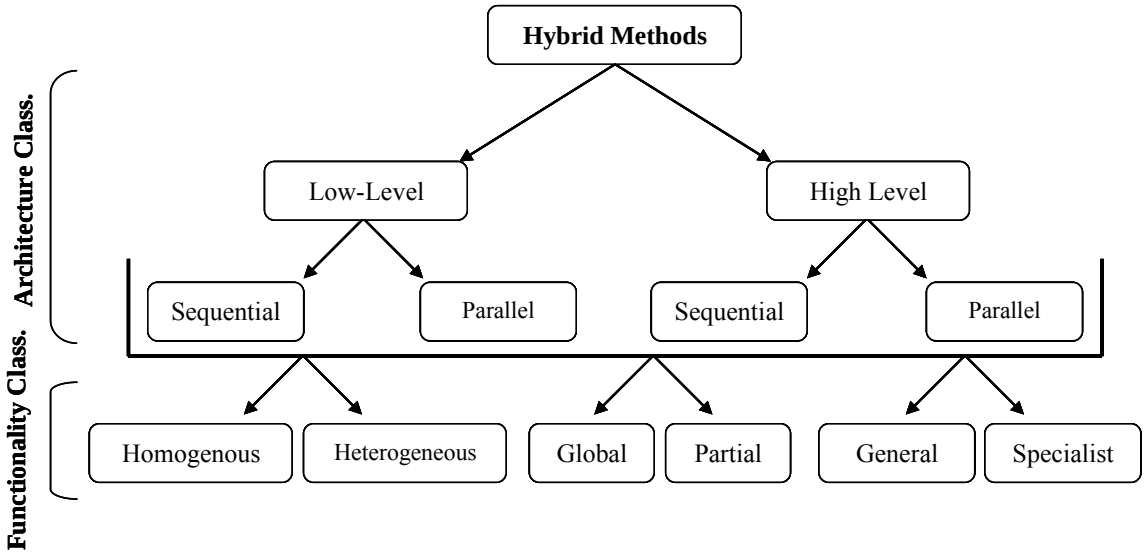

Figure 4-2. Classification of Hybrid Methods (Architecture Design Issues) , Modified from Talbi (2002)

*A. Low level Sequential Hybrid (LSH)***:** This procedure represents algorithms in which a given search method is embedded into a single-solution search method. For example, a procedure introduced by Martin et al. (1992) to solve the traveling salesman problem where a deterministic local search technique (Markov Chain) embed into non-generational global search method (Simulated Annealing) so that the Markov Chain explores only local optima.

*B. Low-level Parallel Hybrid (LPH)***:** A balance exploration (global search) and exploitation (local search) technique is needed for better and efficient optimization procedures. In this procedure, a local optimization method is embedded into a global population-based method

as an attempt to achieve this balance. For instance, when a GA is used as a global optimizer, its standard operators may be replaced by enhanced local optimization methods such as in Fleurent and Ferland (1996) and Partheepan (2003).

*C. High-level Sequential Hybrid (HSH)***:** In this scheme, self-contained methods are executed in a sequence. For example, it is well know that evolutionary algorithms based methods are efficient in quickly locating the global minima or maxima regions. Once those regions are located, it may be useful to apply local search methods for fine tuning. Several previous researches have successfully attempted this procedure such as Abbattista et al. (1995) and Mahinthakumar and Sayeed (2005).

*D. High-level Parallel Hybrid (HPH)***:** This procedure involves several self-contained algorithms performing a search in parallel, and cooperating to find the optimum solution. Instinctively, HPH will perform at least as well as one algorithm alone, and often performs better. An example of HPH is the GAs based island model, where the population is divided into small subpopulations by geographic isolation. A GA evolves each subpopulation and individuals can migrate between subpopulations. The island model is controlled by several parameters such as the topology that defines the connections between subpopulations, the migration rate that controls the number of migrant individuals, the replacement strategy used, and migration interval that affects how often migration occurs. A number of studies managed to implement this procedure with different degrees of success using different global optimization methods, for example in Bachelet et al. (1996) a parallel GA is proposed while in Badeau et al. (1997) a parallel Tabu Search is utilized.

Given a set of optimization methods M_i, the above procedures can be summarized as LSH(M_1(M_2)), LPH(M_1(M_2)), HSH(M_1+M_2), and HPH(M_1, M_2) respectively. The focus of

this study is on hybridization of GA-based heuristic search methods with local search methods, i.e., LPH(M_1(M_2)) procedures.

4.2.2 Global and Local Methods Overview

Among several optimization methods available, this research conducted based a global heuristic search method (Genetic Algorithms) and two local heuristic search methods (Hooke-Jeeves and Powell's). A brief overview of these methods is presented in the sections below:

4.2.2.1 Global Search - Genetic Algorithms

A genetic algorithm is a search procedure used to find true or approximate solutions to optimization and search problems. Genetic algorithms are a particular class of evolutionary algorithms that use techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover (also called recombination). Genetic algorithms are implemented as a computer simulation in which a population of abstract representations (called chromosomes or genotype) of candidate solutions (called individuals or phenotypes) to an optimization problem evolves toward better solutions. Traditionally, solutions are represented in binary as strings of zeros and ones (0s and 1s), but other encodings are also possible. The GA starts with a set of potential solutions (population), usually selected randomly. The performance of each solution is characterized by a fitness value. During the GA search process, the population is iteratively subjected to several probabilistic operators that are analogous to natural selection, mating (genetic recombination), and mutation. Each iteration through these steps forms a generation. Because

fitter solutions are more likely to be selected for mating, the incidence of good traits of a solution tends to increase from one generation to the next. Crossover assists to sample these traits in many different combinations. Typically, the quality of solutions improves quickly at the beginning of the algorithm. As the population converges, improvements diminish. Usually, termination criteria are used to stop the procedure, such as if the total number of generations is met or improvements are sufficiently small that additional computations are not necessary. If the algorithm has terminated due to a maximum number of generations, a satisfactory solution may or may not have been reached.

Genetic algorithms find application mainly in engineering, science, and economics fields. The GA has powerful ability to combine aspects of solutions from different parts of decision space to perform global search. Once the population converged to a small region in the search space, where there is no great difference in the solution, the GA global search ability is weakened. The GA can be computational expensive because of the iterative nature however it's agreeable for use in a parallel and/or distributed computing environment since the fitness evaluation for each individual in a generation can proceed independently. The GA fundamentals are well explained in Goldberg (1989) and Davis (1991).

4.2.2.2 Local Search - Hooke-Jeeves Method

Pattern Search is an important branch of the climbing techniques family of optimization algorithms that can be applied to non-linear optimization problems. The Pattern Search is a fairly simple nevertheless powerful exploratory search technique. It is based upon the conjecture that a set of moves which was successful in the first trials is worth trying again. Hooke-Jeeves technique is a version of the pattern search technique, it originally

introduced in 1961 by R. Hooke and T. A. Jeeves (Hooke and Jeeves, 1961). The Hooke-Jeeves method consists of two parts, one is a local exploratory scheme to explore the neighborhood of the current solution, and the other is the acceleration part to move to a new solution. The exploratory step starts with perturbing the current solution (start with a random guess) positively and negatively in each direction by a fixed value (step size) such that an improved solution is found. In case no better solution found in all directions the exploratory step will be repeated using a reduced step size. The step size gradually reduced until an improvement is found or it met a given limit in which case the algorithm terminates. In case a better solution is found, a pattern direction is determined by calculating the difference between the improved solution and the old solution. Then in the next step the new solution is found by extrapolating the old solution along the pattern direction. A reasonable step size and stopping criteria in terms of number of iteration or step size reduced limit are essential for better performance of the method. Both exploring and pattern search step can be explained as follows:

Exploration

$$P_{ij} = P_{i0}, \qquad for\ i \neq j, \qquad i, j = 1,...,N \tag{4-1}$$

$$P_{ij} = P_{i0} \pm U_i S, \qquad for\ i = j, \qquad i, j = 1,...,N \tag{4-2}$$

Pattern Search *(if exploration successful)*

$$P_{ij} = 2P_{ij} - P_{i0}, \qquad i, j = 1,..,N \tag{4-3}$$

where P_{i0} is the initial starting point, N is number of decision variables, U is the unit vector ($Nx1$) along direction i, and S is the step size. Details description of the algorithm is available in Hooke and Jeeves (1961) and McDonald (2001).

4.2.2.3 Local Search - Powell's Conjugate Direction Method

Powell's Conjugate Direction method is one of the most popular non-gradient search algorithms for unconstrained optimization problem. The method was first developed by M. J. Powell in 1964 and based on the idea of computing conjugate directions without calculating derivatives (Powell, 1964). Powell's method of conjugate directions performs minimization along successive directions that are conjugate with respect to all previous directions. Powell's is the method of choice if a reasonable starting approximation is available. The method generates a conjugate set which is a sequence of line minimizations. It starts by assuming a set consisting of the unit directions, and then gradually calculates new directions from the results of the search. The following heuristic scheme is given to illustrate the procedure: Set a starting point P_0 in an n-dimensional search space. Initialize non-linear independent directions to the basis vector D_i for $i = 1,..., n$. The algorithm repeats the following steps until the error function reaches a minimum or termination criteria are satisfied:

- Save starting position as P_0.
- For $i = 1,..., n$, minimize the error function starting from P_{i-1} along the direction D_i and store the minimum as the next position P_i. After the loop all P_i are computed.
- Let D_i be the direction of the largest decrease. Now this direction D_i is replaced with the direction given by (P_n-P_0).
- The iteration process continues with the new starting position ($P_0 = P_n$), until the minimum is reached.

Details procedure for computing the conjugate direction set is given in Powell (1964) and Press et al. (1996).

4.3 Embedded Hybrid Methods

While there are several hybridization techniques, the emphasis of this research is on algorithms that directly embed the local search steps into a standard genetic algorithm such that the inherent parallelism of GAs is maintained. In the following section, the construct of three parallel hybrid methods is discussed.

4.3.1 GA with Hooke-Jeeves' Search – GA(HJ)

Prior research efforts report that the GA crossover operator could be adjusted to increase exploitation through local search steps (Preux and Talbi, 1999; Partheepan, 2003). In the new GA(HJ) procedure, an alternative crossover operator is developed. This operator is designed based on the Hooke-Jeeves pattern search step that explores the immediate neighborhood of a selected solution. The new operator is applied to a subset (δ)% of the sorted population while the rest of the population, i.e., ($100 - \delta$)% , undergoes the standard GA crossover operator. The subset size δ is increased dynamically with the number of generations to increase the local exploitation as the GA population converges.

The pattern move is applied based on the last best solutions and the current best solutions to generate new solutions. Suppose P_l is the last best solution, and P_c is the current best solution. If $P_c \neq P_l$, then a new solution $P_n = P_c + \theta\,(P_c - P_l)$ is created; otherwise, i.e., if $P_c = P_l$, then a new solution $P_n = P_c + 2\theta$ is created. The parameter θ represents the local step size, and is usually assigned a value between 0.25 and 1.0. This pattern move implies a line search in the direction of local improvements. The resulting new solution is incorporated into the subsequent GA population. If the resulting new solution violates a constraint, then it

is set to a point on the constraint boundary. The main steps of the algorithm are described in the following pseudo code:

```
GA(HJ){
    while termination criteria is not met{
            Evaluate the population ();
            Selection ();
            Sort population ();
            Elitism ();
            Crossover {
                Create two populations:
                Population 1:  top δ% of the sorted population
                Population 2:  (100 – δ)% of the population, randomly picked
                Hooke-Jeeves Crossover () on Population 1{
                    if P_l ≠ P_c
                        P_n = P_c + θ (P_c – P_l)
                    else
                        P_n = P_c + 2θ
                }
                Standard GA Crossover () on Population 2
            }
            Mutation ();
    }
}
```

4.3.2 GA with Powell's Search – GA(Pow)

Similar to GA(HJ) described above, an alternative crossover operator is developed based on Powell's local search method to optimize along successive direction. As in GA(HJ), the new operator is applied to a subset (δ)% of the population while the rest of the population ($100 - \delta$)% undergoes the standard GA crossover operator. The subset size is increased dynamically with the number of generations.

The new operator starts by assuming a set consisting of unit directions, a new direction is gradually calculated based on the last best solutions. Suppose P_c is the current solution in an n-dimensional search space, P_l is the last best solution, and D_i for $i = 1,...,n,$ is

the unit based direction. If $P_c \neq P_l$ the new solution P_n is created based on optimizing the objective function along direction $D_p = P_c - P_l$; otherwise, i.e., if $P_c = P_l$, then a new solution P_n is created based on optimizing along the direction D_i. The resulting new solution is incorporated into the subsequent GA population. If the resulting new solution violates a constraint, then it is set to a point on the constraint boundary. The algorithm will be repeated until the error function reaches a minimum or the termination criteria are met. The main steps of the algorithm are described in the following pseudo code:

```
GA(Pow){
    while termination criteria is not met{
        Evaluate the population ();
        Selection ();
        Sort population ();
        Elitism ();
        Crossover {Evaluate the population ();
            Create two populations:
            Population 1:       top δ % of the sorted population.
            Population 2:       (100 - δ)% of the population, randomly picked
            Powell's Crossover () on Population 1{
                if Pc ≠ Pl
                    Pn will be obtained based on optimizing along direction Dp = Pc - Pl
                else
                    Pn will be obtained based on optimizing along direction Di
            }
            Standard GA Crossover () on Population 2
        }
        Mutation ();
    }
}
```

4.3.3 Local GA Procedure – GA(Local)

In this procedure the settings of the standard GA crossover operator is modified to conduct a local search around the best solutions. The new crossover operator is similar to the standard GA operator except its parameters (e.g., crossover type and probability) are adjusted

to increase the degree of exploitation. As in the previous two methods, the new operator is applied to a subset of the population while the rest of the population undergoes the standard GA crossover operator, and the subset size increases with the number of generations. For instance, the heuristic crossover with 80% probability can be used for the standard crossover operator, while the heuristic crossover with 50% probability can be used for the local crossover operator. The main steps of the algorithm are described in the pseudo code:

```
GA(Local){
    while termination criteria is not met{
        Evaluate the population ();
        Selection ();
        Sort population ();
        Elitism();
        Crossover {
                Create two populations:
                Population 1:  top δ % of the sorted population.
                Population 2:  (100 - δ)% of the population, randomly picked
                Local GA Crossover() on Population 1
                    {Crossover with more exploitation features}
                Standard GA Crossover() on Population 2
                    {A balance exploration and exploitation Crossover}
        }
        Mutation ();
    }

}
```

4.4 Illustrative Application

4.4.1 Application Description

In this study a two-dimensional homogeneous groundwater contaminant source characterization problem is considered. A simulation-optimization procedure is used in which a MATLAB-based two-dimensional groundwater transport model is employed to

simulate the fate and transport of the contaminant plume. The governing equations describing the groundwater transport are fully explained in many references (e.g., Bear, 1972). This problem represents an unknown contaminant release (or multiple contaminant releases) at a single contaminant source (or multiple contaminant sources) that is to be resolved from spatially and temporally distributed concentration observations collected at predefined monitoring wells. Concentration observations at monitoring wells are collected to generate a concentration time-series at each monitoring location. For a possible contamination source characteristics (defined by the source location, size and contaminant loading), the resulting concentrations at the monitoring wells are estimated using a groundwater transport model (forward model). The goal of the optimization method is to search and identify the source characteristics that yield concentration estimates that best match the observations. Figure 4-3 illustrates the groundwater source identification problem in a two-dimensional groundwater domain.

Two instances of the groundwater contaminant source characterization problem are attempted here. The first one is the contaminant source identification problem where the problem assumes that a single contaminant source location (x_1, y_1, x_2, y_2) and the contaminant concentration (C_0) of a constant continuous source are unknown. Thus for this problem there are five decision variables. The second problem is the contaminant source identification and release history reconstruction problem (referred to as the combined problem), where the problem assumes that a single contaminant source location (x_1, y_1, x_2, y_2) and the contaminant loading (C_t) at time interval t at the source are unknown. In this problem, five time intervals are used to represent the contaminant releases history, thus for this problem there are nine

decision variables. More details about these types of inverse problems are provided by, for example, Sun (1994) and Aral et al. (2001).

The combined problem is relatively harder to solve than the source identification problem because of the nature of the problem and the number of parameters to be estimated. Both problems are described below as an optimization model where the normalized root squared error (RSE) between the observed and simulated concentrations is minimized (see Eqn. 4-4). The constraints, shown in Equations (4-5)-(4-7), are included to enforce feasibility bounds on the decision variables.

$$\underset{DV^{Prob}}{\text{minimize}} \sqrt{\frac{\sum_{j=1}^{nw}\sum_{i=1}^{n}\left(C_{ij}^{obs} - C_{ij}^{sim}\right)^2}{\sum_{j=1}^{nw}\sum_{i=1}^{n}\left(C_{ij}^{obs}\right)^2}} \tag{4-4}$$

subject to:

$$0 \le C_0, C_t \le C_{max} \tag{4-5}$$

$$x_{min} \le x_k \le x_{max}; \qquad y_{min} \le y_k \le y_{max}; \qquad k = 1, 2 \tag{4-6}$$

$$x_1 \le x_2; \qquad y_1 \le y_2; \tag{4-7}$$

where

DV^{Prob} is the set of parameters need to be estimated for each problem where,

$DV^1 \in \{x_1, y_1, x_2, y_2, C_0\}$ *for the source identification problem,*

$DV^2 \in \{x_1, y_1, x_2, y_2, C_t\}$ *for the combined problem,* where $t = 1, 2, \ldots, 5$

C_{ij}^{obs} is the observed concentration at time step (i) at observation well (j),

C_{ij}^{sim} is the simulated concentrations at time step (i) at observation well (j),

x_1, x_2, y_1 and y_2 are the coordinates of the lower left and upper right corners of the contaminant source,

n is the total number of time steps,

nw is the total number of observations wells,

C_{max} is the maximum possible concentration,

x_{min}, x_{max}, y_{min} and y_{max} are the 2-D groundwater domain (aquifer) boundaries,

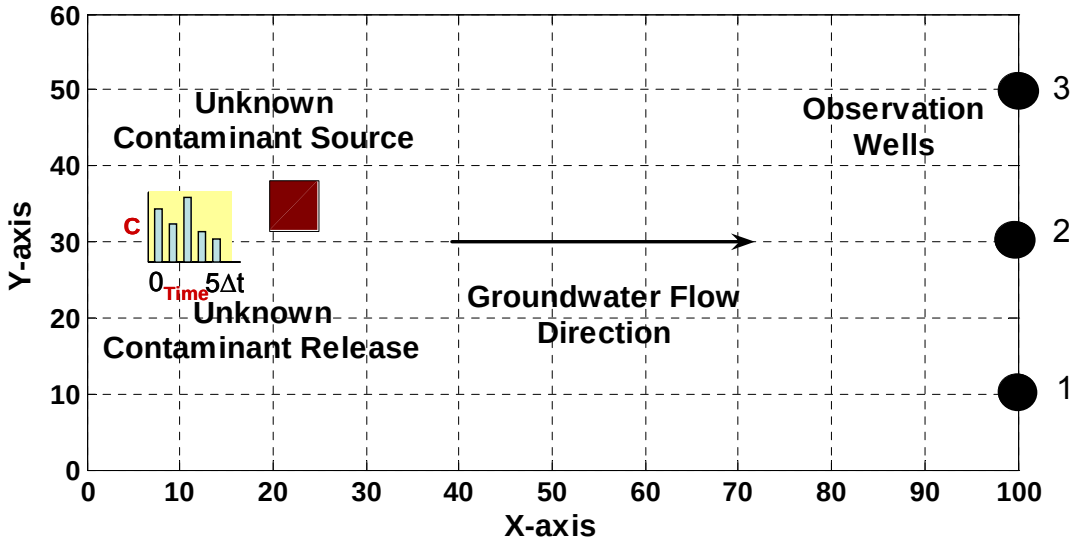

Figure 4-3. Hypothetical Two-Dimensional Groundwater Domain, Axes Units are Meter

Table 4-1. Hypothetical Two-Dimensional Domain Parameters

Parameter	Source Identification Problem Settings	Combined Problem Settings
Problem Size	51x31 grid	
Grid Spacing (dx, dy)	2 m, 2 m	
Number of Time Steps	50	
Total Time	100 day	1000 day
Time Step Size (dt)	2 day	20 day
Dispersion Parameters	α_L = 1 m, α_T = 1 m, D_m = 0.01 m^2/day	
Groundwater Field	Homogeneous	
Velocity	1 m/day	
Number of Observation Wells	3	
True Source Location and Concentrations	x_1=14, y_1=28, x_2=20, y_2=34, C_0=70	x_1 = 14, y_1 = 28, x_2 =20, y_2=34, C_1=70, C_2=40, C_3=30, C_4=20, C_5=30

4.4.2 Problem Domain and Simulator Settings

In this study, a hypothetical homogenous two-dimensional field (100m x 60m) with a single conservative contaminant source was considered. The grid resolution for the simulation model resulted in 1,581 (51x31) finite element nodes within the groundwater domain, and the simulation is performed for 50 time steps, using a total of three monitoring wells located close to the downstream end of the field. Details of the groundwater simulation model parameters are shown in Figure 4-3 and Table 4-1.

Table 4-2. Global Method (GA) Settings

Parameter	GA
Evaluations	5000
Generation	50
Population Size	100
Variables Type	Real
Selection	Stochastic Uniform
Crossover	Heuristic, 0.80
Mutation	Gaussian

Table 4-3. Local Method Settings

Parameter	Hooke-Jeeves	Powell's
Evaluations	5000	5000
Variables Type	Real	Real
Initial Step Size	20	-

Table 4-4. Sequential Hybrid Method Settings

Parameter	GA+HJ		GA+Pow		GA+Local	
	GA	HJ	GA	Pow	GA	Local
Evaluations	50000		5000		5000	
Population Size	100	-	100	-	100	100
Variables Type	Real	Real	Real	Real	Real	Real
Selection	Stochastic Uniform	-	Stochastic Uniform	-	Stochastic Uniform	Binary Tournament
Crossover	Heuristic, 0.80	-	Heuristic, 0.80	-	Heuristic, 0.80	Heuristic, 0.50
Mutation	Gaussian	-	Gaussian	-	Gaussian	Uniform
Initial Step Size	-	20	-	-	-	-

4.4.3 Search Parameters Settings

The following search procedures were used to test and evaluate the performance of the three embedded hybrid procedures described in Section 4.3: a standard Genetic Algorithm (GA); the Hooke-Jeeves (HJ) local search method; the Powell's Conjugate Direction (Pow) method for local search; a standard GA followed by the Hooke-Jeeves method (GA+HJ); a standard GA followed by Powell's method (GA+Pow); and a standard GA followed by GA(Local) for local search (GA+Local). While the first method (GA) implements a global search procedure, the second (HJ) and the third (Pow) methods implement local search procedures. The last three methods (i.e., GA+HJ, GA+Pow, and GA+Local) are sequential hybrid methods that conducts a GA-based global search first and then a local search to improve on the solution found by the GA. Some of these methods were utilized in previous studies (e.g., Mahinthakumar and Sayeed, 2005) to solve groundwater inverse problems. For the results presented below in this paper, all the GA-based methods were implemented utilizing the MATLAB GA toolbox. More details about the GA operators (such as crossover, selection and mutation) used in this research are available in the MATLAB user guide (MATLAB, 2006). For the local search methods, standard settings were utilized to conduct the search. For sequential hybrid procedures, the search starts with the GA then switch to a local search until the termination criterion is met (or the number of evaluations exceeds a specified maximum number of evaluations). The procedure switches from the GA search to a local search when the prediction error (Eqn. 1) improvement is less than 1.0E-5 for five consecutive generations. For the new embedded hybrid methods, the population subset size δ was increased dynamically from 0% to 15% based on the formula shown in Equation 4-8. The search parameter settings for all procedures are summarized in

Tables 4-2, 4-3, 4-4, and 4-5. The boundary parameter values used for this problem to constrain the search are shown in Table 4-6.

$$\delta\ (\%)\ =\ \frac{1}{100}\times round\left(\frac{Max.\ Population\ subset}{Total\ No.\ of\ \ Generations}\times Current\ \ Generation\right) \quad (4\text{-}8)$$

Table 4-5. Embedded Hybrid Method Settings Used for the Problems

Parameter	GA(HJ)	GA(Pow)	GA(Local)
Evaluations	5000		
Generation	50		
Population Size	100		
Population Subset (δ)	[0% to 15 %]*		
Variables Type	Real		
Selection	Stochastic Uniform		
Crossover for Population 1	Heuristic, 0.8		
Crossover for Population 2	Hooke-Jeeves	Powell's	Heuristic, 0.50
Step Size (θ)	0.5	-	-
Mutation	Gaussian		

*δ = 0% at generation 1 and then increases dynamically to 15 % at generation 50

Table 4-6. Allowable Range of Decision Variable Values for both Problems

Variables	Ranges [Minimum-Maximum]
x axis	[0-100] m
y axis	[0-60] m
C_0, C_t	[0-100] mg/l

4.5 Results and Discussion

In this section, the performance of the new hybrid methods is tested and compared with other optimization methods. A set of runs is conducted to investigate the convergence

behavior and the solution quality of the new hybrid methods. Six optimization procedures were used as a reference for a comparative evaluation and assessment of the performance of the new methods.

The first set of runs was conducted to test the overall convergence characteristics for the new methods and to compare with those of the sequential hybrid method. This was applied to the contaminant source identification problem illustrated in Section 4.4.1 and the search settings shown in Section 4.4.3. To examine robustness of convergence of these methods, thirty random trials were conducted for each method. A termination criterion based on total number of evaluations was used, i.e., the procedure was terminated when total number of evaluations reached 5000.

The typical convergence behaviors of the new methods GA(HJ), GA(Pow) and GA(Local) are presented in Figures 4-4, 4-5 and 4-6, respectively. The graphs show the variation of the best and the worst prediction error (i.e., objective function values) obtained at each generation over the 30 trials. This implies that all solutions, among the 30 random trials, at a particular generation number would have had objective function values within the best and worst values at that generation number. In general, the difference between the worst and best values at each generation number (especially towards the end of the convergence) indicates the degree of robustness; the smaller the difference, higher the level of robustness. The difference between worst and best values at the end of the generations for GA(HJ), GA(Pow) and GA(Local) are 0.08, 0.10 and 0.13, respectively. The convergence plots for all three methods are similar to the typical robust convergence behavior; however, the convergence of GA(HJ) procedure shows slightly better performance than the other two procedures.

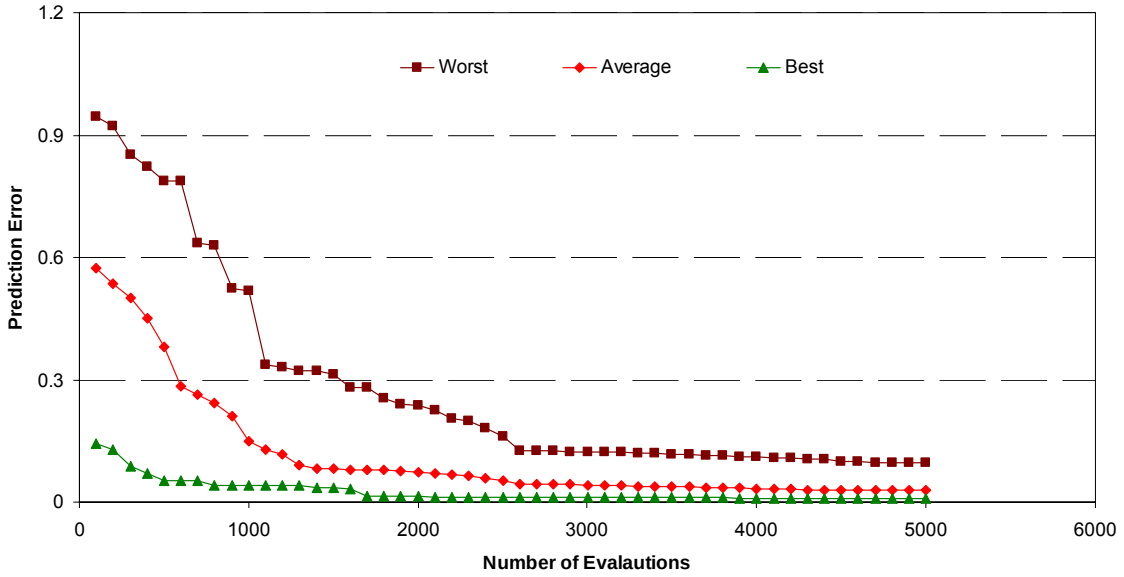

Figure 4-4. Convergence Behavior of GA(HJ) Procedure for the Source Identification Problem, Indicated by the Variation of the Best, Average and the Worst Values for Prediction Error over the 30 Random Trials

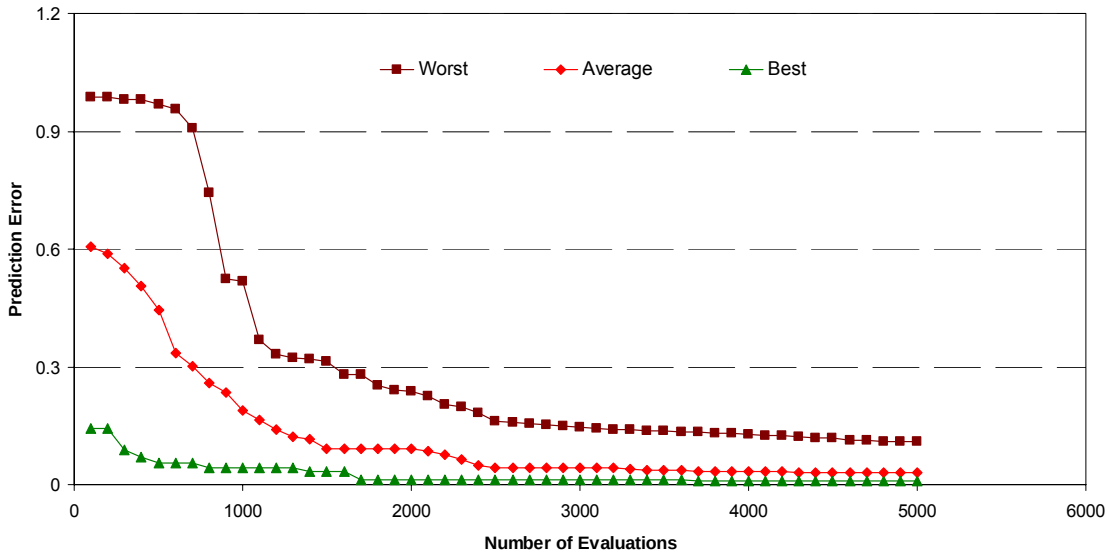

Figure 4-5. Convergence Behavior of GA(Pow) Procedure for the Source Identification Problem, Indicated by the Variation of the Best, Average and the Worst Values for Prediction Error over the 30 Random Trials

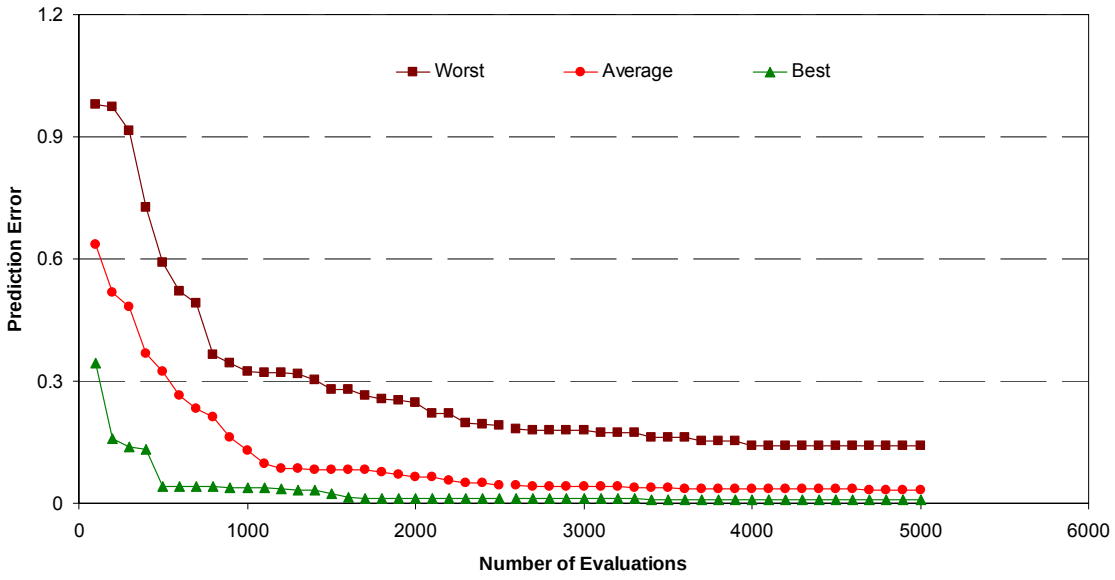

Figure 4-6. Convergence Behavior of GA(Local) Procedure for the Source Identification Problem, Indicated by the Variation of the Best, Average and the Worst Values for Prediction Error over the 30 Random Trials

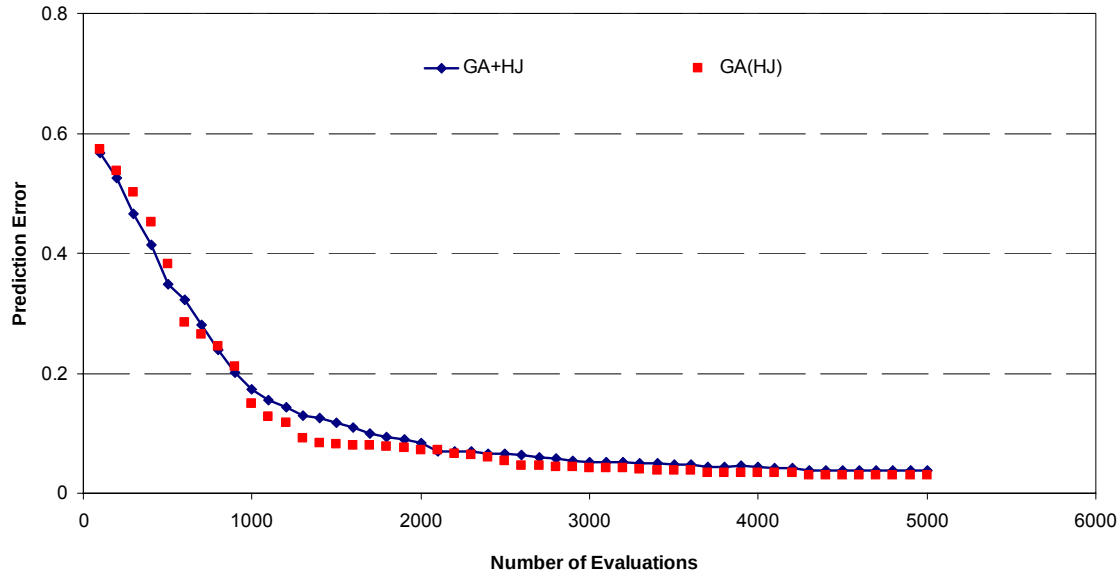

Figure 4-7. Convergence of the Sequential (GA+HJ) and Embedded GA(HJ) Methods shown by the Variation of the Average Values for Prediction Error over the 30 Random Trials

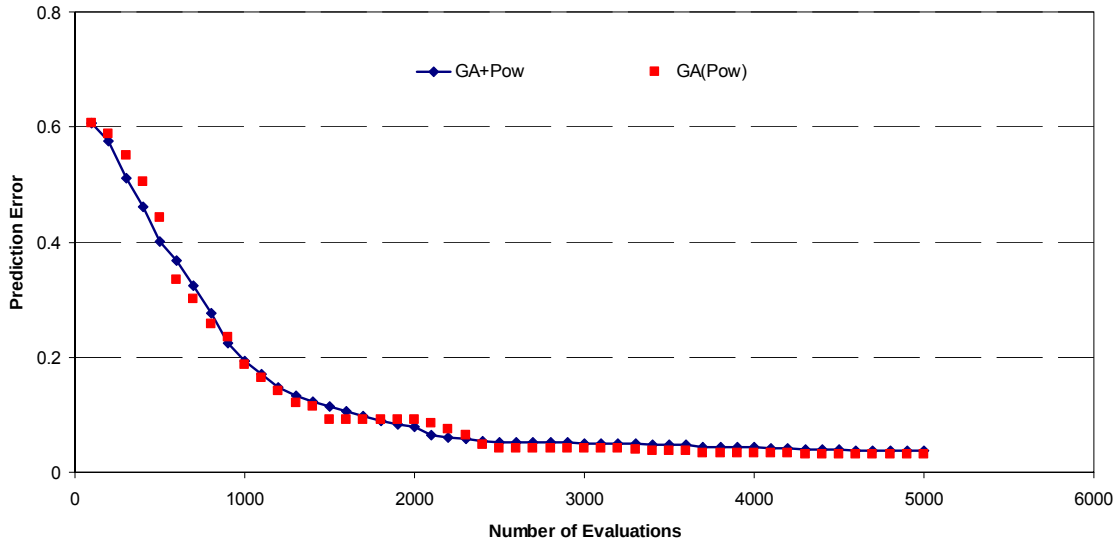

Figure 4-8. Convergence of the Sequential (GA+Pow) and Embedded GA(Pow) Methods shown by the Variation of the Average Values for Prediction Error over the 30 Random Trials

The typical convergence behavior of the new embedded hybrid procedures are compared with those of the GA-based sequential hybrid methods in Figures 4-7, 4-8 and 4-9. These figures show the variation of the average prediction errors obtained at each generation over the 30 trials. Figure 4-7 shows the convergence of the sequential GA+HJ versus the embedded GA(HJ) procedure, Figure 4-8 shows the convergence of the sequential GA+Pow versus the embedded GA(Pow) procedure, and Figure 4-9 shows the convergence of the sequential GA+Local versus the embedded GA(Local) procedure. These figures illustrate that, in general, the hybrid embedded methods show better convergence behavior in comparison to the corresponding sequential hybrid methods.

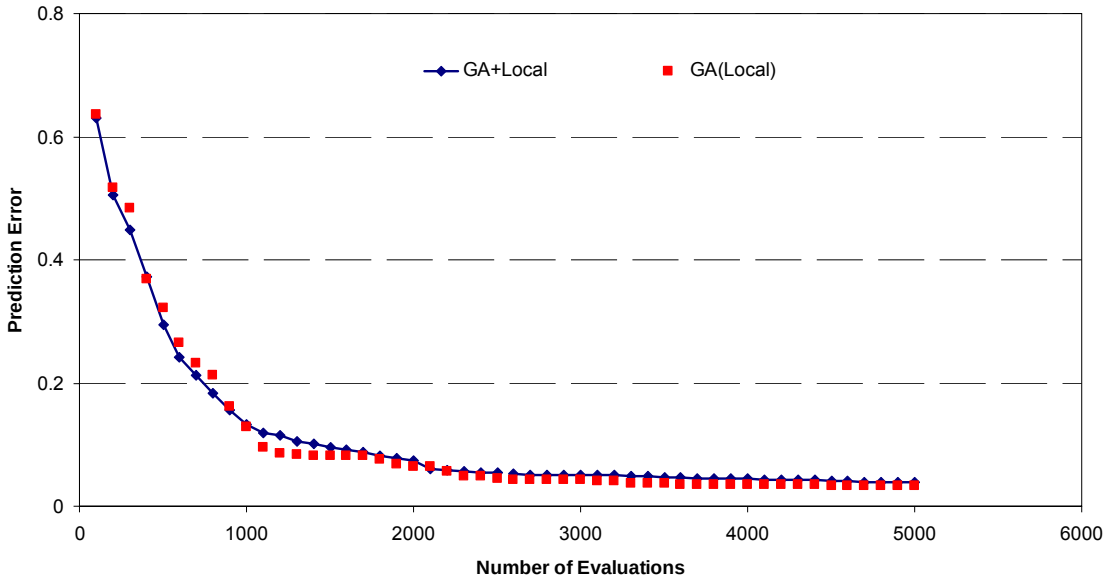

Figure 4-9. Convergence of the Sequential (GA+Local) and Embedded GA(Local) Methods shown by the Variation of the Average Values for Prediction Error over the 30 Random Trials

A second set of runs was conducted to compare the quality of the solutions obtained using the global search, local search and sequential hybrid search methods with those obtained using the new embedded hybrid procedures. Solution quality is described in terms of the prediction error, solution error, and the computational requirements (represented by the number of evaluations). All procedures were applied to both the contaminant source identification problem and the combined problem illustrated in Section 4.4.1 with the search settings shown in Section 4.4.3. Similar to previous investigation, 30 random trials were conducted for each method. For a fair comparison between methods, each procedure was terminated when the total number of evaluations reached 5000 or the prediction error improvement was less than 1.0E-8 for five consecutive generations, i.e., 500 consecutive evaluations.

The corresponding solution quality for each method is shown in Table 4-7 and Table 4-8 for the source identification problem and the combined problem, respectively. The results

include the average prediction error (Eqn. 4-4), the average solution error, and number of evaluations. The solution error (SE) is calculated as follows.

$$Solution\ Error\ (SE)\ \% = \sqrt{\sum_{i=1}^{n}\left(\frac{P^{estimated} - P^{true}}{P^{true}}\right)^2} \times \frac{100}{n} \qquad (4\text{-}9)$$

where n is the number of variables, $p^{estimated}$ is the estimated decision variable value from the optimization method, and p^{true} is the decision variable value of the true solution. There are five and nine decision variables for the source identification problem and the combined problem, respectively. The number of evaluations is the sum of the number of forward simulation model runs executed at convergence of each procedure.

The accuracy of a solution is compared to that of another solution using the objective function error and the solution error. A low normalized RSE means that the concentration profiles at the observation wells are closer to the reference (observed) profile, and a low SE implies that the location of the estimated solution is closer to the true source characteristics. SE is a useful metric only in test cases where the true solution is known (as in the synthetic case studies reported here). In real field applications, only the RSE will be known. Based on the results presented in Table 4-7 for the source identification problem, the embedded hybrid methods show a noticeably better prediction and solution error compared to the global search method and the local search methods, and a slightly better performance in comparison to the sequential hybrid methods. Based on the results presented in Table 4-8 for the combined problem, again the embedded hybrid methods show better prediction and solution error compared to the global search method and the local search methods, and similar performance as the sequential hybrid methods.

The computational burden for a method can be compared based on the number of evaluations required to achieve an acceptable solution quality. A low number of evaluations indicates lower computational burden, indicating a relatively inexpensive method. The number of evaluations is recorded when the procedure terminates, thus for each of the nine methods the stopping criterion may be satisfied after different numbers of function evaluations. The results for both problems indicate that all hybrid embedded methods utilize a larger average number of function evaluations in comparison to that required by the global search, local search and sequential hybrid methods. Although local and global search methods recorded lower number of evaluations, they attained a poorer solution quality in comparison to those obtained using the embedded hybrid methods. On other hand, the sequential hybrid methods achieve a comparable solution quality with low numbers of evaluations in comparison to the embedded hybrid methods. However, the embedded hybrid methods can be executed in a parallel in a distributed computing environment (i.e., function evaluations can be executed in parallel), enabling faster computing and quicker turn-around times for solutions.

Table 4-7. Performance Comparison of the all Nine Methods for Solving the Source Identification Problem

Optimization Methods		Average Prediction Error	Average Solution Error	Average Number of Evaluations
Global Method	GA	0.04261	14.63%	2900
Local Methods	Hooke-Jeeves	0.39383	29.28%	1113
	Powell's	0.46734	34.57%	974
Sequential Methods	GA+HJ	0.03796	9.02%	3698
	GA+Pow	0.03691	8.77%	3829
	GA+Local	0.03989	9.96%	3600
Embedded Methods	GA(HJ)	0.02993	6.55%	3900
	GA(Pow)	0.03181	6.11%	4000
	GA(Local)	0.03376	8.18%	3800

Table 4-8. Performance Comparison of the all Nine Methods for Solving the Combined Problem

Optimization Methods		Average Prediction Error	Average Solution Error	Average Number of Evaluations
Global Method	GA	0.36008	22.12%	3200
Local Methods	Hooke-Jeeves	0.95040	53.17%	1099
	Powell;s	0.81944	46.21%	923
Sequential Methods	GA+ HJ	0.05361	8.66%	3867
	GA+ Pow	0.04477	6.73%	4020
	GA+ Local	0.05542	9.11%	4100
Embedded Methods	GA(HJ)	0.06210	9.22%	4200
	GA(Pow)	0.04732	5.98%	4100
	GA(Local)	0.05489	8.84%	4300

Table 4-9 shows the best (among the 30 trials) predicted values of the decision variables for the source identification problem for each method, and these solutions are compared with the corresponding true values. The corresponding solution error values (Eqn. 6) shown in Table 4-9 indicate good performance of the embedded hybrid and sequential hybrid methods (less than 0.5% deviation from the true source characteristics) for characterizing the source.

Table 4-9. The Best Solution Found by each Method for the Source Identification Problem

Optimization Methods		Best Predicted Solution					Solution Error
		x_1	y_1	x_2	y_2	C_0	
Global Method	GA	13.8	28.2	20.1	33.3	70	0.55 %
Local Methods	HJ	7.1	25.3	24.7	35.5	22.1	17.58 %
	Pow	2.9	27.4	31.8	27.6	68.6	20.14 %
Sequential Methods	GA+HJ	14.1	28	19.1	34.4	70.2	0.49 %
	GA+Pow	13.9	27.6	20.3	33.8	69.7	0.46 %
	GA+Local	14.3	28.1	19.9	34.1	70	0.45 %
Embedded Methods	GA(HJ)	14	28	19.6	34	70.2	0.41 %
	GA(Pow)	13.8	27.9	20.2	34.1	71	0.47 %
	GA(Local)	13.9	28	19.7	33.7	69.2	0.44 %
True Solution		14	28	20	34	70	-

Table 4-10 summarizes for the predicted values of the decision variables for the combined problem for each method, and these solutions are compared with the corresponding true values. The corresponding solution errors are also shown. The solution errors indicate that the embedded and sequential hybrid methods perform adequately (less than 3.5% deviation from the true source characteristics) when characterizing the source.

Table 4-10. The Best Solution Found by each Method for the Combined Problem

Optimization Methods		Best Predicted Solution									Solution Error
		x_1	y_1	x_2	y_2	C_1	C_2	C_3	C_4	C_5	
Global Method	GA	14.1	28.5	19	33	88.2	55.5	38.8	28.9	39.4	8.64%
Local Methods	HJ	21.3	22.5	37	31.5	77.2	56.2	48.8	54.9	47.4	24.84%
	Pow	18.1	26.2	27	43	84.4	53.5	45.8	47.9	52.4	19.94%
Sequential Methods	GA+HJ	14.4	29.3	22.4	35.8	69.6	42.5	31.8	17.9	28.4	2.28%
	GA+Pow	14.5	27.6	23.8	34	70.2	38.5	27.7	18.8	30.4	2.88%
	GA+Local	13.9	27.8	21	33.8	78.2	36.1	28.8	23.2	32.4	2.71%
Embedded Methods	GA(HJ)	13.4	25.4	18.1	36.3	68.8	43.5	31.8	21.9	33.4	2.76%
	GA(Pow)	15.1	26.8	20	38	71.9	35.5	28.4	23.8	27.4	3.12%
	GA(Local)	14.8	24.6	22.2	36.7	72.2	36.6	32.9	22.8	31.5	3.07%
True Solution		14	28	20	34	70	40	30	20	30	-

4.6 Final Remarks

Optimization hybrid methods have been explored and studied to improve the efficiency of the search algorithms. Three embedded hybrid procedures that combined the strength of genetic algorithms and local search methods were presented. Using varieties of optimization methods (global, local and sequential hybrid methods) as a reference, the performance of the embedded hybrid methods were tested and evaluated. The evaluation and comparison were conducted based on the convergence behavior and the solution quality of each method. In this study, two instances of two-dimensional groundwater inverse problem with different complexity were utilized as illustrative applications; the source identification

problem and the release history reconstruction problem. As an observation, the overall convergence behavior of the embedded hybrid methods shows comparable or better performance than standard genetic algorithms and sequential hybrid methods. The results for the embedded hybrid procedures show that the new global-local search hybrid algorithms are able to achieve high-quality solutions for the both application problems. The embedded hybrid methods were developed in the first place to improve the efficiency of sequential hybrid methods in parallel environments. As a remark, all embedded methods showed relatively higher number of evaluations than the sequential hybrid methods. Nevertheless, unlike sequential methods, the embedded hybrid methods can be fully executed in a parallel/distributed environment which gives embedded methods leverage in parallel/distributed environment. Finally, although this study shows reasonable performance improvement in using embedded methods, a broader set of test problems with different level of difficulties is essential and needed as future work. Also this investigation could be extended to a more complex application problem.

CHAPTER 5

NOISY GA-BASED SEARCH TO ADDRESS UNCERTAINTY IN REAL-WORLD INVERSE PROBLEMS

Abstract

Solving inverse problems using the simulation-optimization approach has been increasingly attractive among researches in the last decade. The approach is based on the numerous fitness evaluations for the simulation models. For complex engineering and environmental problems, recurrent fitness evaluations may be computationally expensive. Addressing inverse problems under parameters of uncertainty remained so far an under-explored young research arena. Few optimization methods however, are somewhat successful in solving the inverse problem with a considerable number of successful trials. The Monte Carlo simulation modeling and noisy genetic algorithms are examples of these few methods. These methods adopt a relatively large number of realizations or samples in evaluating the fitness function which makes the simulation-optimization approach computationally prohibited. The present work develops an efficient genetic algorithms based procedure that operates in noisy environment and applies to an example of inverse problems. The idea is based on using fewer realizations in solving the inverse problem while maintaining a comparable or better solution quality to previous noisy genetic algorithms procedures. The new procedure was tested and compared to a standard noisy genetic algorithms procedure by using two instances of inverse groundwater characterization problem. The findings show a considerable promise in reducing the number of evaluations and applying to a groundwater characterization problem under noisy environment.

5.1 Introduction

Inverse problems are mathematical relationships that apply into many fields such as environmental science, engineering, economics and recently medical fields. An inverse problem can be defined as the problem where the system (or model) inputs are unknown while the system (or model) outputs are known (Poeter and Hill, 1997). Solving inverse problems requires determining the model parameter(s) from the model observed data. Inverse problems vary in type according to the estimated model parameters which include input and structural parameters (Sun, 1994).

The available inverse problem solution approaches are mainly based on the observed data (model output). Therefore, highly accurate observed data is vital to appropriately (or robustly) solve inverse problems. This may not be the case in many real-world applications. Simulation-optimization approach is one among many approaches that are adopted to solve inverse problems. The simulation-optimization approach can be defined as the process of finding the best input parameters from among all possibilities without explicitly evaluating each possibility (Yolanda and Anu, 1997). In the simulation-optimization approach, the simulation model (forward model) is coupled with optimization techniques to determine the model inputs that best represent the observed data in an iterative process. When the number of input parameters is large and the simulation model is complex, the simulation-optimization approach may become computationally prohibited. Reviews on simulation-optimization introduction and application are readily available in a variety of dependable references (e.g., Andrad´ottir, 1998).

Solving inverse problems is rather complex due to ill-posedness developing from non-existence, non-uniqueness and instability of the solution (Sun, 1994). Attempting inverse

problem by using inaccurate (or noisy) observed data adds an additional level of complexity to the process. In real-world applications, uncertainties could be associated with simulation model input parameters (such as, for example, uncertainty in spatially distributed groundwater parameters) or with the structural parameter (such as, for example, variability in conceptual or numerical assumptions in the simulation model) (Zheng and Bennett, 2002). There are a few available approaches that may help address uncertainty in a noisy environment. These approaches could be grouped in two categories: methods that incorporate the uncertain parameter in the optimization formulation; and methods that represent the uncertainty via multiple realizations. Chance constrained optimization (e.g., Loughlin and Ranjithan, 1999) and robust optimization (e.g., Wagner and Gorelick, 1987) are examples of optimization techniques that incorporate uncertain parameter into the optimization model. Monte Carlo simulation (e.g., Aly and Pertalta, 1999) and noisy GAs (e.g., Samlley et al. 2000, and Chan and Culver, 2005) are examples of techniques that require multiple realizations (samples) of the uncertain parameter. Realization or sample is a set of random values of the uncertain parameters generated based on a probabilistic method or statistical distribution. Some other methods are available to address uncertainty where they are based on evaluating the solution, similar to a sensitivity analysis, after the optimization has been concluded (e.g., Gorelick, 1987).

The focus of this research is to investigate procedures that require multiple realizations, namely noisy genetic algorithms procedures. Noisy GAs are relatively new techniques within the evolutionary computing field with only limited work reported so far. Samelly et al. (2000) utilized standard noisy GA for remediation of contaminated groundwater under condition of uncertainty in the hydraulic conductivity field. In their work,

the authors used a maximum of 15 realizations (sample sets of hydraulic conductivity fields) to conduct the search. Along the same line, Gopalakrishnan et al. (2003) selected a noisy GA for optimal sampling intended for risk-based groundwater remediation design. The authors were able to identify successfully a reliable design using seven realizations. Chan and Culver (2005) developed a new noisy GA procedure for groundwater remediation design, where they introduced the concept of age and incorporated uncertainty into the optimization procedure instead of the objective function evaluation.

The standard noisy GA procedure utilizes a fairly large number of realizations to reduce the noise from fitness evaluations. Incorporating a large number of realizations in fitness evaluations could be computationally expensive. This research sought to develop an efficient noisy GA optimization procedure, referred to as the archived noisy GA (ANGA), and apply to groundwater inverse problems under noisy conditions. The solution approach used in the present research extends the work by Chan and Culver (2005), in which a groundwater problem is solved using a noisy GA-based procedure. The work reported in this paper focuses on solving inverse problems under uncertain condition using simulation-optimization approach. The method is designed to efficiently reduce the computation burden that is inherent in the standard noisy GA. The performance, based on solution quality and computational need, of the new archived noisy GA is compared with that of the standard noisy GA for an instance of the groundwater source identification problem.

5.2 Groundwater Characterization Problem

One of the challenges commonly encountered in groundwater health risk analysis studies and remediation strategies of contaminated sites is the contaminant source

characterization. Groundwater characterization problem is an instance of inverse problem where identifying possible contaminant source location(s) and/or contaminant concentration(s) are the major objective. The groundwater characterization problem definition and recommended solution approaches can be found in many references such as in Sun (1994), Poeter and Hill (1997), and Aral et al. (2001). In the groundwater field, uncertainty is a common factor that is always connected with the unknown aquifer parameters. This research utilizes a two-dimensional groundwater contaminant source characterization problem (CSI), where two-dimensional groundwater flow and transport models are employed to simulate the flow and the transport of the plume. The governing equations describing the groundwater flow and the groundwater transport are fully explained in many references (e.g., Bear, 1972).

The problem under study simulates an unknown contaminant release at a single contaminant source that is meant to be resolved from spatially and temporally distributed concentration observations collected at a set of pre-specified monitoring wells. The problem assumes that the contaminant source location and the initial contaminant release at the sources are unknown and are represented as continuous values. Concentration observations at the specified monitoring wells are collected to generate a concentration time-series at each monitoring location. For a possible contamination source characteristics (defined by the source location, size and initial contaminant release), the resulting concentrations at the monitoring wells are estimated using a groundwater transport model (forward model). The optimization method attempts to identify the source characteristics that yield concentration estimates that best match the observations. Figure 5-1 illustrates the groundwater source identification problem in a two-dimensional groundwater domain.

The problem is posed as a constrained continuous non-linear optimization model where the prediction error between the observed concentrations and the simulated concentrations is minimized; Equation 5-1 illustrates a typical minimization objective function. A number of constraints are typically included to enforce decision variable bounds and feasibility.

$$\underset{DV}{\text{minimize}}\left(\text{Prediction Error}\right) \tag{5-1}$$

where DV represents parameters or decision variables (e.g., x_1, y_1, x_2, y_2, C_0) used to characterize the source location and the initial concentration. Since the decision variables are continuous variables (real variables), unlimited number of possible combinations may exist for this problem.

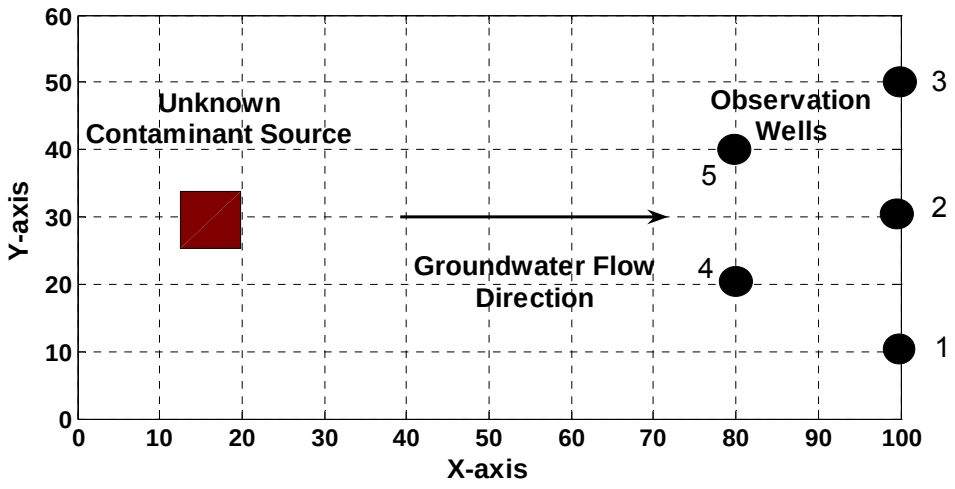

Figure 5-1. Hypothetical Two-Dimensional Groundwater Aquifer, Axes Units are Meter

This research investigated two instances of the groundwater contaminant source characterization problem representing different levels of complexity depending upon the nature of uncertainty. In the first case, the uncertainty is assumed to exist in the observed data

as a result of, for example, measurement error. Thus, a set of random realizations of the observed concentration profiles are used to represent that uncertainty. The second instance of the problem assumes uncertainty in the hydraulic conductivity values, and the predicted concentration profiles were generated using a set of hydraulic conductivity realizations.

5.3 Methodologies: Noisy Genetic Algorithms Optimization

5.3.1 Standard Noisy Genetic Algorithms

The noisy GA was developed in 1996 by Miller and Goldberg (1996); however, similar procedures were previously used for image registration by Grefenstette and Fitzpatrick (1985) and Fitzpatrick and Grefenstette (1988). In addition, noisy GAs were used in few applications such as watershed management problems by Harrell (2001) and groundwater remediation (Smalley et al., 2000; Gopalakrishnan et al., 2001; Aly and Peralta, 1999).

In general, the standard noisy GA (SNGA) is similar to simple GA except that it operates in a noisy environment and tends to evaluate the fitness function differently. In the SNGA procedure, a number of realizations of the uncertain variable are created in each generation, and the same set of realizations is used for every single individual throughout the given generation. A special fitness function called a sampling fitness function utilizes sampling to decrease the amount of noise from fitness evaluations in a noisy condition. The overall fitness of each individual is then determined depending upon the average of fitness values calculated using each of the multiple realizations as shown in Equation 5-2.

$$f_{i,nr} = \frac{1}{nr} \sum_{j=1}^{nr} f_{i,j} \tag{5-2}$$

where $f_{i,j}$ is the j^{th} noisy fitness evaluation of individual i calculated using realization j, nr is number of realizations and $f_{i,n}$ represents an approximation to the actual noisy fitness value. This can be closely equivalent to utilizing a Monte Carlo simulation using a small sample size in the GA fitness evaluation step. Subsequent studies reported that a limited number of realizations (i.e., relatively small sample size) is sufficient in the SNGA (Miller and Goldberg 1996; Gopalakrishnan et al. 2001). The challenge, however, is in determining the best number of sample size (n) for the sampling fitness evolution. Utilizing a larger sample size provides the procedure a more accurate solution quality, but requires additional computation effort to conduct a larger number of simulation model executions.

5.3.2 Archived Noisy Genetic Algorithms

Traditional noisy GA procedures utilize a fairly large number of realizations to reduce the noise from fitness evaluations. It is worth emphasizing that the main objective of this research is to develop a new method that could achieve a robust solution with a reduction in the computational burden by utilizing a fewer number of evaluations. In this approach, multiple realizations of the uncertain parameter are generated. Each generation of the algorithm uses a single realization. The best solution in each generation is then stored into a separate set named the "archive pool." The main steps of the algorithm are described as follows:

Step 1. Generate a set of realizations R_j, where j is the realization number *1, 2, …, n*.

Step 2. Initialize the basic GA by creating an initial population with population size (*p*) for the current generation number *(gn*), for this step *gn = 1*.

Step 3. Evaluate the fitness of the objective function. Only one realization is used per generation without replacement, i.e. a realization will be selected randomly for the realization set and used at most one time, and then discarded. The fitness ($f_{i,j}$) of each individual is determined based fitness value calculated using a single realizations, where $f_{i,j}$ is the jth noisy fitness evaluation of individual i calculated using realization j.

Step 4. The most fit solution, i.e., the best individual (BI), will be stored into a separate pool referred to as the archive pool (ArchP). The best individual will be stored into the ArchP unless an identical copy of the same individual is already stored into the pool.

Step 5. The basic GA operators (selection, crossover and mutation) are applied to the selection pool (SeleP) to generate new p_{gn} individuals for the current generation *gn*, for this step *gn=2*. The SeleP contains (p_{gn-1} + *ar*) individuals, where p_{gn-1} is the number of individuals in the previous population, and ar is the number of individuals in the ArchP.

Step 6. Repeat steps 3, 4, and 5 for *gn = 3, 4, 5, ..., until total number of generations or termination criterion is met*. Note that the selection pool size increases as new best individual added to the ArchP. Thus, the selection population size is not a constant as in the basic GA.

The solution at the end of the procedure is expected to converge to an optimal solution for the problem. This termination criterion could possibly depend upon the number of realizations, the sensitivity of the best solution, and/or the improvement of the solution.

Instead of utilizing only one realization per generation, multiple realizations in each generation, or for some number of generations, could be used to enhance the degree of

robustness of the solution. Thus, a modified procedure, referred to as the Enhanced Archived Noisy GA (EANGA), was also implemented and tested. These modifications are mainly applied to steps 3 and 4. In Step3, instead of one realization, a number of realizations (*nr*) could be used for a number of generations (*ng*). Consequently, the overall fitness of each individual is determined based on the average of fitness values calculated using each of the multiple realizations as shown in Equation 5-2. As a result of modifications in step 3, BI in Step 4 must be stored in the ArchP every *ng* generations.

5.4 Numerical Experiments

5.4.1 Illustrative Case Studies

Two instances of the groundwater characterization problem, described in Section 5.2, were considered in this work. In the first case, the uncertainty exists in the observed data while the second case of the problem assumes uncertainty in the hydraulic conductivity values. Two sets of realizations of the uncertain parameter were generated for each case study, where one was used in the optimization search procedure, and the other to test and compare the robustness of the solutions obtained using the different methods. The search set consists of 300 realizations while the test set consists of 1000 realizations.

In the first case, two sets of realizations of the observed concentration profiles for the observation wells were generated. The generation was executed by using a two-dimensional groundwater transport simulation model and Latin hypercube sampling (LHS) with normal distribution. The LHS is a form of sampling that can be applied to one or multiple variables. The LHS method is commonly used to reduce the number of samples necessary to achieve a reasonably accurate statistical distribution. The probabilistic or statistical technique was first

introduced and described by McKay (1979). It was later investigated and elaborated by Iman et al. (1981). Detailed introduction, computer codes, and manuals can be found in many publications (e.g., Iman et al., 1980). In LHS, variables are sampled using an even sampling method, and then randomly combined sets of those variables are used for one calculation of the target function. The sampling algorithm ensures that the distribution function is sampled evenly, but still with the same probability trend. Equation 5-3 illustrates how the observed concentration profiles are generated.

$$UC_{ijr}^{obs} = C_{ij}^{obs} \times R \tag{5-3}$$

where R is a normal distribution LHS generator with a mean of 1.0 and a standard deviation of 0.1, and C_{ij}^{obs} is the true observed concentration profiles at time step i at well j (these observations were synthetically generated using the forward model). UC_{ijr}^{obs} represents the uncertain observed concentration profiles generated using the groundwater forward model in combination with a probabilistic formula for n time steps recorded at the five monitoring wells (nw) located downstream. The values, UC_{ijr}^{obs}, resulted from Equation 5-3 follow the normal distribution LHS trend with a mean equal to C_{ij}^{obs} and standard deviation equal to (0.1 x C_{ij}^{obs}).

In the second case problem, two sets of realizations of the hydraulic conductivity field were generated using a conceptual model based on the geostatistical approach. The spatial distribution of hydraulic conductivity was constructed through random generation using the turning band method (Thompson et al., 1989). The statistical model for random generation is

a lognormal distribution with mean (μ) equal to 20 and standard deviation (σ) equal to 0.5 as shown in Equation 5-4.

$$K_r = \mu \times e^{\sigma * R_r} \tag{5-4}$$

where K_r is the spatial distribution of the hydraulic conductivity for realization (r), r is the realization index (r = *1, 2, ..., total number of realizations*), and R is a random generator obtained from turning band method for realization (r).

Table 5-1. Hypothetical Two-Dimensional Domain Parameters

Parameter	Source Identification Problem	
	Case 1	Case 2
Problem Size	100m x 60m, (51grid x 31grid)	
Grid Spacing (dx, dy)	2 m, 2 m	
Number of Time Steps	60	
Total Time	120 day	
Time Step Size (dt)	2 day	
Dispersion Parameters	α_L = 1 m, α_T = 1 m, D_m = 0.01 m^2/day	α_L = 2 m, α_T = 0.5 m, D_m = 0.01 m^2/day
Groundwater Field	Homogeneous	Heterogeneous
Average Velocity	1 m/day	Varies
Source of Uncertainty	Model Output Parameters, Observed Concentration	Model Input Parameter, Hydraulic Conductivity Field
No. of Realizations of the Observed Concentration Profile	300 and 1000	N/A
No. of Realization of the Hydraulic Conductivity Field	N/A	300 and 1000
Number of Observation Wells	Five	
True Source Location and Initial Concentration	[x_1=14, y_1=28, x_2=20, y_2=34, C_0=70]	

5.4.2 Simulator Settings

A two-dimensional (2D) hypothetical groundwater aquifer, one with a homogeneous velocity field and another with a heterogeneous velocity field, was considered. The 2D domain is of dimensions 500m x 300m, which is discretized into 1581 (51 x 31) finite element nodes. The simulation duration of 120 days was modeled using 60 time steps, and the concentrations were recorded at five monitoring wells located in the farthest down-gradient end of the aquifer (see Figure 5-1). The general groundwater flow direction was toward the right boundary; detailed geometrical and hydraulic parameters are shown in Table 5-1, and are depicted in Figure 5-1.

5.4.3 Optimization Model

The aim of the optimization models is to determine the best possible source characterization (i.e., identify the best source location and concentration) that satisfy the constraints while taking into account the uncertainty in the two cases (i.e., uncertainty in the observed data for the first case, and the uncertainty in the heterogeneous hydraulic conductivity in the second case).

In the first case, the uncertainty is assumed in the concentration profiles generated at the five observation wells. Equation 5-5 illustrates the objective function (normalized root squared error) that is minimized subjected to a number of constraints defined by Equations (5-6) – (5-8). The optimization model is mathematically described as below:

$$\underset{DV}{\text{minimize}} \ \frac{1}{nr}\sum_{r=1}^{nr}\sqrt{\frac{\sum_{j=1}^{nw}\sum_{i=1}^{n}\left(UC_{ijr}^{obs} - C_{ij}^{sim}\right)^2}{\sum_{j=1}^{nw}\sum_{i=1}^{n}\left(UC_{ijr}^{obj}\right)^2}} \tag{5-5}$$

subject to:

$$C_{min} \leq C_0 \leq C_{max} \tag{5-6}$$

$$x_{min} \leq x_k \leq x_{max}; \quad y_{min} \leq y_k \leq y_{max};, \quad k = 1,2 \tag{5-7}$$

$$x_1 \leq x_2; \quad y_1 \leq y_2; \tag{5-8}$$

where UC_{ij}^{obs} is the uncertain observed concentration profiles generated using the groundwater forward model in combination with a probabilistic formula for n time steps recorded at the five monitoring wells (*nw*) located downstream, $C_{ij}^{sim} = f$ *(contaminant source location, concentration)* where C_{ij}^{sim} is the time-series of the simulated monitoring concentrations, and *DV* represents parameters or decision variables, i.e., (x_1, y_1, x_2, y_2, C_0), used to characterize the source.

For the second case, the uncertainty is assumed to be in the heterogeneous hydraulic conductivity field that was used to obtain the simulated concentration profiles. Equation 5-9 illustrates a typical minimization objective function (normalized root squared error) subjected to the same constraints defined by Equations (5-6) – (5-8). The optimization model for the second case is mathematically illustrated as follows:

$$\underset{DV}{\text{minimize}} \ \frac{1}{nr}\sum_{r=1}^{nr}\sqrt{\frac{\sum_{j=1}^{nw}\sum_{i=1}^{n}\left(C_{ij}^{obs} - UC_{ijr}^{sim}\right)^2}{\sum_{j=1}^{nw}\sum_{i=1}^{n}\left(C_{ijr}^{obj}\right)^2}} \tag{5-9}$$

where $UC_{ijr}^{sim} = f$ *(contaminant source location, concentration)* where UC_{ij}^{sim} is the time-series of the uncertain simulated monitoring concentrations using various hydraulic conductivity realizations *r*, and *DV* represents parameters or decision variables, i.e., (x_1, y_1, x_2, y_2, C_0), used to characterize the source.

5.4.4 Setting for the Search Algorthims

The optimization procedures defined by the new noisy GA and the standard noisy GA were evaluated using the example problem to determine the efficiency of the optimization methods under noisy environment. The SNGA, ANGA, and the EANGA procedures were applied to the two instances of groundwater source characterization problem described above. Several trials were first conducted for each case to tweak and fine-tune the search parameters. The population size for all noisy GA procedures was set to 100, and the termination criterion was set to 100 generations or a negligible fitness error improvement for five consecutive generations (i.e., an objective function error less than 1.0E-09 is recorded for five consecutive generations). In the SNGA procedure, 20 different realizations were randomly selected in each generation to perform the search and to determine the fitness function (Eqn. 1). For the ANGA, only one realization was used in each generation to perform the search. This single realization was changed in each generation. In the EANGA procedure, for every five generations a set of different five realizations is used to perform the search. Other algorithmic parameter values and the boundary parameter values used each of the example problem are shown in Table 5-2 and Table 5-3, respectively. As GA-based procedures are probabilistic methods, a set of 30 random trials were conducted to evaluate the robustness of the procedure.

Table 5-2. Parameter Settings for the Example Problem

Parameter	Settings for the Optimization Procedures		
	SNGA	ANGA	EANGA
Population Size	100		
Decision Variables Type	Real		
Generation	100		
Selection	Tournament Selection		
Crossover	Heuristic Crossover, 80 %		
Mutation	Gaussian Mutation, 20 %		
Number of Realizations per Generation Used in Calculating the Fitness Function	20*	1*	5**

* Different set of realization(s) is used for each generation
** Different set of realizations is used every five generations

Table 5-3. Allowable Range of Decision Variables Values for both Problems

Variables	Ranges [Minimum-Maximum]
x axis	[0-100] m
y axis	[0-60] m
C_0	[0-100] mg/l

5.5 Results and Discussion

The efficiency of the new noisy GA methods is evaluated and then applied to the source identification problem using two forms of uncertainty conditions as described above. The performance of the new methods is evaluated against that of the standard noisy GA method. The results are demonstrated in the context of the convergence behavior, the quality of the solution obtained, and the computational cost including number of function evaluations utilized to conduct the simulation-optimization approach for each method.

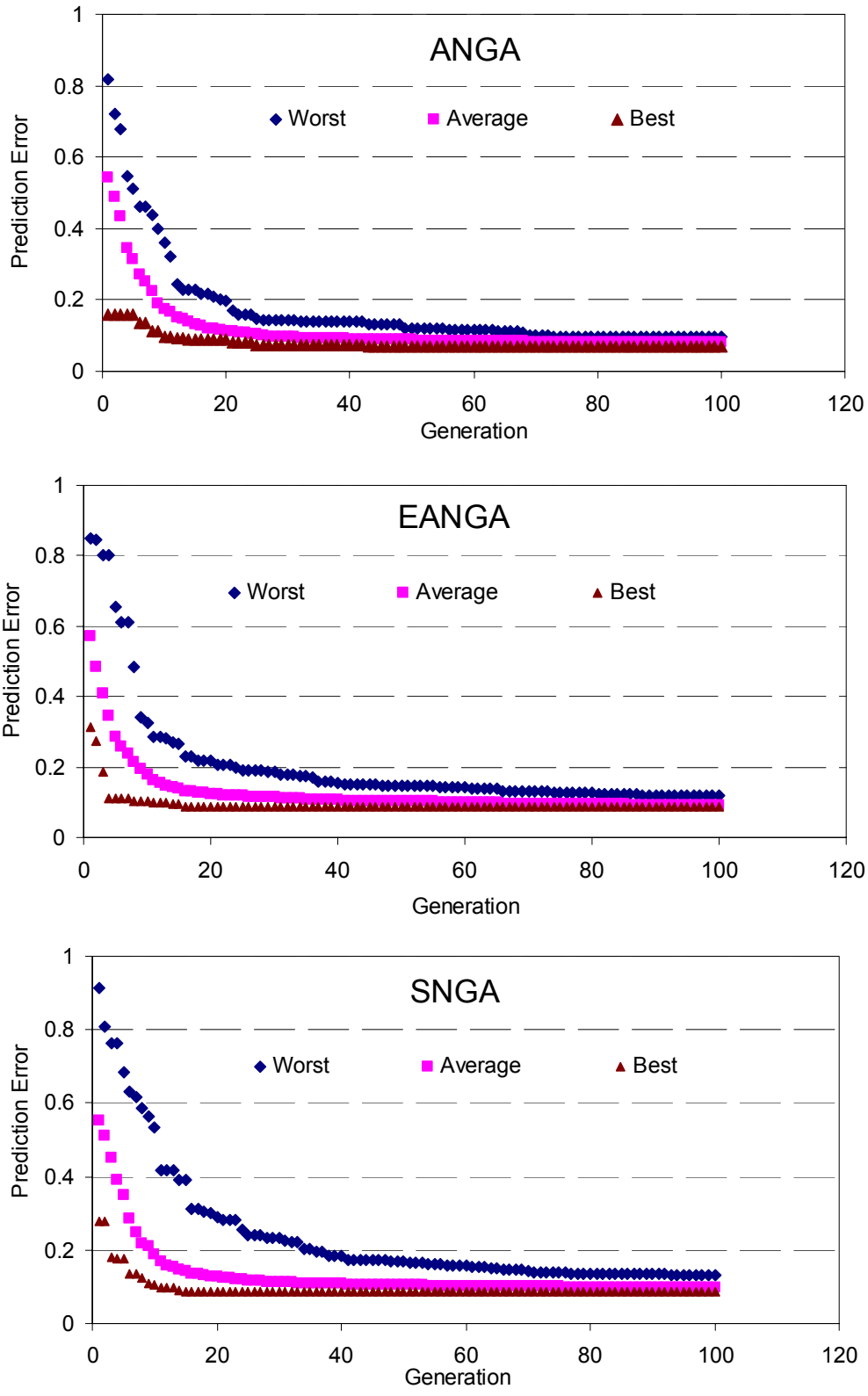

Figure 5-2. Variation of the Best, the Average and the Worst Values for Prediction Error over the 30 Random Trials for the Three Noisy GA-based Procedures Corresponding to Case Study 1

The typical convergence behavior of the three GA-based noisy methods for the two different cases is depicted in Figure 5-2. The results from the 30 random trials conducted for case 1 problem are used here to illustrate the typical behavior of these methods. Figure 5-2 shows the variation of the best, the average, and the worst objective function values (i.e., prediction errors) obtained at each generation over the 30 trials. This implies that all solutions, among the 30 random trials, at a particular generation number would have had objective function values within the best and the worst values at that generation number. The rate of change of the best values indicates the convergence of the search procedure over generation. The difference between the best and the worst values at each generation number indicates the degree of robustness of each search method; the smaller the difference, the higher the level of robustness. The difference between the worst and the best values at the end of the generations for ANGA, EANGA and SNGA are 0.029, 0.033, and 0.043, respectively, indicating better performance by the new archived GA procedures The convergence plots for all the three methods seem similar to the typical robust convergence behavior; however, the convergence of ANGA shows slightly better performance than the other two procedures.

In a post-processing step, the quality of the obtained solutions for each method is then evaluated using two performance metrics. The first metric is the prediction error, which is similar to the prediction error function (Eqn. 5-5 for case study 1 and Eqn. 5-9 for case study 2) except for using the test set realizations to generate the uncertain parameters and using the best found solution (i.e., the converged solution) to generate the simulated concentration. The second metric calculates the success rate of the obtained solutions (i.e., simulated concentration profiles generated using the best found solutions). The success rate calculates

the ratio of the number of time-steps with a certain error target (e.g., the number of time-steps with less than 10% of the maximum error among all time-steps), as illustrated in Equation 5-10 and Equation 5-11.

$$Error = \frac{\left| C_{ijr}^{obs} - C_{ijr}^{sim} \right| \times 100}{C_{ijr}^{obs}} \quad (5\text{-}10)$$

$$Sucess\ Rate\ (\%) = \frac{(No.\ of\ Error \leq Z\%) \times 100}{nr \times nw \times n} \quad (5\text{-}11)$$

where *nr* is the number of realizations, *nw* is the number of wells, *n* is the number of concentration profile time-steps, and *Z%* is the pre-specified acceptable error target. Smaller prediction error values and larger success rate percentages imply better solution quality. Also, smaller standard deviations and narrower boundary ranges, i.e., (maximum value – minimum value), indicate a robust performance.

5.5.1 Case Study 1

The uncertainty, as indicated previously, in this case study exists in the observed concentration profiles. The results demonstrated in this case are based on synthetically generated observations at fixed monitoring well locations using groundwater simulation and statistical models. Equation 5-5 illustrates how the simulation-optimization method minimizes the prediction error function. To test the quality and the robustness of the solution, the best found parameters (decision variables) were used to generate a simulated concentration profile. The concentration profile then will be matched with the observed concentration realizations (i.e., the test set realizations) to calculate a set of prediction errors

(Eqn. 5-5). Also the success rates were determined for each optimization procedure (Eqn. 5-11).

Table 5-4. The Worst, the Average and the Best Converged Prediction Error among 30 Trials Using the Test Set Utilizing the Noisy Procedures for Case Study 1

Archived Noisy GA Using the Test Set Realizations	Among 30 Trials			
	Maximum	Mean	Minimum	Std Deviation
Worst Prediction Error	0.152	0.128	0.116	0.007
Average Prediction Error	0.124	0.101	0.091	0.007
Best Prediction Error	0.095	0.072	0.060	0.007
Success Rate (less than 10% error)	78.06%	61.23%	46.75%	9.33%
Success Rate (less than 20% error)	96.64%	77.69%	58.97%	11.47%
Enhanced Archived Noisy GA Using the Test Set Realizations	Among 30 Trials			
	Maximum	Mean	Minimum	Std Deviation
Worst Prediction Error	0.142	0.123	0.116	0.006
Average Prediction Error	0.119	0.097	0.091	0.006
Best Prediction Error	0.092	0.068	0.060	0.007
Success Rate (less than 10% error)	79.94%	62.35%	48.01%	7.90%
Success Rate (less than 20% error)	96.64%	79.31%	59.23%	9.91%
Standard Noisy GA Using the Test Set Realizations	Among 30 Trials			
	Maximum	Mean	Minimum	Std Deviation
Worst Prediction Error	0.159	0.126	0.116	0.010
Average Prediction Error	0.132	0.100	0.090	0.011
Best Prediction Error	0.105	0.071	0.060	0.012
Success Rate (less than 10% error)	81.61%	60.52%	42.79%	10.17%
Success Rate (less than 20% error)	97.46%	76.43%	55.32%	11.75%

The prediction errors (best, average, and worst), and the success rates (10% and 20%) for all search procedures using the test set realizations are presented in Table 5-4. Table 5-4 also reflects the statistics for the 30 trials including maximum, mean, minimum, and standard deviation for all the methods. Figure 5-3 illustrates the mean of the prediction error for each method combined with the corresponding maximum and minimum values. Similarly, Figure

5-4 presents the mean of the success rate for each method along with the corresponding maximum and minimum.

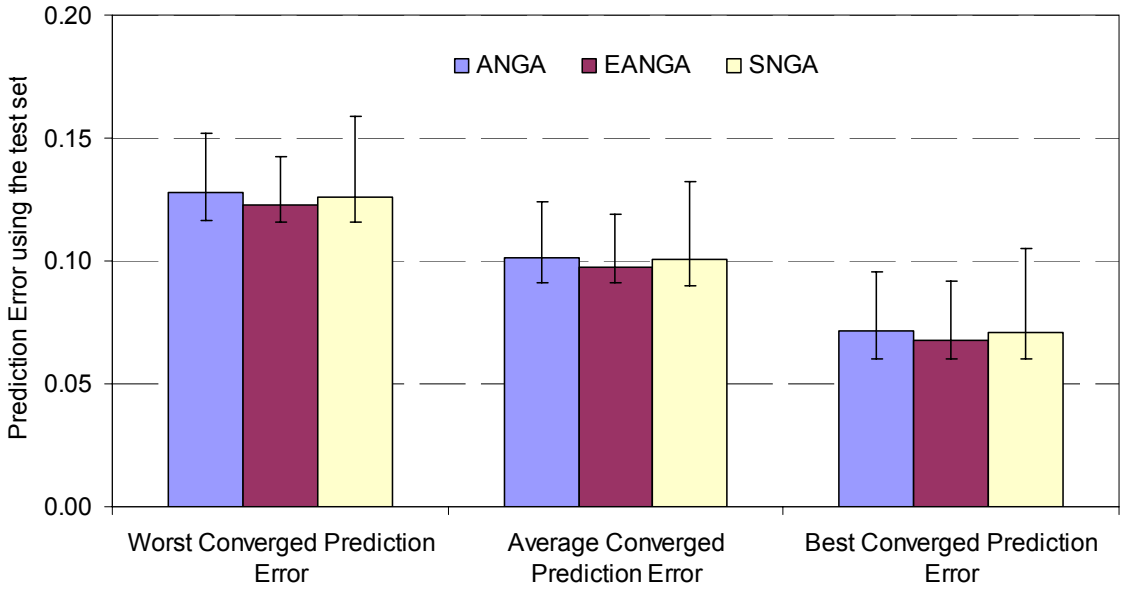

Figure 5-3. The Worst, the Average, and the Best Converged Prediction Error Associated with the Maximum and the Minimum Values over the 30 Random Trials for Case Study 1 Utilizing the Noisy Procedures

The results, presented in Table 5-4 and depicted in Figure 5-3 clearly indicate that the prediction error values and their ranges for all the applied methods are closely similar, with a slightly better numbers for the EANGA procedure. It is also obvious that the ANGA procedure has slightly shorter solutions ranges (maximum and minimum values) in comparison to the SNGA procedure. Data presented in Table 5-4 and depicted in Figure 5-4 show that the success rates recorded with their ranges for all the procedures are highly similar, with a slightly better value for the EANGA. Again, the ANGA procedure record rather shorter boundary ranges compared to the SNGA. Collectively, the results shown above indicate that the overall performance of the three procedures is comparable. The EANGA

method, however, has a little advantage over the rest. On the other hand, the number of function evaluations using the ANGA procedure is 20 times less than the number of function evaluations utilized by the SNGA procedure and it is five times less than the number of function evaluations used by the EANGA procedure. In addition, Figure 5-5 indicates the best source characterization solution obtained from among the 30 trials. Noticeably, the source characterization parameters for all the methods are very close to the true values except for the initial concentration values.

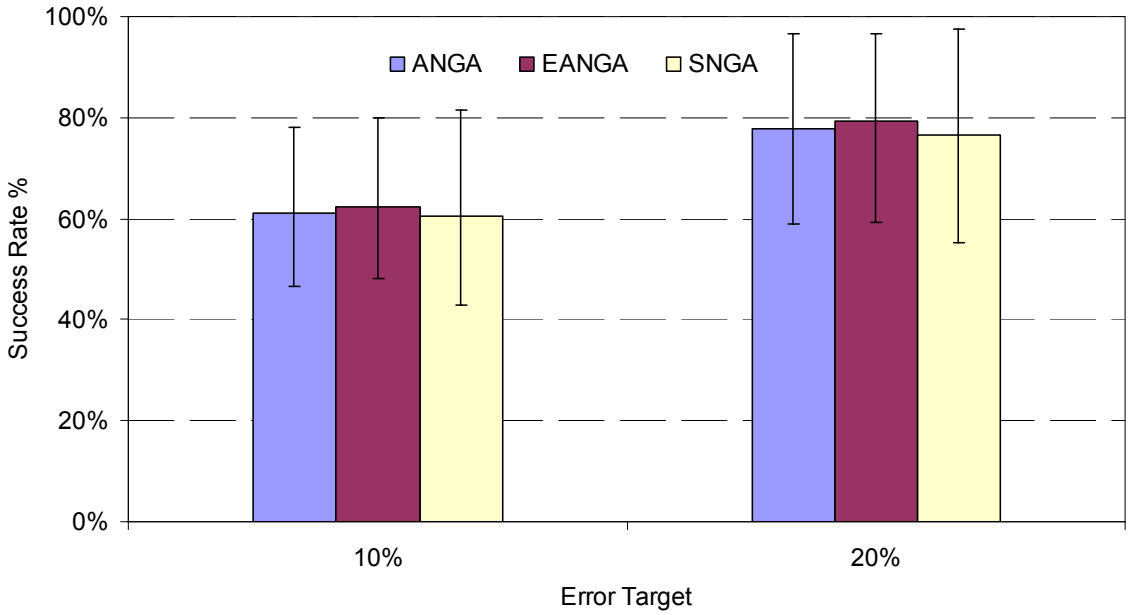

Figure 5-4. Success Rates Associated with the Maximum and the Minimum Values among 30 Trials Utilizing the Noisy Procedures for Case Study 1

5.5.2 Case Study 2

Unlike the first case study, the uncertainty in this study is assumed in the simulated concentration profile as a result of uncertainty in the hydraulic conductivity field. The results presented in this case study are from synthetically generated observations at predefined

monitoring wells using groundwater simulation. Equation 5-9 illustrates how the simulation-optimization method minimizes the prediction error function.

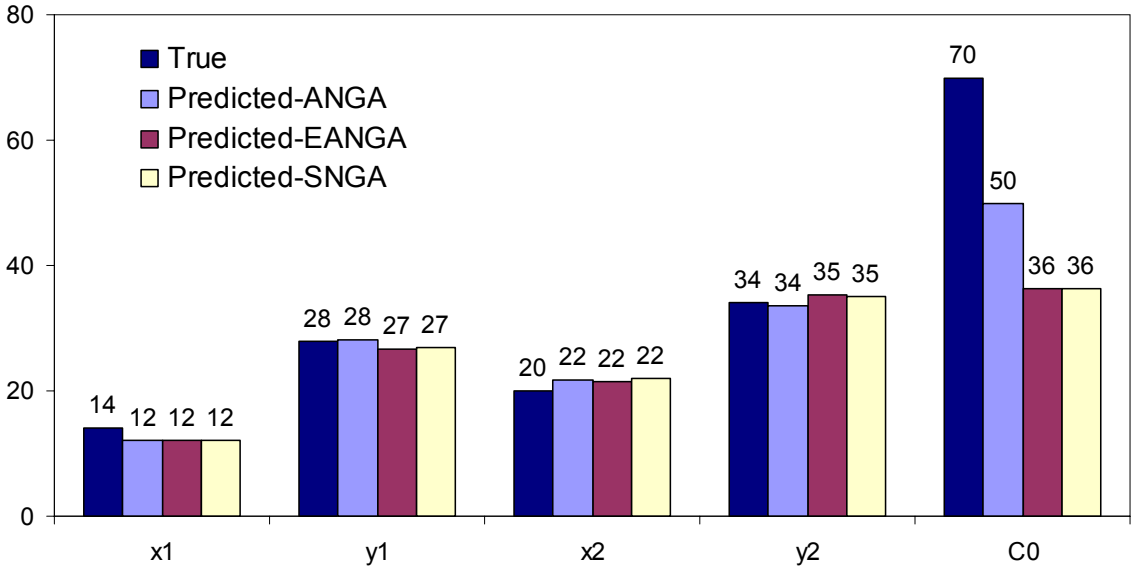

Figure 5-5. The Best Source Characterization Solution among 30 Trials Utilizing the Noisy Procedures for Case study 1

The results shown in Table 5-5 report the prediction errors (best, average, and worst), and the success rates (10% and 20%) for all the search procedures utilizing the test set realizations. Also Table 5-5 presents the statistics for the 30 trials including maximum, mean, minimum, and standard deviation for all the methods. The mean of the prediction error for each method combined with the corresponding maximum and minimum values are presented in Figure 5-6. Figure 5-7 presents the mean of the success rate for each method along with the corresponding maximum and minimum.

Table 5-5. The Worst, the Average, and the Best Converged Prediction Error among 30 Trials Using the Test Set Utilizing the Noisy Procedures for Case Study 2

Archived Noisy GA Using the Test Set Realizations	Among 30 Trials			
	Maximum	Mean	Minimum	Std Deviation
Worst Prediction Error	0.661	0.553	0.436	0.076
Average Prediction Error	0.297	0.228	0.201	0.035
Best Prediction Error	0.080	0.061	0.036	0.013
Success Rate (less than 10% error)	59.14%	56.52%	53.03%	2.23%
Success Rate (less than 20% error)	72.47%	69.04%	63.68%	2.95%
Enhanced Archived Noisy GA Using the Test Set Realizations	Among 30 Trials			
	Maximum	Mean	Minimum	Std Deviation
Worst Prediction Error	0.570	0.544	0.487	0.024
Average Prediction Error	0.242	0.209	0.196	0.012
Best Prediction Error	0.082	0.054	0.029	0.016
Success Rate (less than 10% error)	59.02%	55.65%	48.34%	3.28%
Success Rate (less than 20% error)	72.90%	69.03%	59.54%	4.02%
Standard Noisy GA Using the Test Set Realizations	Among 30 Trials			
	Maximum	Mean	Minimum	Std Deviation
Worst Prediction Error	0.671	0.563	0.413	0.082
Average Prediction Error	0.330	0.230	0.190	0.035
Best Prediction Error	0.083	0.064	0.033	0.014
Success Rate (less than 10% error)	61.14%	55.52%	52.03%	2.42%
Success Rate (less than 20% error)	74.47%	67.04%	61.68%	3.23%

The data presented in Table 5-5 and depicted in Figure 5-6, show that the prediction error values and their ranges for all the methods are very close with a slightly better numbers for the EANGA procedure. It is also clear that the EANGA procedure has slightly shorter solutions ranges in comparison to the other procedure. The results shown in Table 5-5 and depicted in Figure 5-7 demonstrate that the success rates recorded with their ranges for all the procedures follow a similar trend. The prediction error graph indicates that as well. Similar to case study 1, the results shown above prove that the overall performance of the three procedures is comparable. The EANGA method was identified to have a little advantage over the rest of the methods. While maintaining a close solution quality performance, the

computation effort used by the ANGA procedure is 20 times less than the computation effort used by the SNGA procedure and it is five times less than the computation effort utilized by the EANGA procedure. Figure 5-8 presents the best source characterization solution obtained from among the 30 trials. Unlike case study 1, the source characterization parameters for all the methods are slightly off in comparison to the true values with a limited advantage of the EANGA method. This may attribute to the higher level of complexity in the case study 2.

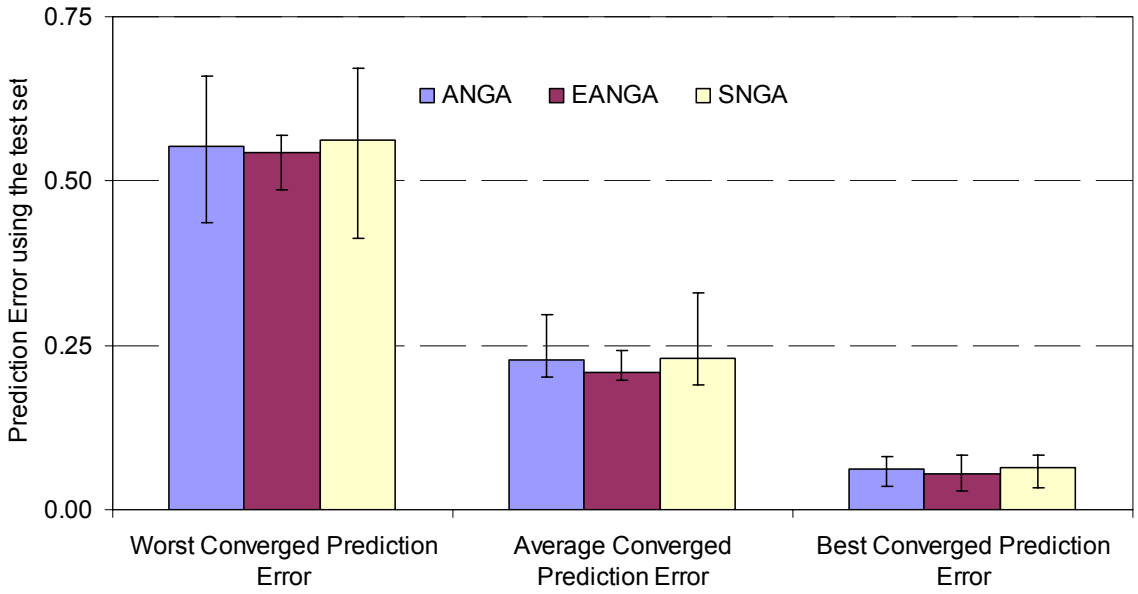

Figure 5-6. The Worst, the Average, and the Best Converged Prediction Error Associated with the Maximum and the Minimum Values over the 30 Random Trials for Case study 2 Utilizing the Noisy Procedures

5.6 Final Remarks

This work presented here explored the GA-based optimization procedures aiming to improve the efficiency of the search algorithms in solving inverse problem under uncertainty environments. The newly developed procedures, the archived noisy GA and its enhanced version EANGA, were evaluated against the standard noisy GA method for solving two-

dimensional groundwater contaminant source characterization problem. Two cases of the CSI problem were introduced, representing different uncertainty conditions, as illustrative applications. In the first case the uncertainty is assumed in the model output (observations reading), while in the second case the uncertainty exists in the model input parameters (hydraulic conductivity field). The comparison of the GA-based noisy approaches was conducted based on the convergence behavior, the solution quality, and the number of evolutions of each method. In the context of the solution quality, the overall results in both cases indicate that the three procedures reflect similar prediction errors. The EANGA procedure reflects better performance than the other two procedures in each of the case studies. Also, the results indicate that the ANGA procedure performs slightly better than the SNGA procedure. In terms of computational efficiency, the results demonstrate that the ANGA procedure is able to achieve 20 times fewer function evolutions than the SNGA procedure and five times fewer than the EANGA procedure. The best source characterization parameters for case study 1 were close to the true parameters, while the best source characterization parameters for case study 2 were slightly off. The results of this work illustrate that using the archived noisy GA approaches achieves reasonable performance improvement, indicating a promising approach. It is also observed that using the ANGA approaches need more investigations, and they could be conducted by building upon these preliminary results. Future work could consider a more complex application problem under different uncertainty conditions, could be efficiently explored, investigated, and resolved within a reasonable time frame and acceptable computational resources.

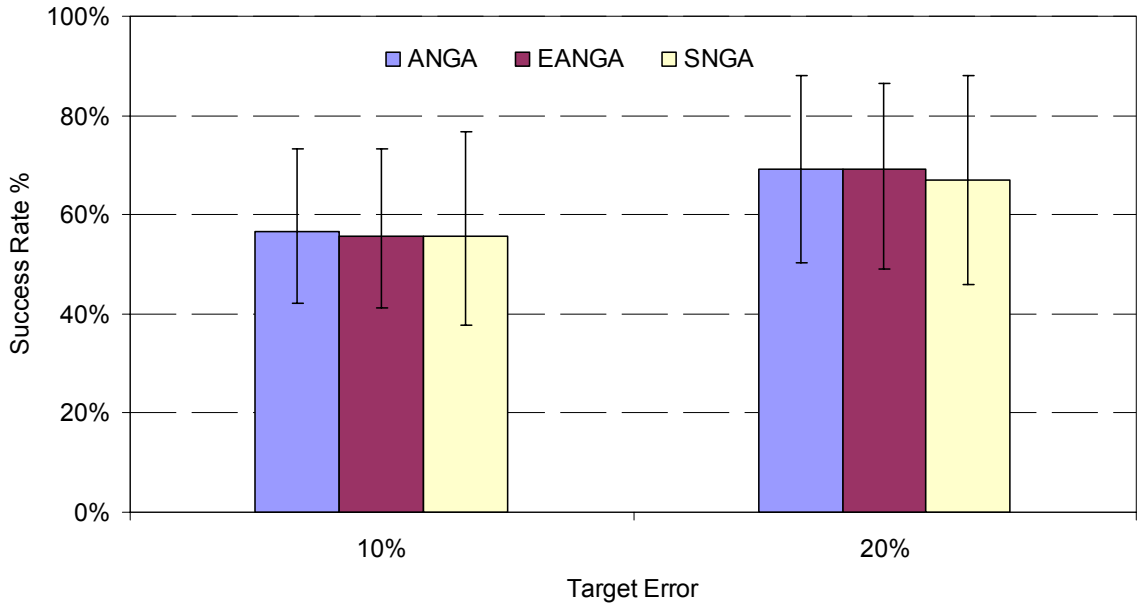

Figure 5-7. Success Rates Associated with the Maximum and the Minimum Values among 30 Trials Utilizing the Noisy Procedures for Case Study 2

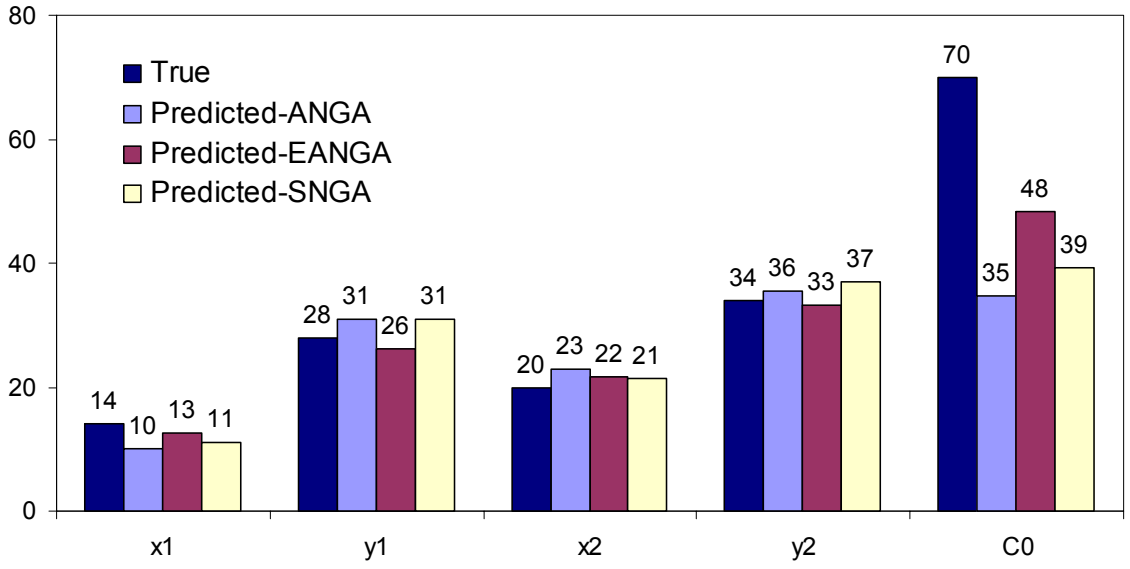

Figure 5-8. The Best Source Characterization Solution among 30 Trials Utilizing the Noisy Procedures for Case Study 2

CHAPTER 6

GRID-ENABLED SIMULATION-OPTIMIZATION FRAMEWORK PERFORMANCE AND EVALUATION

Abstract

Many engineering and environmental problems that involve the determination of unknown system characteristics from observation data can be categorized as inverse problems. A common approach undertaken to solve such problems is the simulation-optimization approach where simulation models are coupled with optimization or search method. Simulation-optimization approaches, particularly in environmental characterization, are computationally expensive due to the complex three dimensional simulation models and the large number of such simulations involved. Emerging grid computing environments (e.g., TeraGrid) show promise for improving the computational tractability of these approaches. However, harnessing grid resources for most computational applications is a non-trivial problem due to the complex hierarchy of heterogeneous and geographically distributed resources involved in a grid. The paper reports and discusses the development and evaluation of a grid-enabled simulation-optimization framework for solving environmental characterization problems. The framework is designed in a modular fashion that simplifies coupling with simulation model executables, allowing application of simulation optimization approaches across problem domains. The framework architecture utilizes standard communications protocols and the latest message passing interface with an application programming interface (MPI2-API) to establish a connection between a centralized search application and forward models running on TeraGrid resources. The performance of the

framework was first tested and benchmarked against an existing early developed MPI based implementation. Sets of performance and scalability results utilizing a groundwater source history reconstruction problem are presented. In addition, the quality solution of the groundwater application is also reported.

6.1 Introduction

Inverse problems occur often in engineering and science applications where the unknown values of some model parameter(s) are required using observed data. In general, inverse problems are problems where the results or sequences are known, but not the cause. Many instances of inverse problems are present in real-world applications. For example, in groundwater contamination characterization problems, the unknown contamination location and/or the release profile are estimated based on observation data. Similarly, in drinking water distribution system security management, monitoring data from a contamination sensor network is used to determine the contamination location and characteristics so that appropriate evasive actions could be quickly determined. Another application is in medical imaging where the exact properties of internal organs are assessed via non-invasive screening and diagnosis.

Inverse problems are typically ill-posed as opposed to the well-posed problems. The three conditions for a well-posed problem suggested by Jacques Hadamard are: existence, uniqueness, and stability of the solution or solutions (Maz'ya and Shaposhnikova, 1998). Due to ill-posedness, inverse problems are inherently difficult to solve and computationally expensive compared to the equivalent forward problems. In this approach, the simulation model is coupled loosely/tightly with optimization techniques in order to determine the

model inputs that best approximate the observed data. This process is applied iteratively until some stopping criteria are met; a comprehensive description about this approach is available in Andradóttir (1998) and Azadivar (1999).

Computational grid is a technology that enables the clustering of a broad variety of computing resources, such as supercomputers and storage systems, and allows them to be used as a single resource. Grid computing could be defined as the process of breaking down a computational problem and then distributing its components among a number of computational resources (Foster and Kesselman, 1999). Computational grids promote reliable and economical access to high-end computing resources, continuing to emerge as a new paradigm in scientific and engineering computation. The emergence of computational grids has created new possibilities for solving engineering and/or environmental inverse problems.

Recent investments in national high speed network infrastructure have allowed the aggregation of geographically distributed high-performance computing resources into computational grids. In part, because computational grids promote reliable and economical access to, and sharing of, high-end computing resources, they have emerged as a new paradigm in scientific and engineering computation. Given the computational resource demands of the simulation-optimization solution approach, computational grids have the potential to enable the solution of inverse problems that previously would not have been possible.

While researches have been actively developing strategies for adapting many different types of applications to computational grids, this study specifically focuses on grid-enabled simulation-optimization applications. Example of a related work is the grid-enabled

simulation-optimization framework developed by Parashar et al. (2004). The framework is applied for solving oil reservoir management problems using a data-driven approach. The dynamic, data-driven systems framework consists of a several autonomic components that discover and interact with one another on a peer to peer basis to solve the optimization problem. Another related work is the development of a grid architecture for engineering optimization and design search documented in (Cox et al., 2001; Gang et al., 2004). They jointly developed a grid-enabled simulation-optimization framework using open standard communications technologies and have demonstrated applications in computational fluid dynamics.

The essential difference between the engineering optimization and the simulation-optimization applications reviewed in this study appears to be the communications substrate used to facilitate communications between services running on distributed grid resources. This choice is critical, as it dictates the interactions between framework components with the potential of needlessly complicating the design and implementation of the framework components. Economical application framework using a practical approach is adopted for the solution of environmental inverse problems using heuristic search algorithms.

This paper documents the developments, evaluation, and performance of a grid-enabled framework for solving environmental characterization problems, namely, source history reconstruction problem (SHR) in the groundwater domain. Essential aspects of the SHR problems and the simulation-optimization solution approach were described briefly. An important requirement guiding the development of the framework is the scaling efficiency. This paper documents a battery of performance tests used to study the limitations of the current framework architecture, and provides guidance for future development efforts.

6.2 Computational Framework Architecture

Simulation-optimization is a general term used to describe a family of optimization techniques which utilizes simulation models for the evaluation of objective and constraint functions. A wide variety of applications lend themselves to simulation-optimization techniques, including engineering design optimization, optimization of stochastic systems, model calibration, and solution of inverse problems. This study is built upon the LAarge Scale Simulation-Optimization framework (LASSO), (Tryby et al., 2005). The framework consists of a centralized optimization application that utilizes a master-worker or client-server task distribution strategy. The optimization, master, and worker processes are executed on grid-based computational resources. The worker processes interface with instances of the forward model for distributed task execution, the worker consists of an adaptor and a simulator. Results are returned to the master for processing by the centralized optimization application; Figure 6-1 illustrates an overview of the framework. In the following sections, key components of the application architecture are described in greater detail.

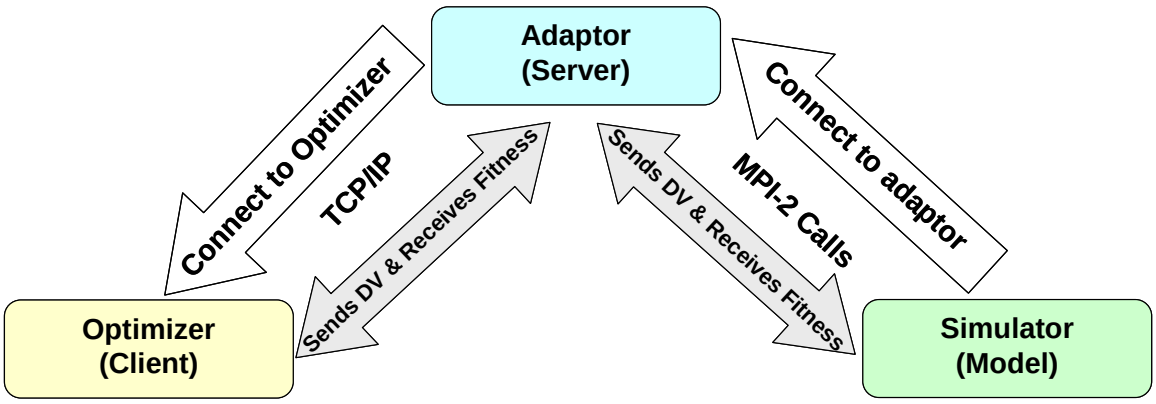

Figure 6-1. The Framework Overview

6.2.1 Simulation Model -- Groundwater Transport Model

The simulation model used in this study is the Parallel Groundwater transport and REMediation codes (PGREM3D), a suite of massively parallel codes used for numerical simulation of three-dimensional groundwater transport and remediation problems (Mahinthakumar, 1999). The flow module solves the steady state groundwater flow equation and the transport module solves multi-component reactive contaminant transport (Bear 1979). Detailed description of the equations describing these modules is provided by (Mahinthakumar, 1999). In this study, the flow module is run once to generate the flow field and only the transport module is run iteratively in the simulation-optimization application.

6.2.1.1 Numerical Methodology

The numerical methodology used in the flow and the transport modules is the Galerkin finite element method with eight-node linear hexahedral elements. A logically rectangular grid structure is assumed but irregular geometries are supported using distorted elements. A Crank-Nicolson approximation is employed for the time derivative terms. A lumped mass formulation (Huyakorn and Pinder, 1983) is used for all time-derivative and non-derivative (zeroth spatial derivative) terms. The coupled non-linear system is solved using a modified form of the Sequential Iterative Algorithm (SIA). Several Krylov subspace iterative solvers are implemented in the code for the matrix solution (Mahinthakumar et al., 1997).

6.2.1.2 Parallel Implementation

The simulator codes are written in FORTRAN using double-precision arithmetic. The codes are parallelized using a two-dimensional domain decomposition in the x and y

directions (Mahinthakumar and Saied 2002). Explicit message passing interface library (MPI) is utilized to exchange information between these domains. The MPI provides portability to a wide range of parallel architecture, where the use of the MPI communicators is a convenient way to couple processes. The simulation model is implemented to accommodate fine, semi-coarse and coarse grained parallelism. Fine grained parallelism is applied by executing the simulation model on multiple computer processors via the MPI communication library. The semi-coarse grained parallelism is implemented using the MPI Group library, in which multiple instances of the simulation model are executed simultaneously in groups. The coarse grained parallelism is implemented by using multiple servers executed simultaneously, where each server consists of a number of groups. This technique is illustrated below considering that a given total number of processors are available. Each server consists of a number of processors equal to [total number of processors/number of servers]. In each server one processor P_0^{SM} acts as a server master to establish communication between the server and the simulator. The remaining number of processors [number of processors in a server-1] will be divided into number of groups. Thus, in each server, each group consists of a number of processors equal to [(number of processors in a server-1) /(number of servers)]. Within each group one processor acts as a group master P_0^{GM} to establish interprocessor communication with P_0^{SM}. Two separate processors are needed for the server master and the simulator master assuming that the server and the simulator can be executed in separate sites. A schematic of the simulation model parallelization with three level of parallelization is shown in Figure 6-2. More details on the use of the MPI can be found in many references (e.g., Gropp et al., 1999).

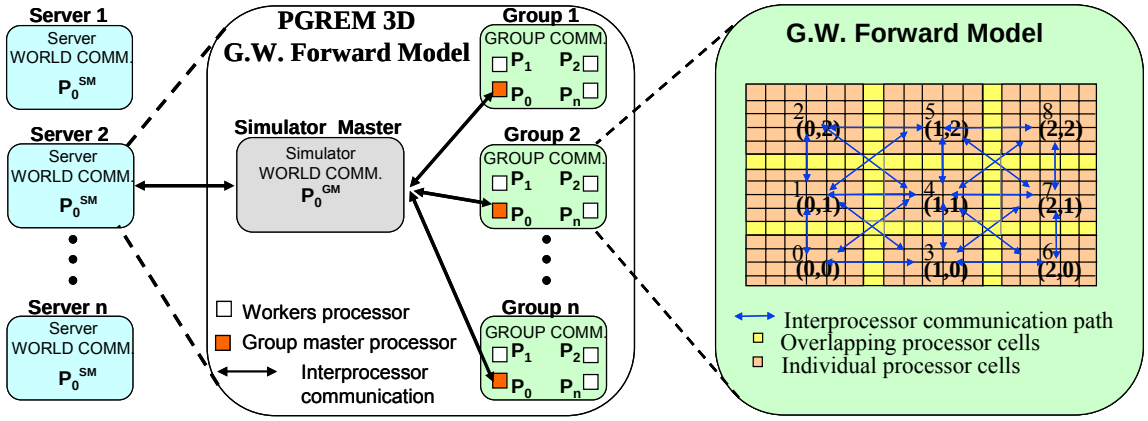

Figure 6-2. A schematic of the Simulation Model Parallelization

6.2.2 Optimization Approach – Genetic Algorithms

Several different search procedures have been implemented in Java including GAs and EAs and local search methods, making up the centralized optimization application. In this case, the optimization model representation of the inverse problem is solved using genetic algorithms (GA). Genetic algorithms are search procedures based on the natural selection and reduction. The GA was developed in the 1960s by John Holland and since then was used to solve different optimization problems. The GA starts with a set of potential solutions (population) that describes a potential solution to the inverse problem, usually these solutions are randomly selected. The performance of each solution is characterized by a fitness value. The objective function (representing prediction error) of the optimization model is used to quantify a fitness value indicating how well an individual solves the inverse problem. Fitness values are calculated using the results of forward model evaluations. During the GA search process, the population is iteratively subjected to several probabilistic operators that are analogous to natural selection, mating (or genetic recombination), and mutation. Each iteration through these steps forms a generation. Because fitter solutions are more likely to be selected for mating, the incidence of good traits of a solution tends to

increase from one generation to the next. Crossover assists to sample these traits in many different combinations. Typically, the quality of solutions quickly improves at the onset of the algorithm. As the population converges, improvements diminish. Usually, termination criteria are used to stop the procedure, such as total number of generation is met or improvements are sufficiently small that additional computations are not necessary. The application of a GA-based search to inverse problems is advantageous because of their robustness and global search characteristics. Some drawbacks include the computational intensity of a typical GA search and the slow final convergence prior to termination. However, GA is agreeable for use in a parallel and/or distributed computing environment since the fitness evaluation for each individual in a generation can proceed independently, (Goldberg, 1989; Davis, 1991).

6.2.3 Adaptor

An adaptor code, developed in C language, utilizes to facilitate a connection between the optimizer and the simulator code. Another objective of the adaptor is to maintain the loose coupling between the optimizer and the simulator code. The adaptor sets up and maintains socket connections with a task pool client written in Java (the optimizer) and a legacy MPI executable written in FORTRAN (the simulator) and passes data between the two codes. At first a connection is established between the simulator and the adaptor using MPI-2 calls such as publish, lookup, connect and accept. Then the adaptor connected to the optimizer using a TCP/IP socket connection. After the connection is established between the optimizer and the simulator via the adaptor, tasks (decision variables) and results (fitness) will be sent and received through the adaptor, see Figure 6-1.

6.2.4 Framework Parallelism

The optimization application is coupled with the forward model (PGREM3D) for fitness evaluation. Each forward model evaluation is handled via a number of processors (typically 1-8) called number of processors per group. The MPI is used to group processors, associate processors to computational domains, and for fine grained message passing within each of these groups. The solution procedure adopted in this study involves three levels of parallelism: one level is exhibited by the search procedure and the other two levels are exhibited by the forward model. Each one of the search procedure iteration exhibits a coarse grained parallel structure that requires an uncoupled forward model evaluation for each individual in the population. The optimization application acts as the master process in the master-worker task distribution strategy. The master, worker, and task pool used in the framework were designed and implemented as part of Vitri (Baugh, 2003). The master maintains a pool of remote tasks − a bundle of individuals requiring evaluation. Aggregating individuals in this manner reduces communications overhead. Worker processes running on distributed grid resources, having established a TCP-IP socket connection with the master, signal their readiness and draw tasks from the task pool (Tryby et al., 2005). The worker process transfers the remote task to the MPI zeroth processor. From this point forward, standard MPI group communications are utilized. The forward model manages multiple MPI groups, and each group evaluates an individual in the task bundle. The results of these simulations are then aggregated into a result bundle and returned to the optimization application for processing by the search algorithm. Finally, the next generation of the search is initiated.

6.3 Illustrative Case Study

An illustration case study is introduced in this juncture in a demonstration to measure the performance of the computation framework and to evaluate the efficiency of the solution approach for an inverse problem. The case study includes a description of a groundwater inverse problem and the simulation model settings for such application. Also the computation infrastructural and the resources utilized in this research are presented.

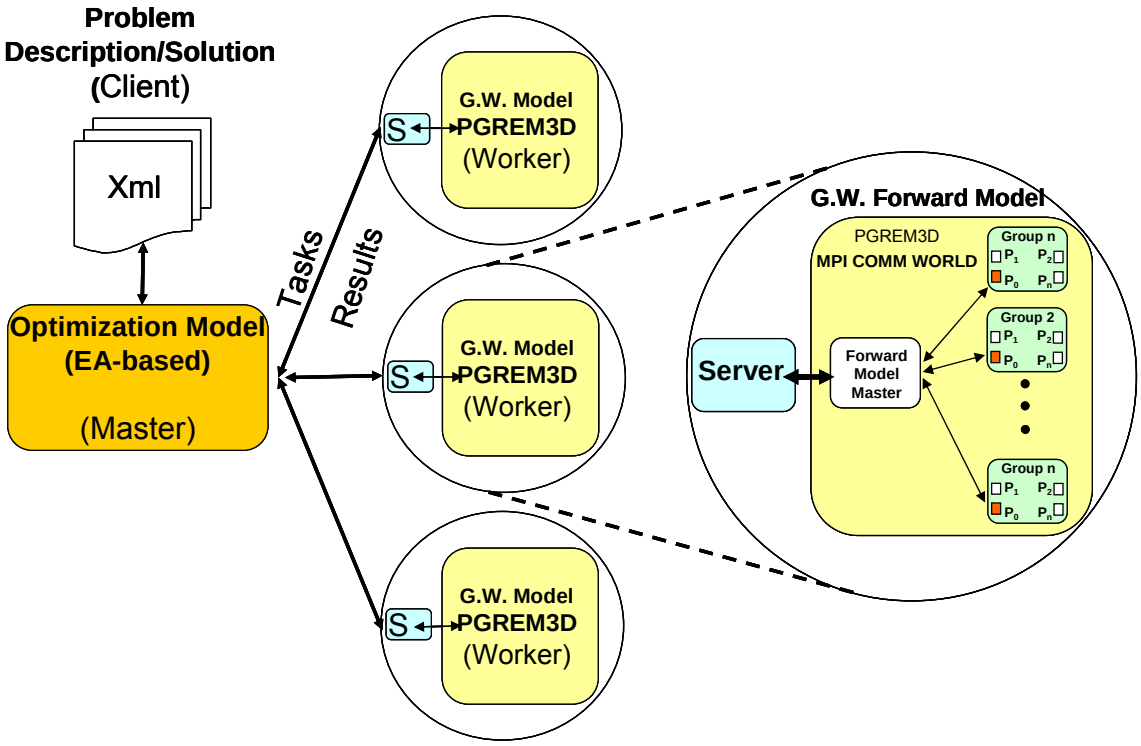

Figure 6-3. A Schematic of Simulation Model Parallelization

6.3.1 Application - Source History Reconstruction Problem

The study employs a three-dimensional groundwater source history reconstruction problem in the aim of conducting the performance analysis, whereas the three-dimensional groundwater transport model (PGREM3D) is employed to address the problem. Reconstructing the groundwater releases history from observational concentration

measurements is an inverse problem. Solving this problem is important in environmental forensics, where responsible parties are identified and incurred liabilities and/or responsibilities are legally enforced. Generally, in real-world problems, the contaminants source locations are known, but the contaminants release time histories are unknown.

In this application, unknown historical contaminant release schedules at given source locations are resolved from spatially and temporally distributed concentration data observed at a set of monitoring wells. The problem assumes that the contaminant source locations are known, but that the contaminant release histories at those sources are unknown. Observed concentrations at a number of monitoring well points are collected periodically (see Figure 6-4) to generate a concentration time series at each monitoring location. A general concentration time series is parameterized as follows;

$$\mathbf{C} = [c(0), c(\Delta t), c(2\Delta t), ...] \tag{6-1}$$

where c (M/L^3) is a concentration and Δt is a periodic time interval. The goal is to reconstruct the source release history over a finite time horizon t_r extending from the time (t_{s0}) in the past when monitoring activities started and towards the present; this time span t_r is referred to as the release history reconstruction period, with a periodic interval Δt_r. Monitoring activities are conducted at n_m monitoring wells. The number of samples taken at monitoring wells over the release history reconstruction period is equal to $(t_r/\Delta t_m)n_m$, where Δt_m is the periodic monitoring interval. In the general case, t_0 and t_{s0} do not necessarily correspond; however, for the problem being studied here it is assumed that sampling activities started at t_0.

The observed concentrations are modeled using the PGREM3D simulation, which provides the relationship $C_m = f(C_r)$, where C_r is the source release concentration time series

and C_m is the time series of modeled monitoring concentrations. The objective is to identify C_r given C_m, thus the solution of inverse problem is necessary. The inverse problem is posed as an optimization model where the objective function is the minimization of the root square error (RSE) between the observed and the modeled concentrations in respect to C_r;

$$\underset{C_r}{\text{minimize}} \sqrt{\sum_{j=1}^{n_m} \sum_{i=1}^{n_s} (\mathbf{C}_{ij}^{o} - \mathbf{C}_{ij}^{m})^2} \tag{6-2}$$

where the concentration time series C^o_{ij}, C^m_{ij} are observed and modeled concentrations, respectively, $i = 1, \ldots, n_s$ is the readings (time steps) recorded at a monitoring well and $j = 1, \ldots, n_m$ is the index of monitoring well locations. The time series of released concentrations C_r are the set of decision variables over which the problem is solved. The following constraints are added to enforce non-negativity for the decision variables.

subject to:

$$\mathbf{C}_{\max} \geq \mathbf{C}_r \geq 0, \tag{6-3}$$

where C_{max} is the maximum expected concentration. Depending on the number of contaminant sources, the number of decision variables is equal to the product of the number of contaminant sources times the number of time durations. The source history reconstruction was formulated using a single contaminant source with 18 monitoring wells sampled at a frequency, Δt_m, equal to 10 times the simulation model time step. Thus, a total of 1800 observations are used corresponding to 18 wells and 100 periodic samples. The release history reconstruction period, t_r, was equal to the simulation duration, and the reconstruction

period frequency, Δt_r, was equal to 100 times the simulation model time step. Thus, there were 10 decision variables in the optimization problem.

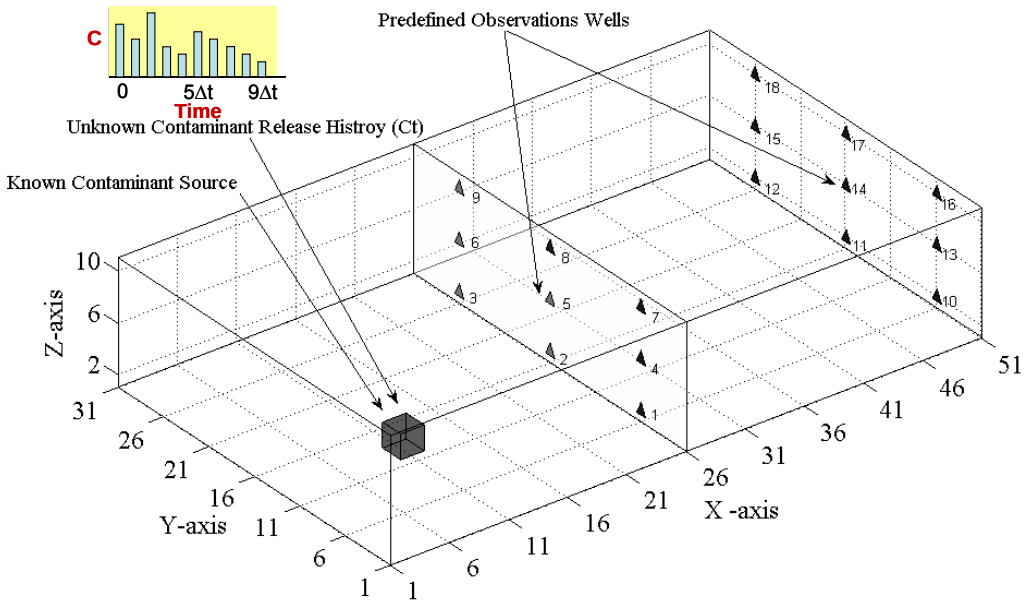

Figure 6-4. Hypothetical Three-Dimensional Domain

6.3.2 Simulation Model Settings

A hypothetical three-dimensional groundwater domain with heterogeneous velocity fields was considered. The 3D domain is of size 500m x 300m x 10m, the grid resolution for the simulation model resulted in 17,391 (51x31x11) finite element nodes within the groundwater domain. The simulation duration was for 400 time steps (8000 days) using 18 monitoring wells distributed evenly (9 wells each) in the middle (centre) and farthest end (perimeter) of the field (Figure 6-4). The steady state flow is computed using randomly heterogeneous hydraulic conductivity field generated using a turning band code (Thompson et al., 1999). The executing time using a single node at TeraGrid-NCSA cluster for the homogenous setting takes around 20 seconds. Detailed geometrical and hydraulic parameters are presented in Table 6-1, and depicted in Figure 6-4. Utilizing the mentioned simulation

setup, examples of the concentration profiles are shown in Figure 6-5 for monitoring wells 5 and 14.

Table 6-1. Hypothetical 3-Dimentional Domain Parameters

Parameter	Values
Problem Size	500m x 300 m x 10 m
Grid Size	51x31x11 grid
Number of Time Steps	150
Time Step Size (dt)	100 day
Grid Spacing	dx=10m, dy=10m, dz=1 m
Dispersion Parameters	α_L=10m, α_{TH}=5m, α_{TV}=1m
Flow Field	Heterogeneous
Velocity	Variable in Space
Executing Time	≈ 19 sec
Source Location	(x_1= 8, y_1=9, z_1=3, x_2=10, y_2=11, z_2=5)

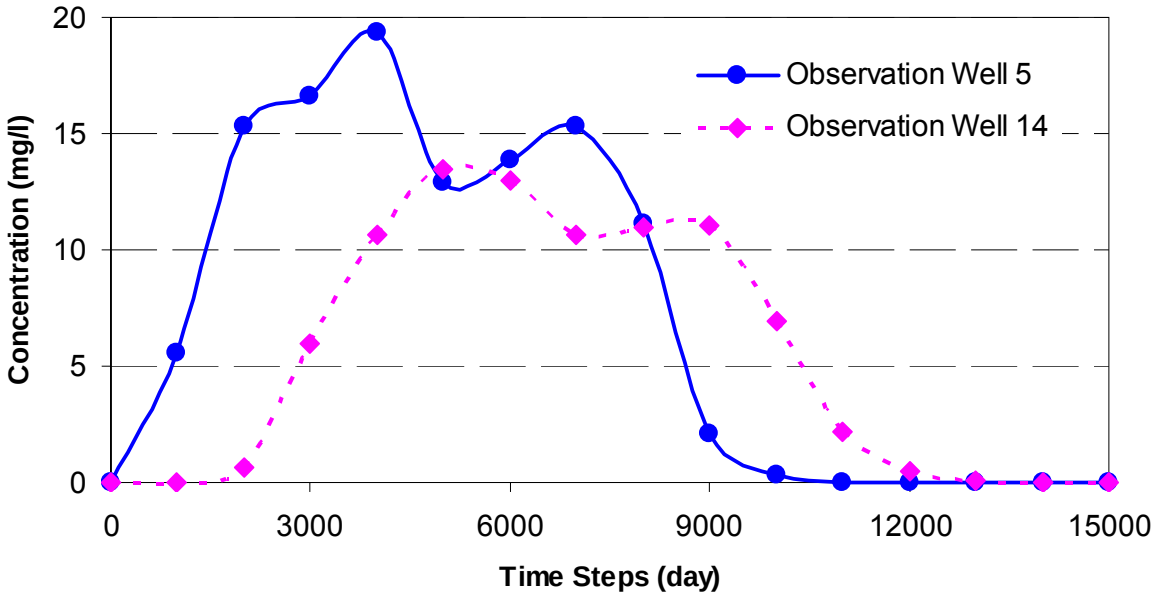

Figure 6-5. Concentration Profiles Monitored at Observation Wells 5 and 14

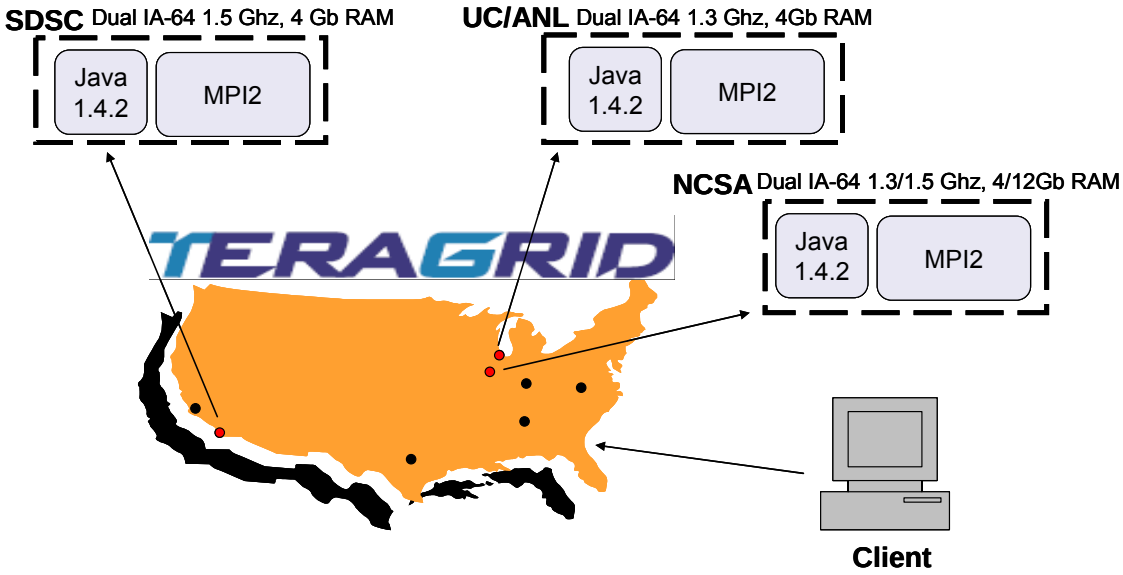

Figure 6-6. The TeraGrid Testbed Used

6.3.3 Computational Framework Infrastructure and Resources

The National Science Foundation (NSF) TeraGrid is a distributed computing infrastructure providing peta scale data collection, analysis, and storage; and tera scale computational capabilities. The TeraGrid is a heterogeneous agglomeration of computational resources distributed across the United States connected through a high speed network. The Distributed Terascale Facility (DTF) was funded in 2001 anchored by four locations, NCSA, SDAC, UC/ANL, and Caltech. In 2002 the existing TeraScale Computing System at PSC was combined with the DTF, and thus the TeraGrid was born; currently TeraGrid consists of nine sites (Catlett, 2002; 2005). The TeraGrid has a specialized interconnection network, referred to as the backplane, designed for high bandwidth data transfer. The TeraGrid backplane is designed for peak loads and can better accommodate data bursts between sites (Reed, 2003). For the present study, the computational experiments were performed on three IA-64 Intel Itanium2 clusters (1.5 Ghz) available at NCSA, SDSC, UC/ANL, on the TeraGrid (see Figure 6-6). All clusters are running SuSE Linux and are using Myricom's

Myrinet cluster interconnecting network, and the general parallel file system (GPFS). A summary of the resources dedicated to computationally intensive tasks available on those three sites are shown in Table 6-2.

Table 6-2. TeraGrid Resources Used

Site	CPU Type	CPU Speed, GHz	No. Nodes	No. CPUs	Peak Performance TFlops
NCSA	IA-64 Intel Itanium2	1.5	631	1262	6.0
SDSC	IA-64 Intel Itanium2	1.5	262	524	3.1
UC/ANL	IA-64 Intel Itanium2	1.5	62	124	0.61

6.4 Preliminary Evaluation

The development of the framework passes through many progress stages. Preliminary evaluation was conducted during early stages of the framework development to measure the performance of the in-progress LASSO implementation and to identify performance bottlenecks necessary to be addressed in the ongoing development. The above-mentioned release history reconstruction problem is the target application used for the performance analysis, and it is solved here using genetic algorithms.

Sayeed and Mahinthakumar (2002; 2005) developed an Efficient Parallel Optimization (EPO) framework implemented in FORTRAN-MPI for solving groundwater source characterization problems exhibiting multilevel parallelism similar to those solved in this paper. The software design approach adopted, however, resulted in tight coupling between the search algorithm and the simulation model. This resulted in a highly efficient implementation, but one that makes it difficult to couple with other simulation models. This study strives for a generic search framework which is loosely coupled with the simulation

model, yet efficient. The EPO implementation is used in this work as a performance reference. The intention was not an outright performance comparison between Java and FORTRAN MPI, since Java is anticipated to be slower in the first place. Rather, this work attempts to compare the two implementations and to gauge performance improvements in LASSO relative to conventional high performance benchmark. A number of conclusions and sound suggestions were provided aiming to improve the performance of the framework including (1) the communications overhead contributes significantly to evaluation times especially when the numbers of processors was greater than or equal to 32, (2) file I/O between the execution servers and their simulation executables make model evaluations more costly, and (3) the ratio of Java/FORTRAN wall times in test study ranged from 1.2 to 3.2. Detailed results and observations are cited in Mirghani et al. (2005a).

6.5 Results and Discussion

Considering the computational burden inherent in the inverse problems and the solution approach attempted in this paper, enhancement of the computational performance of the framework is required. Sets of runs were conducted on single sites and cross site to measure the performance and evaluate the current simulation-optimization framework. Another set of runs were used to demonstrate the quality solution of the groundwater source history reconstruction problem.

6.5.1 Application Solution Results

To address this problem the parallel simulation model was coupled with a standard GA to generate modeled (simulated) data. The GA searches for source concentrations in

order to generate a concentration profile (modeled profile) that matches the concentration profile (observed profile) recorded at 18 observation wells using the forward model. Since the true solutions of these hypothetical SHR problem are known in advance, the quality of the solutions obtained by the GA can be evaluated based on how well they predict the true solution. A solution error metric, shown in Equation 6-4 below, is used to measure the quality of the solution.

$$Solution\ Error = \frac{1}{n}\left(\sum_{i=1}^{n} \frac{\left| p_i^{true} - p_i^{pred} \right|}{p_i^{true}} \right) \tag{6-4}$$

where p_i^{true} is the observed value of parameter i (i.e., decision variables), p_i^{pred} is the predicted value of parameter i, and n is the number of parameters.

Table 6-3. GA Settings Used for the SHR Problem

Parameter	GA Setting
Selection Operator	Tournament Selection
Population Size	128
Population Type	Double
Number of Generation	100
Crossover Probability	0.70
Mutation Probability	0.2
Decision Variables Range (C_t)	0-100

Several trials were first conducted to estimate the appropriate GA parameters, such as population size, number of generation, crossover type and probability, and mutation rate. The

GA parameters and the boundary parameters used for the problem scenarios are shown in Table 6-3.

As GA is a probabilistic method, the results for the SHR problem obtained from 30 GA trials, in each trial a total of 20000 evaluations (population size set to 200 and executed for 100 generations) were used to conduct the GA search. Figure 6-7 shows the best predicted concentrations found compared to the true ones. Typical examples of the observed concentration profile compared to the modeled (predicted) concentration profiles utilizing observation wells 5 are shown in Figure 6-8. Clearly, the results prove that the GA was able to closely estimate the parameters with objective function error equal to 2.103 and solution error equal to 0.247%.

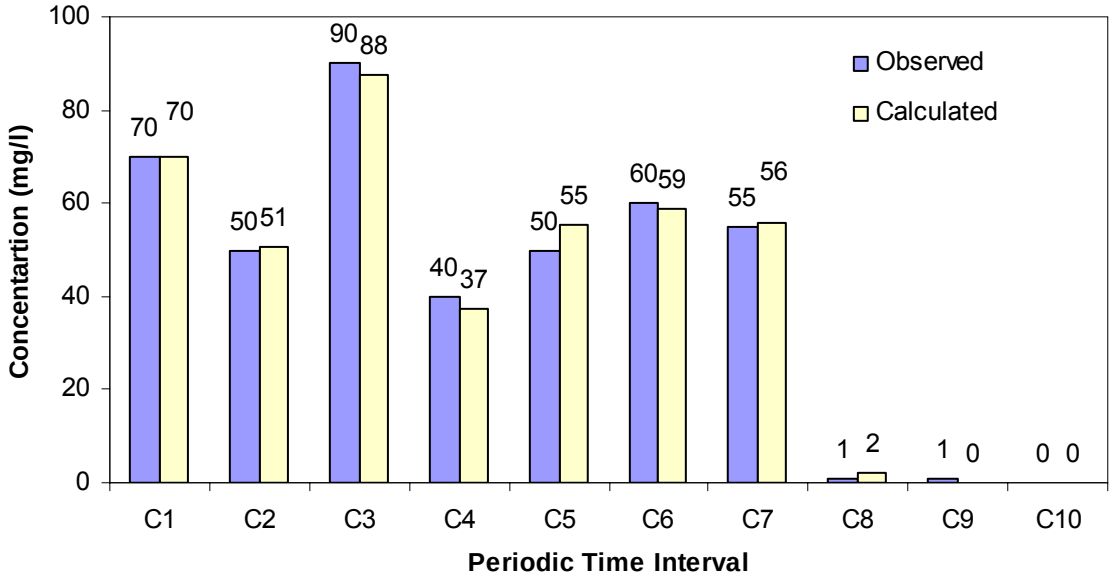

Figure 6-7. True and Predicted Concentration for the SHR Problem

6.5.2 Framework Performance Results

The performance analyses were carried out on single site (i.e., conducted separately on TeraGrid-NCSA, TeraGrid-UC or TeraGrid-SDSC) and cross site (i.e., conducted jointly

on TeraGrid-NCSA, TeraGrid-UC and TeraGrid-SDSC) platforms. Studies were conducted to investigate fine grained parallelism, semi-coarse grained parallelism, coarse grained parallelism, the chunk size effect, and overall scaling behavior. The simulation model settings described in Table 6-2 are utilized in these analyses to solve the groundwater source history reconstruction problem discussed in Section 6.3.1 above. The number of forward model evaluations performed during a search is a function of GA population size and the random seed. The population was set to 128 and was allowed to advance for five generations before times were recorded, while the random seed remains constant. Thus, the theoretical number of forward model evaluations for the experiment was 640. The actual number of evaluations performed, however, was 459. The difference in numbers is attributed to the reason that the framework eliminates duplicate model evaluations from occurring between generations. Out of three trials, the reported results are the best recorded results.

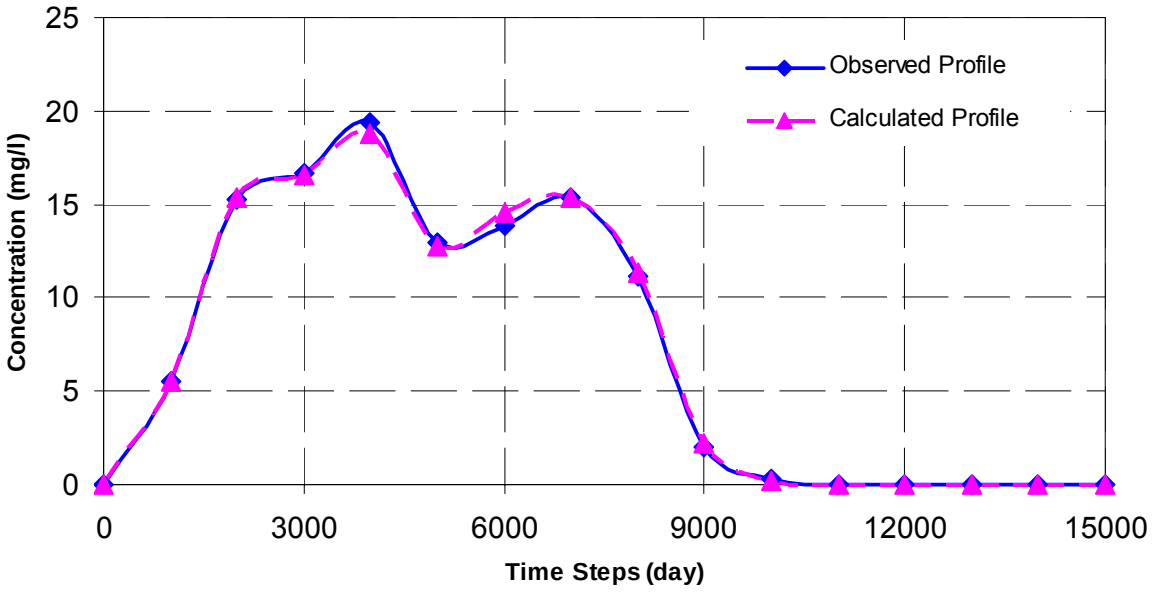

Figure 6-8. Predicted (Modeled) and True (Observed) Concentration Profiles Monitored at Observation Wells 5

6.5.2.1 Single Site Runs

6.5.2.1.1 Fine Grained Parallelism

One set of runs was used to investigate the effect of fine grained parallelism within the simulator model on the overall computation time in the three TeraGrid sites separately. In this study, the wall time corresponding to increasing number of processors per group was determined while number of groups, number of evaluations and number of tasks remained constant. Figure 6-9 illustrates that normalized evaluation time (i.e., total wall time divided by number of evaluations) recorded at NCSA site decreases until four processors per group, and then increases thereafter. While normalized time recorded on UC and SDSC increase as number of processors increases. That is because improvement in fine grained parallelism is associated with the problem size. As the size of the problem used in this study is relatively small, the improvement remains limited. Also, since the computation time is relatively small the overhead associated with the optimizer and the communication between the simulator and optimizer starts to dominate at higher processor counts.

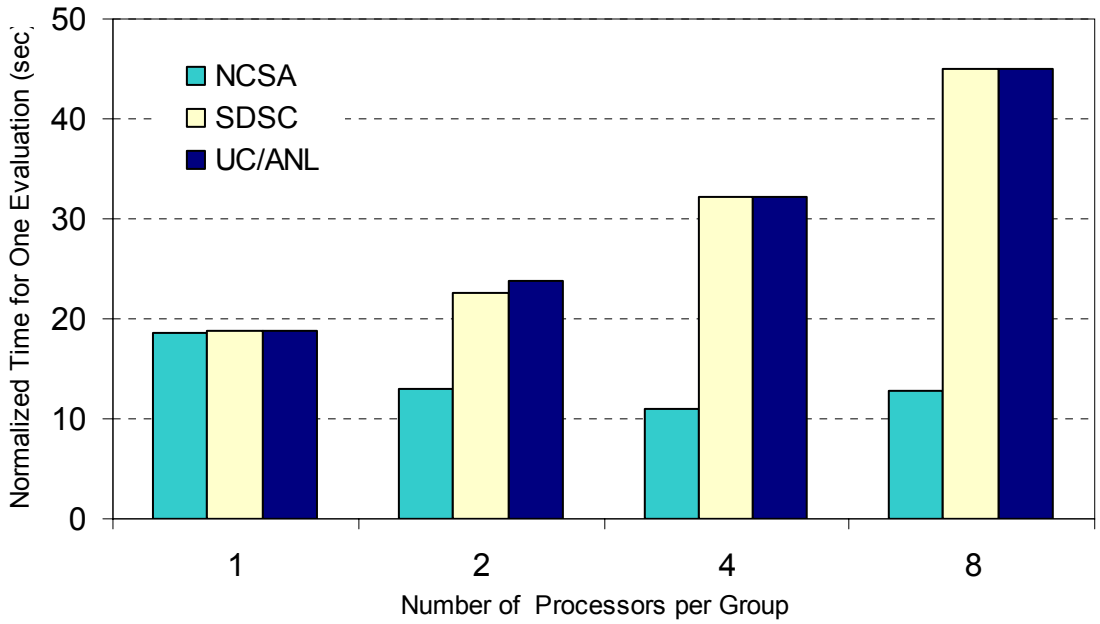

Figure 6-9. Fine Grained Parallelism, Number of Groups =1, Task/Group=1:1, Population Size = 128

6.5.2.1.2 Semi-Coarse Grained Parallelism

The second set of runs was used to investigate semi-coarse grained parallelism. The evaluation time is determined while the number of groups is increased. Other parameters such as number of evaluations, number of processors per group, and number of tasks per group were kept unchanged. Theoretically, the application is expected to scale linearly with the number of processors used. Speedup results shown in Figure 6-10 indicate that the framework scales almost identical to the theoretical scale until 32 groups, and then scales slightly sub-linearly when the number of group exceeds 64. This could be attributed to the effect of communication that plays an important role in increasing the wall time when more than 32 groups are used.

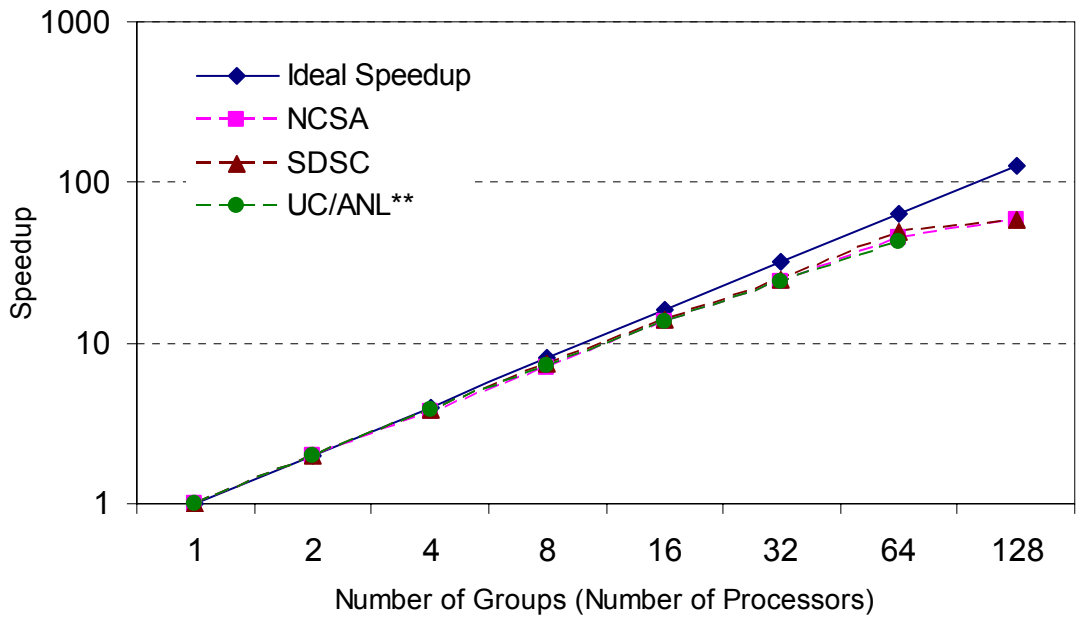

Figure 6-10. Semi-Coarse Grained Parallelism Speedup, Procs/Group =1:1, Tasks/Group= 1:1, Population Size = 128, ** Resources to Run 128 Groups are Not Available at UC/ANL

6.5.2.1.3 Coarse Grained Parallelism

The third set of runs was conducted to investigate the coarse grained parallelism. In this run the evaluation time was determined while the number of servers is increased. Other parameters such as number of evaluations, number of processors per group, number of groups per server, and number of tasks per group were still remain unchanged. Figure 6-11 illustrates the normalized evaluation time corresponding to number of servers utilized on the three sites. In theory, the application should scale linearly with number of servers used. For a fair comparison between Teragrid sites, a percentage normalized time was determined (i.e., the percentage of normalized wall time divided by normalized wall time for a single Server) for each site and compared against the ideal time. The results depicted in Figure 6-12 indicate that the framework scales almost identical to the theoretical ideal time until 16 servers. When the number of group exceeds 32, the framework scales slightly sub-linearly. As previously

mentioned the communication effect could be the cause for the increasing wall time when more than 32 servers are used.

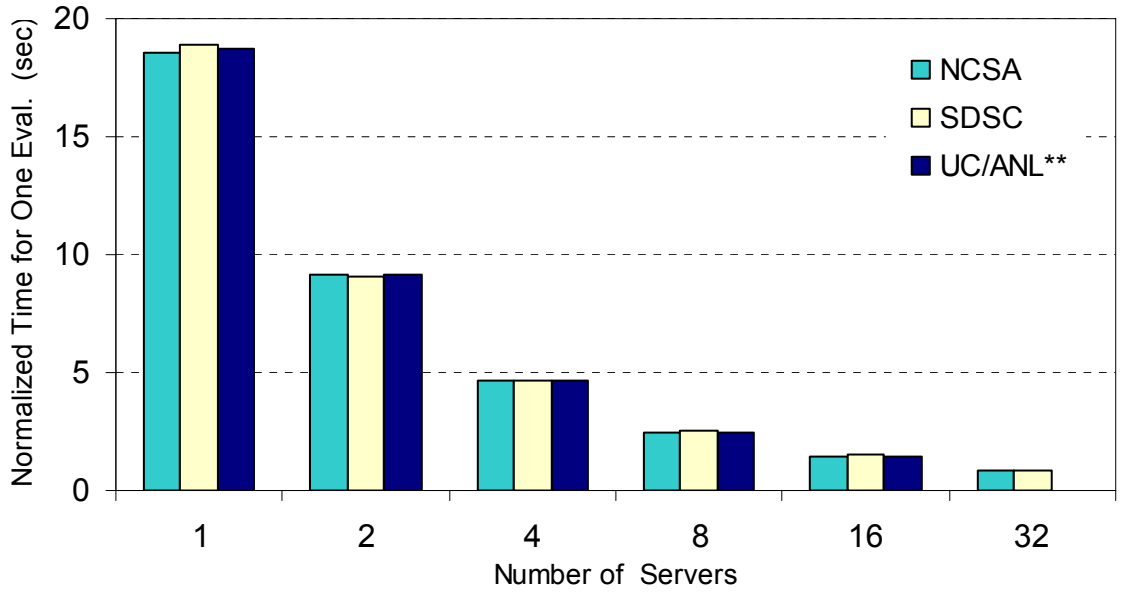

Figure 6-11. Coarse Grained Parallelism, Groups/Server = 1:1, Tasks/Group = 1:1, Population size = 128, **Resources to Run 32 Servers are Not Available at UC/ANL

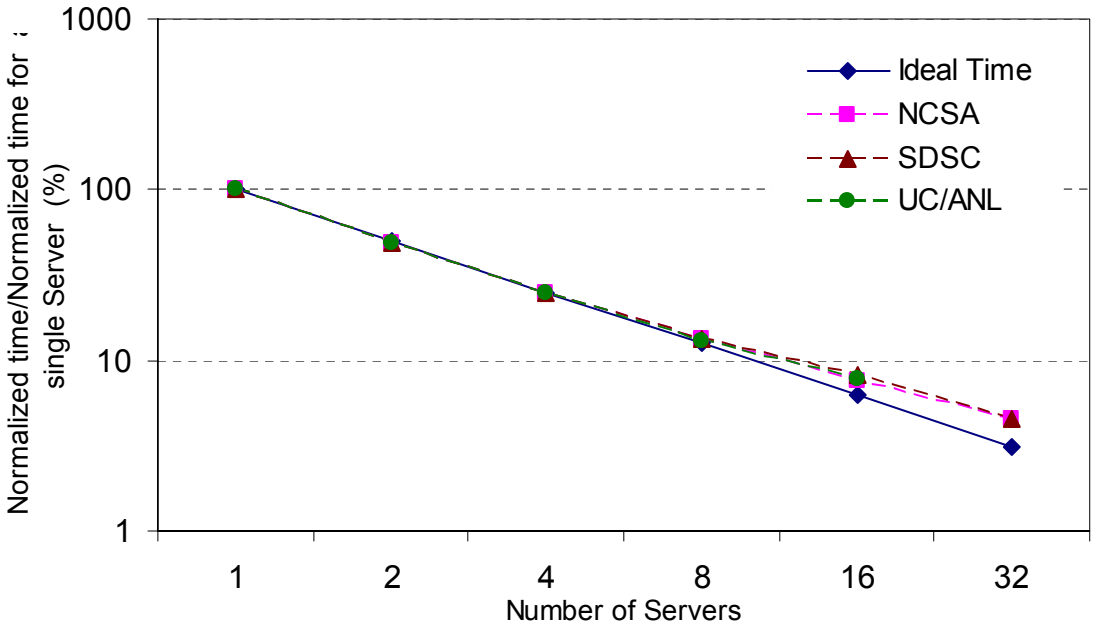

Figure 6-12. Coarse Grained Parallelism, Groups/Server = 1:1, Tasks/Group = 1:1, Population Size = 128, ** Resources to Run 32 Servers are Not Available at UC/ANL

6.5.2.1.4 Effect of the Chunk Size

The fourth set of runs was targeted to investigate the chunk size effect; i.e., number of tasks sent by the optimizer (the client) in one sweep. In those runs we observed the normalized evaluation time with increasing chunk size (or decreasing number of messages) at each site. Parameters including the number of groups and number of processors per group were kept constant. The results indicate that the normalized time for one evaluation decreases as the chunk size increases at the beginning then starts to increase as the chunk size continues to increase. Figure 6-13 shows only the results obtained on NCSA site; however, the result from SDSC and UC follow similar patterns (see Appendix B-I). A possible interpretation for this behavior is for small chunk sizes (i.e., less than 8), the latency dominates the evaluation time, while for large chunk sizes (i.e., greater than 32) the bandwidth and load imbalance dominates the evaluation time.

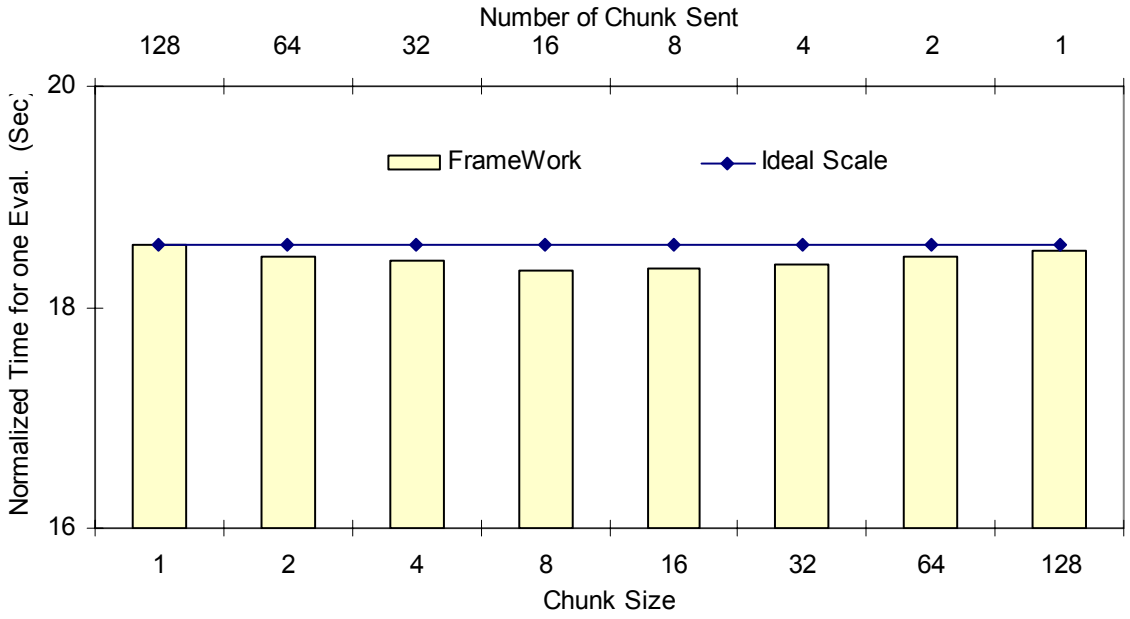

Figure 6-13. The Effect of Chunk Size Utilizing NCSA Site, Number of Groups = 1, Population Size = 128, Procs/Group =1:1

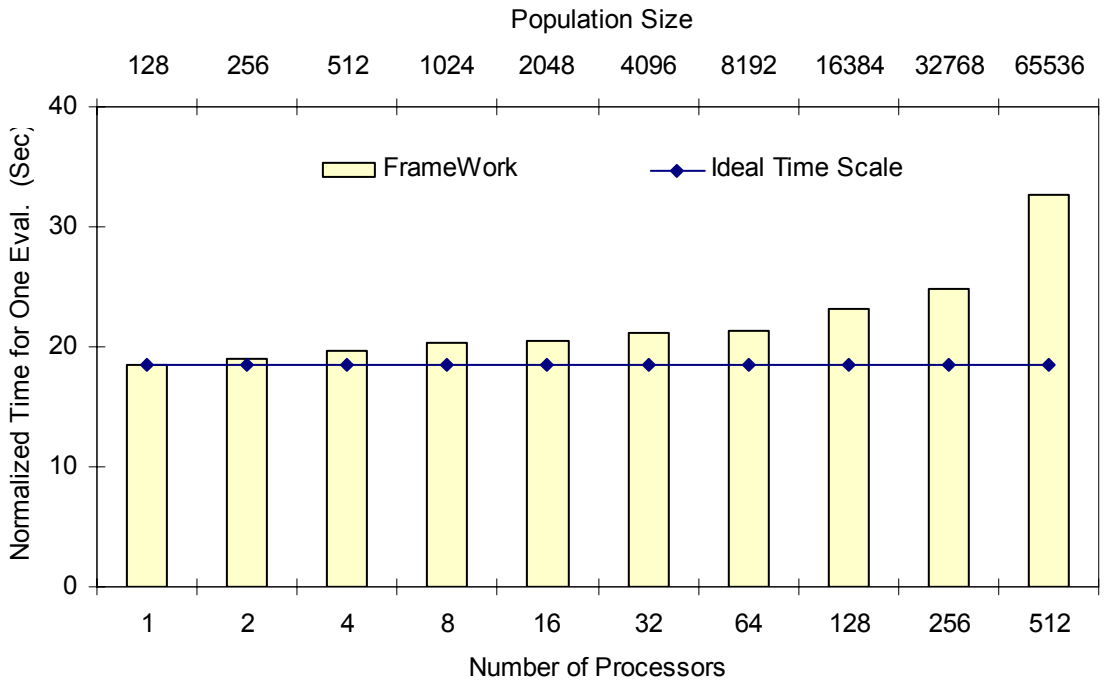

Figure 6-14. Scalability Result Utilizing NCSA Site, Procs/Group =1:1, Tasks/Procs = 1:1, Number of Servers =1

6.5.2.1.5 Scaling Analysis

The next set of runs was carried out to analyze the scalability of the framework. Throughout these runs the normalized evaluation time was observed with doubling problem size (i.e., population size) and number of processors. The study kept a separate set of parameters unchanged; these parameters comprise number of servers and number of processors per group. The normalized time in this case is equal to (total wall time) multiple by (number of processors) divided by (number of evaluations). The results plotted in Figure 6-14 are obtained using NCSA site and illustrate the normalized time and the theoretical time (Ideal Scale) corresponding to number of evaluations and number of processors. The figure shows that the normalized time for one evaluation slightly increases (less than 10%) as number of processors increases up to 16 processors. Beyond 16 processors the normalized time moderately increases (less than 15%) as number of processors increases until it reaches

64 processors. As the number of processors increases up to 512, the normalize time increases only about 35%. Scalability results obtained using SDSC and UC sites follow a similar trend to the NCSA results (see Appendix B-II).

6.5.2.2 Cross Site Runs

In this timing study, the optimizer (client) is initiated on one Teragrid site while the simulator (server) is initiated on another Teragrid site. Four sets of runs were used to investigate the framework performance across sites. On the first set of runs both optimizer and simulator sit on Teragrid ANL/UC site, for the second set of runs the optimizer sits on Teragrid ANL/UC and the simulator sits on Teragrid SDSC, on the third set of runs the optimizer sits on Teragrid ANL/UC and the simulator sits on Teragrid NCSA, and for the fourth set of runs the optimizer sits on Teragrid ANL/UC and the simulator sits on both Teragrid NCSA and TeraGrid SDSC. In these experiments the evaluation time is determined while the number of groups increases. The experiment kept parameters such as number of evaluations, number of processors per group and number of tasks per group unchanged. Timing study results shown in Figures 6-15 and 6-16 indicate that when the number of groups is equal to 32 the framework performs almost identically in both cross site and single site (i.e., ANL/UC-UC/ANL) runs. When number of groups is less than 32, the framework performs considerably better for single site runs than for cross site runs. This is probably due to the effect of increased number of messages sent across sites over the slower wide area network that is dominated by latency. The results shown in Figure 6-16 indicate that the cross site runs scale almost identically to the theoretical ideal time until 32 groups, and then scales sub-linearly when the number of group exceeds 32. Again this supports the important role

that the effect of communication plays in increasing the wall time when more than 32 groups are used.

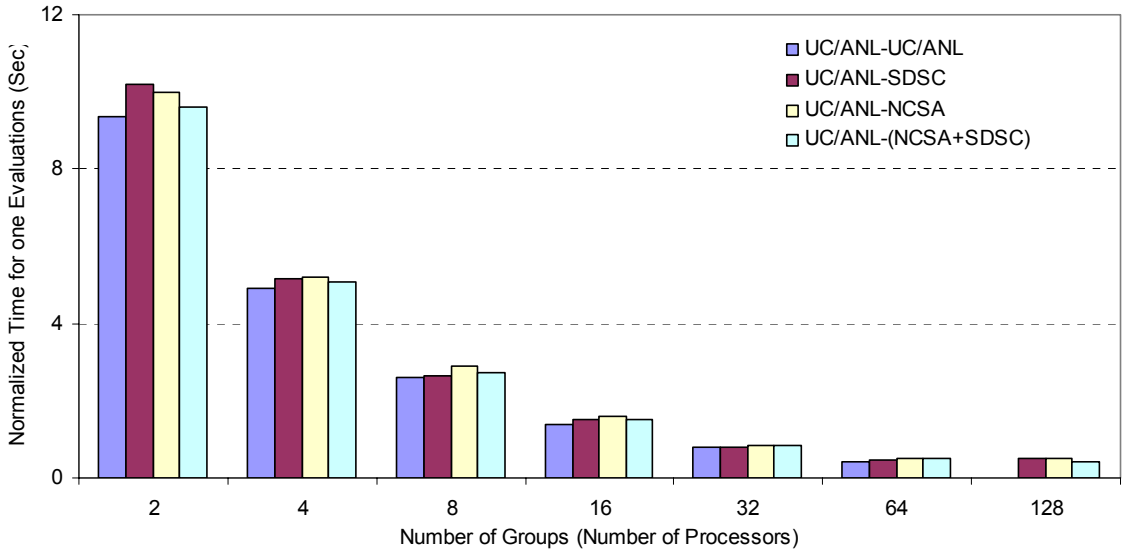

Figure 6-15. Cross site runs (Number of Groups vs. Normalized Time for One Evaluation), Procs/Group =1:1, Tasks/Group= 1:1, Resources to Run 128 Groups are Not Available at UC/ANL

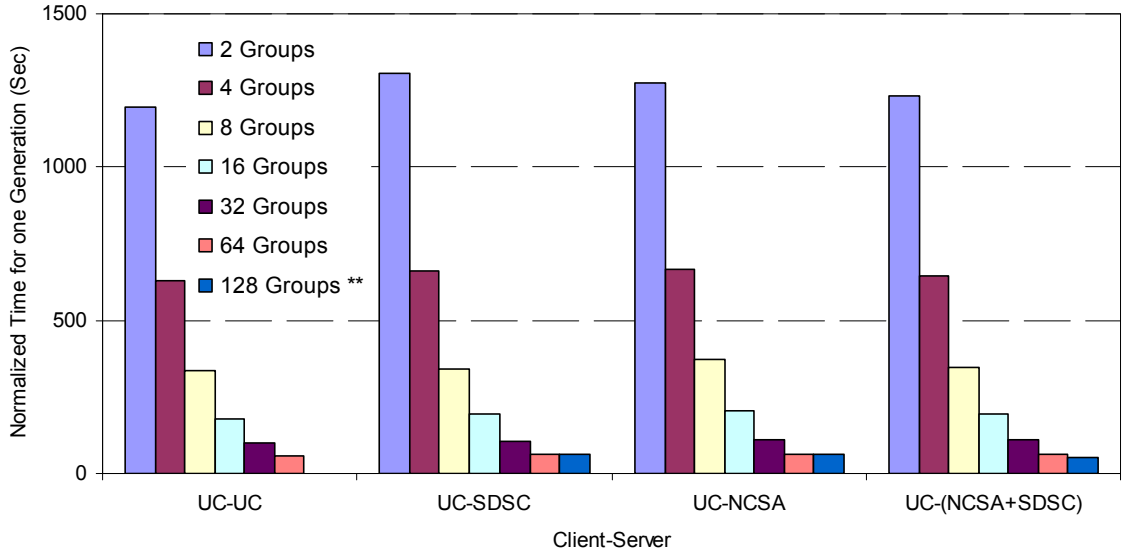

Figure 6-16. Cross site runs (Client-Server vs. Normalized Time for one Generation), Procs/Group =1:1, Tasks/Group= 1:1, ** Resources to Run 128 Groups are Not Available at UC/ANL

Table 6-4. Computation Resources Utilized for the SHR Problem

Parameters	Values
Number of Sites Utilized to Run the Optimizer (Master)	1 (ANL/UC)
Number of Sites Utilized to Run the Simulator (Worker)	2 (NCSA and SDSC)
Number of Servers in each Simulator site (Coarse Grained)	2
Number of Groups in each Server (Semi-Coarse Grained)	32
Number of Processors per Group (Fine Grained)	1
Total Number of Nodes (Processors)	73 (146)

The computational resources settings used to conduct the SHR problem were based on the framework cross site performance results described above and availability of resources at TeraGrid sites (i.e., amount of resource that is easy to access without considerably long wait time (matter of hours or days) in the scheduler queue). Three computational sites were used to conduct the search; the optimizer was run on the ANL/UC site while the simulator was executed on NCSA and SDSC sites jointly. For the problem considered in this paper, the number of forward model evaluations performed during a search is a function of the population size (128) multiplied by the number of generations (100); the total number of evaluations was 12,800. These evaluations were distributed among two sites (NCSA and SDSC); each site is utilizing two servers. Each server is utilizing a 32 processor parallel forward simulation model groups with a single processor per group. Thus, the study utilized a total of 73 computing nodes distributed in three sites. A single node is assigned for the optimization engine on the ANL/UC site, and two nodes for the servers with 34 nodes (68 processors) for the groups with masters in each NCSA and SDSC sites. Table 6-4 summarizes the allocation of computation resources. The normalized computation time for a single evaluation utilizing the above framework settings took approximately 0.210 second, versus 18.878 second on a single processor, i.e., solving the problem took approximately 35

minutes (2,055.77 seconds) instead of approximately two days (184,626.84 seconds). The speedup results show that the performance of the framework for this particular problem is approximately 90 times faster compared to the single processor results.

6.6 Final Remarks

This paper described the development and analyzed the performance of a grid-enabled simulation-optimization framework implemented on TeraGrid sites. An illustrative environmental characterization problem in the groundwater domain was utilized to test and evaluate the framework in terms of computational performance and solution capability. A genetic algorithm optimizer and a parallel finite-element simulator were used in this study.

During the early development stage, the grid-enabled parallelized framework was benchmarked against a FORTRAN-MPI implementation in order to identify performance bottlenecks and implement improvements. The performance of the framework was then investigated at single and cross site TeraGrid resources. Performance was analyzed at several levels: coarse grained within the population-based search algorithm; and fine and semi-coarse grained within the simulation model. In addition, the effect of chunk size and scalability within the population-based search algorithm were investigated.

In general, the reported slow-down in some performance results are yet within an acceptable level considering the advantages of the modular framework design and the geographically distributed resources. The results indicate that significant raw performance improvements are possible through the choice of appropriate parallelism parameters. Though the TeraGrid sites utilized in these tests are homogeneous with respect to microprocessor

technology, significant performance differences were observed between each site possibly due to system configuration differences.

Overall, this study demonstrates that a hierarchically parallel framework can be used to harness the power of distributed heterogeneous resources such as the TeraGrid for solving computation intensive simulation-optimization problems by reducing the computation time by several orders of magnitudes, while maintaining the solution quality.

CHAPTER 7

SUMMARY AND CONCLUSIONS

This dissertation research focused on addressing inverse problems using a simulation-optimization approach where the optimization is conducted using evolutionary algorithms. Methodological and computational investigations were performed to improve the time efficiency of the simulation component and the EA-based search procedures. These methodological and new EA-based techniques were applied to and tested using illustrative applications in groundwater contaminant characterization. Finally, the development of a distributed computational framework implemented on the TeraGrid sites is reported, where the solution procedure employs a distributed version of the EA-based search procedure.

High performance computing was utilized to develop a multilevel parallel computation framework to improve the time efficiency of the simulation-optimization approach. The parallelized framework was investigated in two ways: coarse grained within the population-based search algorithm; and fine and coarse grained within the groundwater transport simulation model. The architecture was implemented and executed on the NCSA TeraGrid site for solving both homogenous and heterogeneous three-dimensional contaminant source identification case studies. A set of scenarios was introduced to investigate the groundwater contaminant source identification problem. The results indicate that the solution for the problem performs adequately for all scenarios. The effect of non-uniqueness was more pronounced in more complex scenarios (e.g., more decision variables) than in the simpler cases. Overall, these implementations show a successful reduction in the evaluation time from hours to minutes, while still maintaining the solution quality.

Surrogate modeling using an artificial neural network-based approach was used as an alternative technique to improve the simulation time efficiency of the simulation-optimization approach. The surrogate models were developed and applied to both the two and three-dimensional groundwater simulation transport models for hypothetical groundwater domains. The results show the considerable success achieved by the surrogate models in mimicking accurately the groundwater simulation models for all case study scenarios. The surrogate models were coupled with an evolution strategies-based search procedure for solving instances of the source identification problem with different levels of complexity. The results also indicate that the search algorithm was able to estimate correctly the decision variables for most scenarios while the use of the surrogate model yielded significantly faster convergence. Also, the results demonstrate that the effect of the non-uniqueness in the solution is more significant as the problem complexity increases. In addition, the results from the timing study analysis indicate that the use of the surrogate model achieves three orders of magnitude improvement in the execution time in comparison to using the original numerical simulation model.

Hybrid methods, a subset of combinatorial optimization techniques, offer a potentially effective approach to address instances of problems, including inverse problems that are thought to be hard since they have a large and/or a complex decision space. New EA-based hybrid procedures were developed and tested to evaluate the accuracy of the solutions as well as the time efficiency in comparison to those of the EA-based sequential hybrid search algorithms. The new methods embed the power of local search in an EA-based global search. These new procedures were applied to two instances of the contaminant source characterization problem, namely source identification problem, and combined problem.

Results for the embedded hybrid methods, in comparison with the sequential hybrid method, showed that the new methods were able to achieve comparable or higher quality solutions for the two-dimensional application problems. The results also showed that all new embedded methods consumed relatively a higher number of evaluations than the sequential hybrid methods; however, the new hybrid methods were able to fully maintain parallelism to attain maximum computational efficiency. This investigation should be extended to a more complex three-dimensional application problem.

The real-world applications of groundwater inverse problems are commonly faced with numerous conditions of uncertainty, including uncertainty in the system structural parameters and/or the system input parameters. Traditional techniques based on computationally expensive procedures are usually used to solve this problem, where a relatively large number of random realizations of the uncertain parameters is incorporated to represent the uncertainty. A new EA-based noisy algorithm, the Archived Noisy Genetic Algorithm (ANGA), was developed to include random realizations to model incorporate the uncertainty in a more time-efficient approach. The new procedure, ANGA, was tested using a two-dimensional groundwater source identification problem. ANGA was applied to two instances of the problem with different types of uncertainty. For the first case, the uncertainty was assumed to exist in the observation data, and for the second case the uncertainty was assumed to be in the system input parameters (namely, the hydraulic conductivity fields that directly influence the flow and contaminant transport). The preliminary results are promising, where they show that equal or better solution quality can be achieved with fewer simulation model evaluations when compared with the standard noisy genetic algorithms.

The development and the performance of a grid-enabled computational framework implemented on TeraGrid sites were presented and discussed. The groundwater release history reconstruction problem was utilized to evaluate the framework in terms of computational performance and solution quality. The problem was solved using the simulation-optimization approach enabled within the framework where the simulation is performed using a three-dimensional groundwater transport code while the optimization is conducted using genetic algorithms. The framework was investigated for both single and cross site runs that included: coarse grained parallelism within the search algorithm; and fine and semi-coarse grained parallelism within the simulation model. Additionally, the effect of chunk size and scalability within the search algorithms were also investigated. Overall, the framework implementations show promise in drastically reducing the evaluation time for the groundwater inverse problems by several orders of magnitudes, while maintaining the solution quality. The slowdown (i.e., sub-linearity) observed in some performance results were within an acceptable level taking into considering the advantages of the framework design.

REFERENCES

Abbattista, F., Abbattista N. and Caponetti, L. (1995). "*An Evolutionary and Cooperative Agent Model for Optimization*", In IEEE Int. Conf. on Evolutionary Computation ICEC'95, Perth, Australia, pp. 668–671.

Aly, A. A. and Peralta, R. C. (1999). "*Optimal design of aquifer cleanup systems under uncertainty using a neural network and a genetic algorithm*", Water Resour. Res., 35(8), 2523–2532.

Andradóttir, S. (1998). "*A Review of Simulation Optimization Techniques*", In D.J. Medeiros, E. F. Watson, J. S. Carson, and M. S. Manivannan (eds.), Proceedings of the 1998 Winter Simulation Conference, 151-158.

Aral, M. M., Guan, J. and Maslia, M. L. (2001). "*Identification of contaminant source location and release history in aquifers*", Journal of Hydraulic Engineering (ASCE), Volume 6, Issue 3, pp. 225-234.

Atmadja, J. and Bagtzoglou, A. C. (2001). "*State of Art Report on Mathematical Methods for Groundwater Pollution source identification*", Environmental Forensics, 2(3) 205-214, 2001b.

Azadivar, F. (1999). "*Simulation Optimization Mythologies*", in Procedure of the 1999 Winter Simulation Conference, 93-100.

Bachelet, V., Preux, P. and Talbi, E.G. (1996). "*Parallel Hybrid Meta-heuristics: Application to the Quadratic Assignment Problem*", In Parallel Optimization Colloquium POC96, Versailles, France, pp. 233–242.

Back, T. (editor) (1997). "*Handbook of Evolutionary Computation*", IOP Publishing Ltd. and Oxford University Press.

Badeau, P., Guertin, F., Gendreau, M., Potvin, J-Y. and Taillard, E. D. (1997). "*A Parallel Tabu Search Heuristic for the Vehicle Routing Problem with Time Windows*", Transportation Research 5(2), 109–122.

Baeck, T. (1995). "*Evolutionary Algorithms in Theory and Practice*", Oxford University Press, Oxford.

Baugh, J.W. (2003). "*Vitri 2.0.*", Available: http://www4.ncsu.edu/ jwb/vitri/.

Bear, J. (1979). "*Hydraulics of groundwater*", New York: MCGraw-Hill.

Beyer, H-G. (2001). "*The Theory of Evolution Strategies*", Berlin, Germany: Springer-Verlag.

Bishop, C. M. (1995). "*Neural networks for pattern recognition*", Oxford: Oxford University Press.

Catlett, C. (2002). "*The TeraGrid: A primer*", Available: http://www.teragrid.org/.

Catlett, C. (2005). "*TeraGrid: A foundation for US Cyberinfrastructure*", Proceedings of NPC 2005. http://www.teragrid.org.

Chan Hilton, A. B. and Culver, T. B. (2005). "*Groundwater remediation design under uncertainty using a robust genetic algorithm*", Journal of Water Resources Planning and Management, ASCE, 131(1), 25-34.

Cotta, C., Talbi, E-G. and Alba, E. (2005). "*Parallel hybrid metaheuristics*", in Parallel Metaheuristics, Edited by E. Alba, pp. 347-370, John Wiley & Sons, 2005.

Cox, S. J., Fairman, M. J., Xue, G., Wason, J. L. and Keane A. J. (2001). "*The grid: Computational and data resource sharing in engineering optimization and design search*", in *IEEE Proceedings of the 2001 ICPP Workshops*. Valencia, Spain: IEEE, September, pp. 207–212.

Datta, B. and Chakrabarty, D. (2003). "*Optimal identification of unknown pollution sources using linked optimization simulation methodology*", Proc. of Symp. on Advances in Geotechnical Engineering (SAGE 2003), Indian Institute of Technology, Kanpur, India, 368–379.

Davis, L. (editor) (1991). "*Handbook of genetic algorithms*", Van Nostrand Reinhold, New York, N.Y.

Demuth, H. and Beale, H. M. (2005). "*Neural Network Toolbox for Use with MATLAB*, version 4", Mathworks, Natick, Mass., 2005.

El-Din, A. G., Smith, D. W. and Gamal El-Din, M. (2004). "*Application of artificial neural networks in wastewater treatment*", Journal of Environmental Engineering and Science, Ottawa, Vol. 3, pg. S81, 15 pgs.

Espinoza, F. P. and Minsker, B. S. (2006). "*Development of the enhanced self-adaptive hybrid genetic algorithm (e-SAHGA)*", Water Resour. Res., 42, W08501, doi:10.1029/2005WR004221.

Fitzpatrick, J. M. and Grefenstette, J. J. (1988). "*Genetic algorithms in noisy environments*", Machine Learning, 3, 101-120, 1988.

Fleurant, C. and Ferland, J. A. (1996). "*Genetic and Hybrid Algorithms for Graph Coloring*", Annals of Operations Research 63.

Flood, I. and Kartam, N. (1994). "*Neural networks in civil engineering: principles and understanding*", Journal of Comp. Civil Eng. 8(2), pp. 131- 145.

Foster, I. and Kesselman, C. (1999). "*The grid, blueprint for a new computing infrastructure*", Morgan Kauffman Publishers, San Francisco, 677p.

Gang, X., Song, W., Keane, A. J. and Cox, S. J. (2004). "*Developing services for design optimization on the grid*", IEEE.

Goldberg, D. (1989). "*Genetic Algorithms in Search, Optimization, and Machine Learning*", Addison-Wesley, 412 pp.

Gopalakrishnan, G., Minsker, B. S. and Goldberg, D. E. (2001). "*Optimal sampling in a noisy genetic algorithm for risk-based remediation design.*", Proc., World Water and Environmental Resources Congress, ASCE, Reston, Va.

Gopalakrishnan, G., Minsker, B. S. and Goldberg, D. E. (2003). "*Optimal sampling in a noisy genetic algorithm for risk-based remediation design*", Journal of Hydroinformatics 5, 11-25.

Gorelick, S. M., Evans, B. E. and Remson, I. (1983). "*Identifying sources of groundwater pollution: an optimization approach*", Water Resour. Res. 19, 779-790. No. 3.

Grefenstette J. J. and Fitzpatrick J. M. (1995). "*Genetic search with approximate function evaluations*", (Ed.), Proceedings of an International Conference on Genetic Algorithms and their Applications, pp 112-120, Hillsdale, NJ, 1985.

Gropp, W., Lusk, W. and Skjellum, A. (1999). "*Using MPI: Portable Parallel Programming with the Message –Passing Interface*", 2nd The MIT Press, Cambridge, MA.

Harrell, L. J. (2001). "*Evolutionary algorithm-based design of a system of wet detention basins under uncertainty for watershed management*", Proc., World Water and Environmental Resources Congress, ASCE, Reston, Va.

Holland, J. H. (1975). "*Adaptation in Natural and Artificial Systems*", University of Michigan Press, Ann Arbor.

Hooke, R. and Jeeves, T. A. (1961). "*Direct Search Solution of Numerical and Statistical Problems*", (Vol. 8, pp. 212-229).

Huyakorn, P. S. and Pinder, G. F. (1983). "*Computational Methods in Subsurface Flow*", Academic Press, New York, N.Y.

Iman, R. L., Davenport, J. M. and Zeigler, D. K. (1980). "*Latin hypercube sampling (program user's guide)*", OSTI:5571631.

Iman, R. L., Helton, J. C. and Campbell, J. E. (1981). "*An approach to sensitivity analysis of computer models, Part 1. Introduction, input variable selection and preliminary variable assessment*", Journal of Quality Technology 13 (3): 174–183.

Johnson, V. M. and Rogers, L. L. (2000). "*Accuracy of neural network approximation in simulation-optimization*", Journal of Water Resources planning and Management-ASCE, 126(2):48-56 Mar/Apr.

Loughlin, D. H. and Ranjithan, S. R. (1999). "*Chance-constrained genetic algorithms*", Proc., Genetic and Evolutionary Computation Conference. (GECCO'99), Vol. 1, W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, V. Jakiela, and R. E. Smith, eds., Morgan Kaufmann, San Franscisco, 369–376.

Mahar, P. S. and Datta, B. (1997). "*Optimal monitoring network and ground-water-pollution source identification*", Journal of Water Resources Planning and Management (ASCE) 123(4):199–207.

Mahar, P. S. and Datta, B. (2000). "*Identification of pollution sources in transient groundwater systems*", Water ResourcesManagement 14(3): 209–227.

Mahar, P. S. and Datta, B. (2001). "*Optimal identification of groundwater pollution sources and parameter estimation*", Journal of Water Resources Planning and Management (ASCE) 27(1):20–29.

Mahinthakumar, G. (1999). "*PGREM3D: Massively Parallel Codes for Groundwater Flow and Transport*", Available: http://www4.ncsu.edu/~gmkumar/pgrem3d.pdf.

Mahinthakumar, G. and Sayeed, M. (2005). "*Hybrid genetic algorithm local search methods for solving groundwater source identification inverse problems*", Journal of Water Resources Planning and Management (ASCE), vol. 131, no. 1, pp. 45–57, Jan/Feb.

Mahinthakumar, G., Saied F. and Valocchi, A. J. (1997). "*Comparison of some parallel krylov solvers for large scale contaminant transport simulations*", High Performance Computing (Editor: A. M. Tentner), p. 134-139.

Martin, O. C., Otto, S. W. and Felten, E. W. (1992). "*Large-Step Markoy Chains for the TSP: Incorporating Local Search Heuristics*", Operation Research Letters 11, 219–224.

MATLAB (2006). "*MATLAB, the language of Technical Computing, User Guide*", version 7.2.0.232 Release 2006a, the Mathwork Inc., Natick, MA.

Maz'ya, V. and Shaposhnikova, T. O. (1998). "*Life and Work of Jacques Hadamard*", American Mathematical Society, hardcover, 574 pages, ISBN 0-8218-0841-9.

McDonald, P.H. (2001). "*Fundamentals of infrastructure engineering: Civil engineering systems*", New York: Marcel Dekker.

McKay, M. D., Conover, W. J. and Beckman, R. J. (1979). "*A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code*", Technometrics 21: 239–245.

Miller, B. L. and Goldberg, D. E. (1996). "*Optimal sampling for genetic algorithms*", IlliGAl Rep. No. 96005, Univ. of Illinois, Champaign, Ill.

Mirghani, B., Tryby, M., Baessler, D., Karonis, N., Ranjithan, R. and Mahinthakumar, K. (2005a). "*Development and Performance Analysis of a Simulation-Optimization Framework on TeraGrid Linux Clusters*", The 6th LCI International Conference, Chapel Hill, NC.

Mirghani, B., Tryby, M., Nicholas, K., Ranjithan, R. and Mahinthakumar, G. (2007c under review). "*Grid-Enabled Simulation-Optimization Framework for Environmental Characterization*", Journal of Computing in Civil Engineering.

Mirghani, B., Tryby, M., Ranjithan, R., and Mahinthakumar, G. (2006). "*A Grid-enabled Simulation-Optimization Approach for Solving Groundwater Characterization Problems*", the International Conference on Computing in Civil and Building Engineering, Montréal, Canada, 2006.

Mirghani, B., Tryby, M., Ranjithan, R., Baessler, D., Nicholas, K. and Mahinthakumar, K. (2005b). "*A Grid-enabled Simulation-Optimization Framework for Environmental Characterization Problems*", Poster presentation at Super Computing Conf., Seattle WA.

Mirghani, B., Tryby, M., Ranjithan, R., Zechman, E. and Mahinthakumar, G. (2007b under review). "*Computational Simulation-Optimization Approach for Solving Inverse Problems*", Advances in Water Resources.

Mirghani, B., Zechman, E., Mahinthakumar, G. and Ranjithan, R. (2007a under review). "*Enhanced Simulation-Optimization Approach using Surrogate Modeling for Solving Inverse Problems*", Journal of Hydroinformatics.

Morshed, J. and Kaluarachchi, J. J. (1998). "*Parameter estimation using artificial neural network and genetic algorithm for free-product migration and recovery*", Water Resour. Res., 34(5), 1101–1114.

Neupauer, R. M. and Wilson, J. L. (2001). "*Adjoint-derived location and travel time probabilities for a multi-dimensional groundwater system*", Water Resour. Res. 37, 1657-1668. No. 6.

Neupauer, R. M., Borchers, B. and Wilson, J. L. (2000). "*Comparison of inverse methods for reconstructing the release history of a groundwater contaminant source*", Water Resources Research 36(9):2469–2475.

Oh, I. S., Lee, J. S. and Moon, B. R. (2004). "*Hybrid genetic algorithms for feature selection*", IEEE Trans. Pattern Anal. Machine Intell, 26 1424–1437.

Ong, Y. S., Nair P. B. and Keane A. J. (2003). "*Evolutionary Optimization of Computationally Expensive Problem via Surrogate Modeling*", American Institute of Aeronautics and Astronautics Journal, Vol. 41, No. 4, pp. 687-696.

Ostfeld, A. and Salomons, E. (2005). "*Securing Water Distribution Systems Using Online Contamination Monitoring*", Journal of Water Resources Planning and Management, Vol. 131, No. 5, September 1, 2005.

Parashar, M., Klie, H., Catalyurek U., Kurc, T., Bangerth, W., Matossian, V., Saltz, J. and Wheeler, M. F. (2004). "*Application of grid-enabled technologies for solving optimization problems in data-driven reservoir studies*", Future Generation of Computer Systems.

Partheepan, R. (2003). "*Hybrid Genetic Algorithms*", M.Sc. Thesis, North Carolina State University.

Poeter, E. P. and Hill M. C. (1997). "*Inverse models: A necessary next step in groundwater modeling*", Groundwater, 35 (2): 250-260 MAR-APR 1997.

Powell, M. J. D. (1964). "*An efficient method for Finding the Minimum for a function of several variables without calculating derivatives*", The Computer Journal, 7, 155-162.

Press, W. H., Teukolsky, S. A., Vetterling, W. T. and Flannery, B. P. (1997). "*Numerical recipes in FORTRAN 77 and FORTRAN 90"*, Cambridge University Press, Cambridge, U.K.

Preux, P. and Talbi, E-G. (1999). "*Towards hybrid evolutionary algorithms*", International Transactions in Operational Research, Vol.6, No.6, pp.557-570.

Rajanayaka, C., Samarasinghe, S. and Kulasiri, D. (2002). "*Solving the Inverse Problem in Stochastic Groundwater Modelling with Artificial Neural Networks*", Proceedings of the Biennial Congress of the International Environmental Modelling and Software Society; Lugano, Switzerland. pp.154-159.

Ranjithan, S., Eheart, J. W. and Garrett, J. H. Jr. (1993). "*Neural network-based screening for groundwater reclamation under uncertainty*", Water Resources Research, 29(3):563-574.

Reed, D. A. (2003). "*Grids, the TeraGrid, and Beyond*", IEEE Computer, 36(1): p. 62-68.

Sayeed, M. (2003). "*An efficient parallel optimization framework for inverse problems*", PhD dissertation, Department of Civil. Construction and Environmental Engineering, North Carolina State University, Raleigh, USA.

Sayeed, M. and Mahinthakumar, G. (2002). "*A multilevel parallelization scheme based on MPI communicators for solving groundwater inverse problems*", High performance computing 2002, Society for Computer Simulation International, p. 137-144.

Sayeed, M. and Mahinthakumar, G. (2005). "*An efficient parallel implementation of hybrid optimization approaches for solving groundwater inverse problems*", ASCE Journal of Computing, 19(4):329-340.

Schwefel, H-P. (1995). "*Evolution and Optimum Searching*", Wiley Interscience, John Wiley & Sons, New York.

Shigidi, A. and Garcia, L. (2003). "*Parameter Estimation in Groundwater Hydrology Using Artificial Neural Networks*", Journal of Computing in Civil Engineering, Vol. 17, No. 4, October 1.

Sidauruk, P., Cheng, A. and Ouazar, D. (1998). "*Ground water contaminant source and transport parameter identification by correlation coefficient optimization*", Ground Water 36, 208-214. No. 2.

Simunek, J., Angulo-Jaramillo, R., Schaap, M., Vandervaere, J-P. and Van Genuchten, M. Th. (1998). "*Using an inverse method to estimate the hydraulic properties of crusted soils from tension disc infiltrometer data*", Geoderma, 86(1-2), 61-81.

Singh, R. M., Datta, B. and Jain, A. (2004). "*Identification of unknown groundwater pollution sources using artificial neural network*", Journal of Water Resources Planning and Management, Vol. 130, No. 6, November 1.

Singh, R. M., Datta, B. and Jain, A. (2006). "*Identification of Groundwater Pollution Sources Using GA-based Linked Simulation Optimization Model*", J. Hydrologic Engrg., Volume 11, Issue 2, pp. 101-109.

Smalley, J. B., Minsker, B. S. and Goldberg, D. E. (2000). "*Risk-based in situ bioremediation design using a noisy genetic algorithm*", Water Resour. Res., 36(10), 3043–3042.

Sun, M. (1997). "*A New Search Procedure for Distributed Parameter Identification*", Proceedings of the 29 Southeastern Symposium on System Theory (SSST'97).

Sun, N.Z. (1994). "*Inverse problem in groundwater modeling, Theory and application of Transport in porous media*", (Ed. Jacob Bear), Vol. 6, Kluwer Academic Publishers, 337p.

Talbi, E.G. (2002). "*A taxonomy of hybrid metaheuristics*", Journal of Heuristics Volume 8, P541-564.

Thompson, A. F. B., Aboubu, R. and Gelhar, L. W. (1989). "*Implementation of the three-dimensional turning bands random field generator*", Water Resources Research 25(10):2227–2243.

Tryby, M., Mirghani, B., Ranjithan, R., Baessler, D., Nicholas, K. and Mahinthakumar, K. (2005). "*LASSO: A Grid-Enabled Simulation Optimization Framework*", Grid Workshop at Super Computing Conf., Seattle WA.

Tsai, F. T-C., Sun, N-Z. and Yeh, W. W-G. (2003). "*Global-local optimization for parameter structure identification in three-dimensional groundwater modeling*", Water Resour. Res., 39(2), 1043, doi:10.1029/2001WR001135.

Wagner, B. J. (1992). "*Simultaneously parameter estimation and contaminant source characterization for coupled groundwater flow and contaminant transport modeling*", J. of Hydrology 135, 275-303.

Wagner, B. J. and Gorelick, S. M. (1987). "*Optimal groundwater quality management under parameter uncertainty*", Water Resour. Res., 23(7), 1162–1174.

Yeh, W-G. (1986). "*Review of parameter identification procedures in groundwater hydrology: The inverse problem*", Water Resources Research 22(2):95–108.

Yolanda, C. and Anu, M. (1997). "*Simulation Optimization: Methods and Applications*", Proceedings of the 1997 Winter Simulation Conference, 118-126.

Zheng, C. and Bennett, G. D. (2002). "*Applied Contaminant Transport Modeling*", Second Edition, John Wiley & Sons, New York, 621 pp.

Zio, E. (1997). "*Approaching the inverse problem of parameter estimation in groundwater models by means of Artificial Neural Network*", Progress in Nuclear Energy Vol. 31, No. 3, pp. 303-315.

APPENDICES

Appendix A-I Solution of the Two Dimensional Problem Scenarios with Noise

2D-1 Noise

Table 3-7A. Best Source Characterization Solution for Scenario 2D-1-Noisy Utilizing Collected Observations from One Well

Parameters	True	Predicted (1 Wells)
Variable x_c	8	7.73
Variable y_c	10	14.32
Variable C_o	70	61.64
Objective Function Error	-	0.0058
Solution Error %	-	14.64%

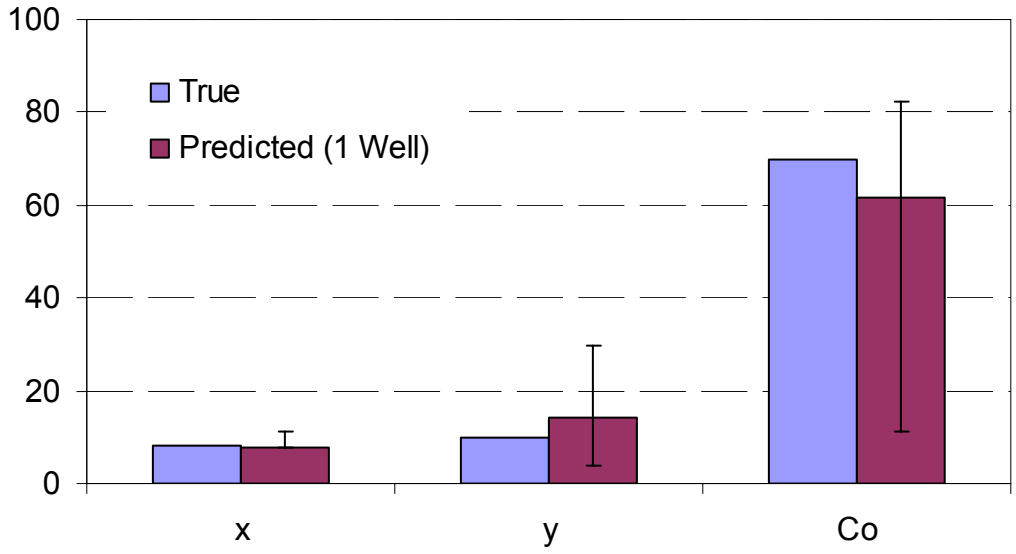

Figure 3-12A. Best Source Characterization Solution for Scenario 2D-1-Noisy Utilizing Collected Observations from One Well along with Result for 30 Trials

2D-2 Noise

Table 3-8A. Best Source Characterization Solution for Scenario 2D-2-Noidy Utilizing Collected Observations from One and Three Wells

Parameters	True	Predicted (1 Wells)	Predicted (3 Wells)
Variable x_c	8	7.80	8.04
Variable y_c	10	16.92	10.01
Variable s	2	2.66	2.03
Variable C_o	70	29.84	68.82
Objective Function Error	-	0.0055	0.0194
Solution Error %	-	40.561%	0.874%

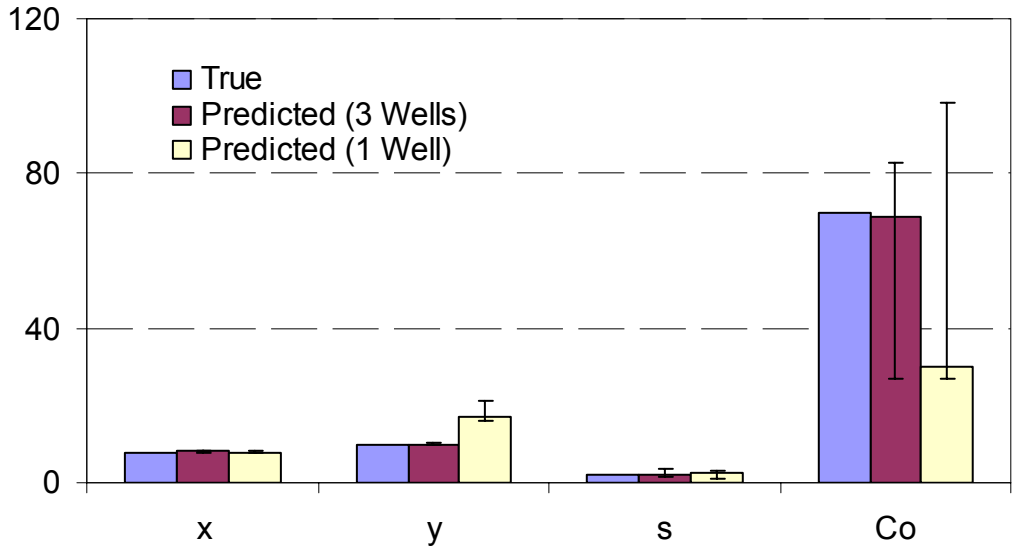

Figure 3-13A. Source Characterization Solution for Scenario 2D-2-Noisy Utilizing Collected Observations from One and Three Wells along with Results for 30 Trials

2D-3 Noise

Table 3-9A. Best Source Characterization Solution for Scenario 2D-3-Noisy Utilizing Collected Observations from One, Three and Five Wells

Parameters	True	Predicted (1 Well)	Predicted (3 Wells)	Predicted (5 Wells)
Variable x	8	7.65	8.00	8.01
Variable y	10	13.01	10.01	10.02
Variable s_x	2	1.47	2.53	2.34
Variable s_y	2	3.70	2.55	2.48
Variable C_o	70	39.03	50.07	54.13
Objective Function Error	-	5.00E-03	9.70E-03	1.11E-02
Solution Error %	-	40.902%	14.038%	11.749%

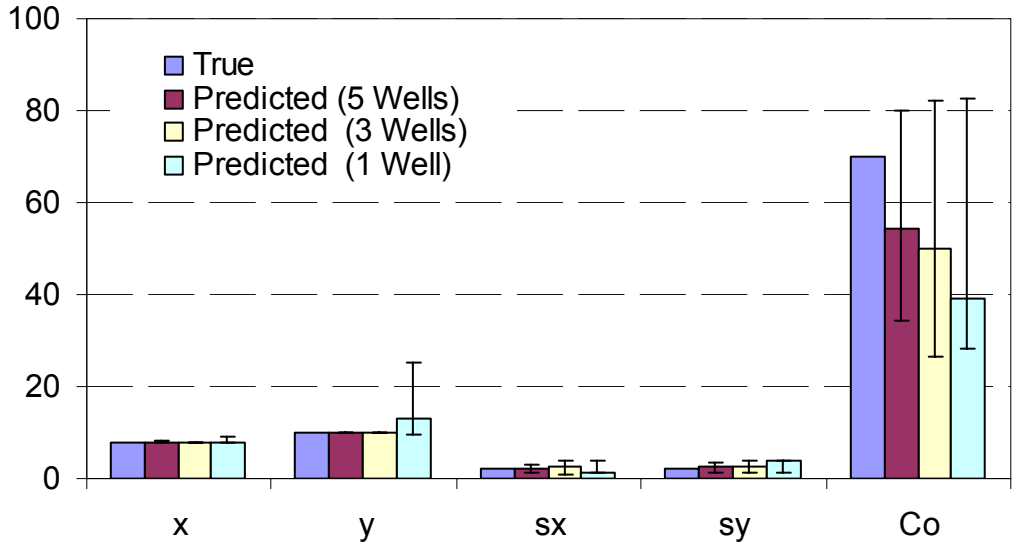

Figure 3-14A. Source Characterization Solution for Scenario 2D-3-Noisy Utilizing Collected Observations from One, Three and Five Wells along with Results for 30 Trials

Appendix A-II Solution of the Three Dimensional Problem Scenarios with Noise

3D-1 Noise

Table 3-10A. Best Source Characterization Solution for Scenario 3D-1-Noisy Problem Utilizing Collected Observations from Four Wells

Parameters	True	Predicted (4 Wells)
Variable z_c	4	5.46
Variable x_c	8	7.91
Variable y_c	10	10.00
Variable C_o	70	69.34
Objective Function Error	-	0.028
Solution Error %	-	9.64 %

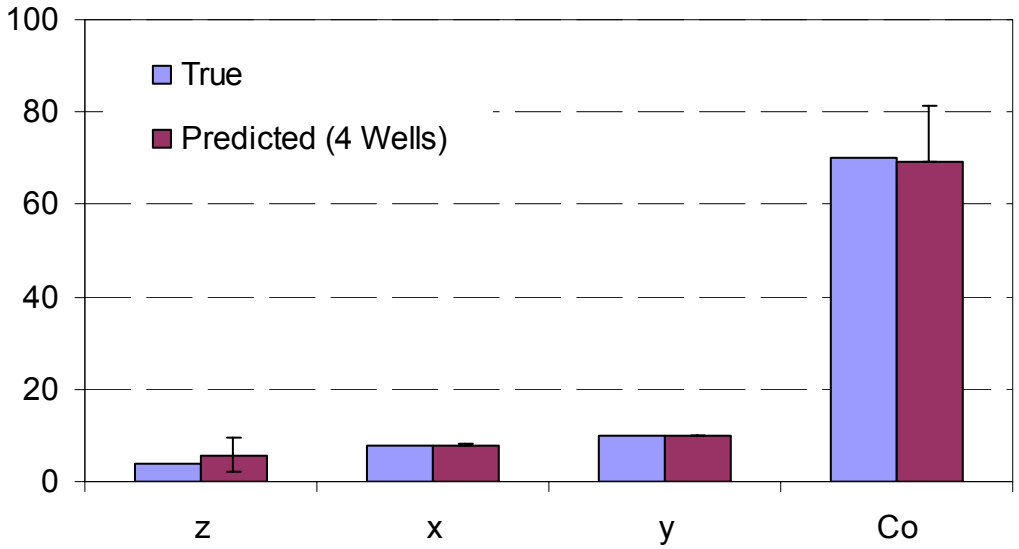

Figure 3-15A. Source Characterization Solution for Scenario 3D-1-Noisy Utilizing Collected Observations from Four Wells along with Results for 30 Trials

3D-2 Noise

Table 3-11A. Best Source Characterization Solution for Scenario 3D-2-Noisy Problem Utilizing Collected Observations from Four Wells

Parameters	True	Predicted (4 Wells)	Predicted (9 Wells)
Variable z_c	4	6.44	3.92
Variable x_c	8	7.90	7.95
Variable y_c	10	10.03	10.00
Variable s	2	4.01	2.10
Variable C_o	70	15.31	66.10
Objective Function Error	-	0.025	0.043
Solution Error %	-	48.23 %	2.67 %

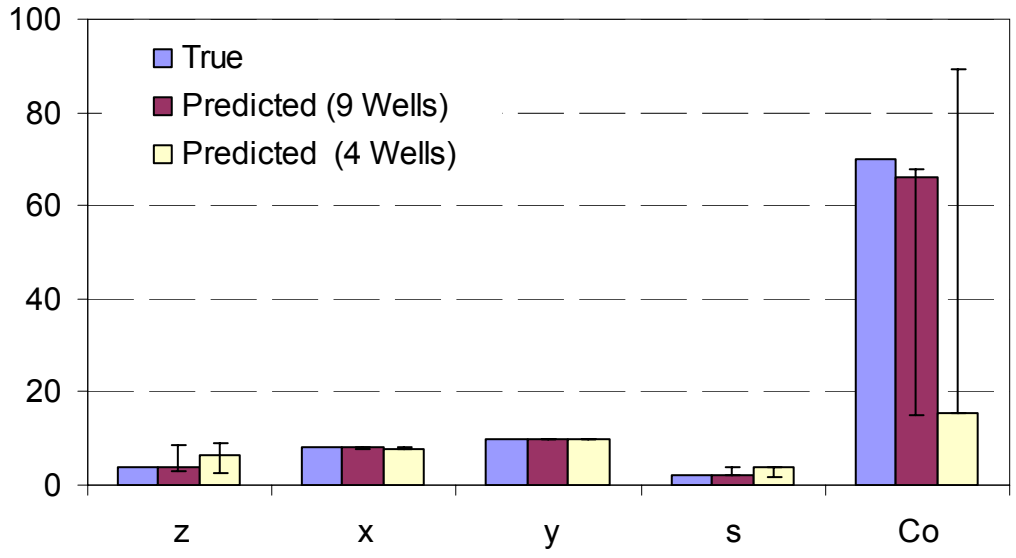

Figure 3-16A. Source Characterization Solution for Scenario 3D-2-Noisy Utilizing Collected Observations from Four and Nine Wells along with Results for 30 Trials

3D-3 Noise

Table 3-12A. Best Source Characterization Solution for Scenario 3D-3-Noisy Problem Utilizing Collected Observations from Four Wells, Nine Wells and Eighteen Wells

Parameters	True	Predicted (4 Wells)	Predicted (9 Wells)	Predicted (18 Wells)
z	4	7.45	6.05	6.55
x	8	8.01	7.90	7.92
y	10	9.98	9.96	9.98
s_z	2	3.55	3.15	3.14
s_x	2	2.66	3.64	3.46
s_y	2	3.66	2.86	2.69
C_o	70	24.48	25.70	28.12
Objective Function Error	-	0.019	0.023	0.026
Solution Error %	-	49.31 %	42.67 %	41.36 %

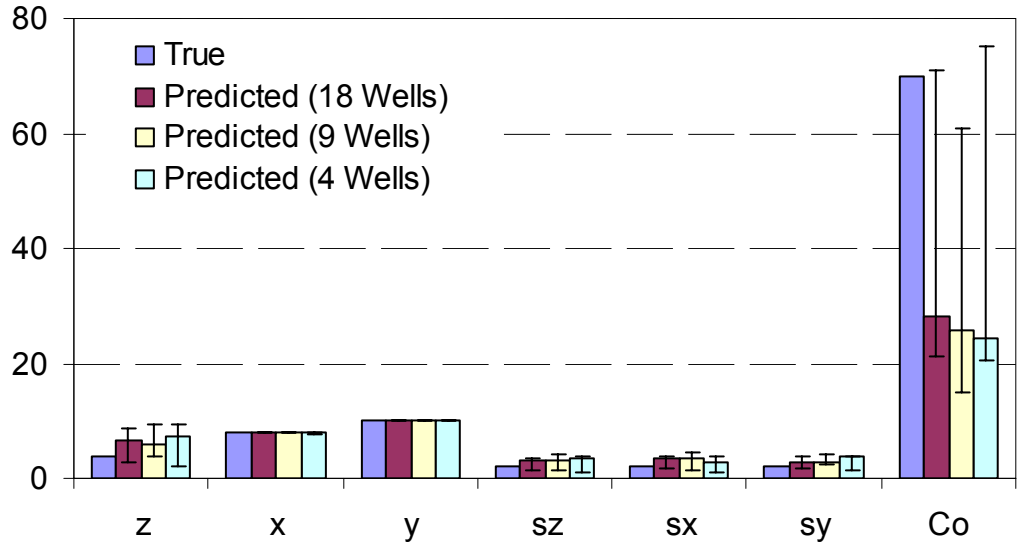

Figure 3-17A. Source Characterization Solution for Scenario 3D-3-Noisy Utilizing Collected Observations from Four, Nine and Eighteen Observation Wells with Resuls for 30 Trials

Appendix B-I Effect of the Chunk Size

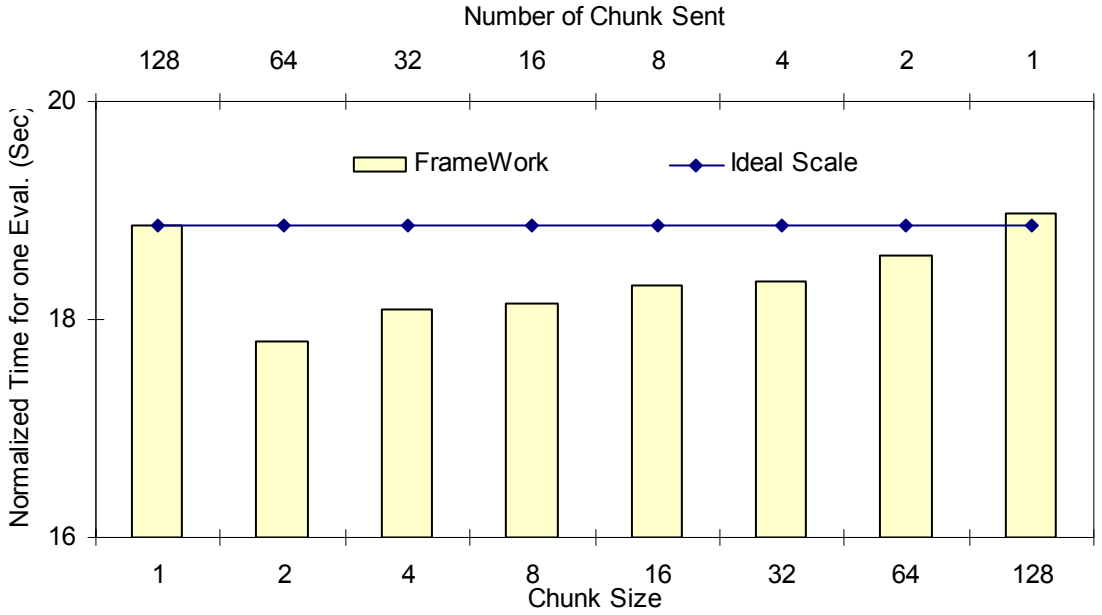

Figure 6-13A. The effect of Chunk Size Utilizing SDSC Site, Number of Groups = 1, Population Size = 128, Procs/Group =1:1

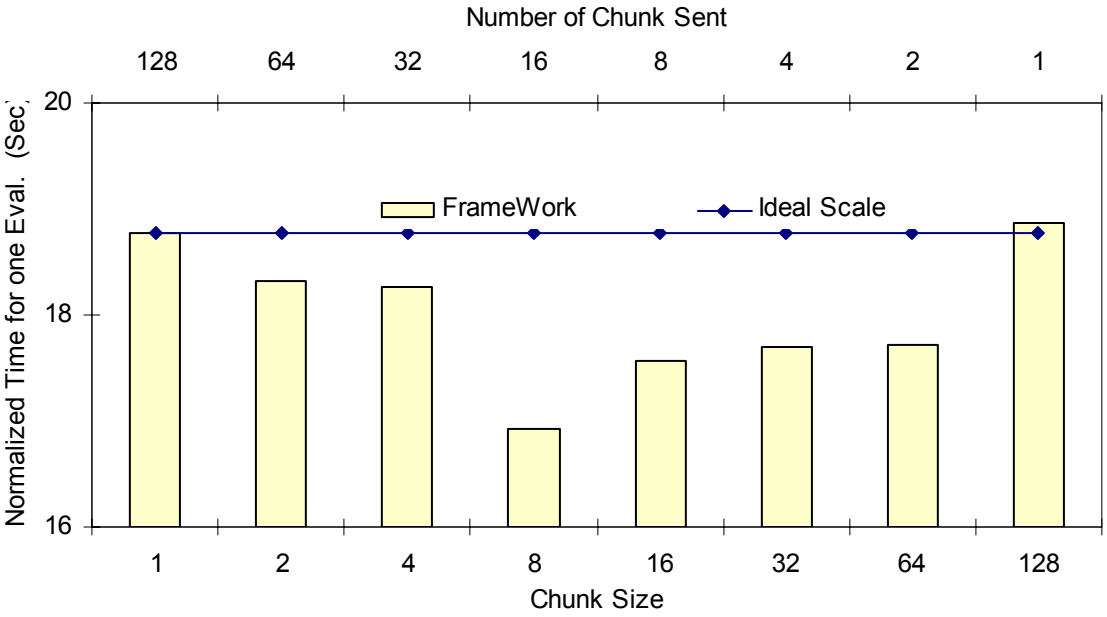

Figure 6-13B. The effect of Chunk Size Utilizing UC/ANL Site, Number of Groups = 1, Population Size = 128, Procs/Group =1:1

Appendix B-II Scaling Analysis

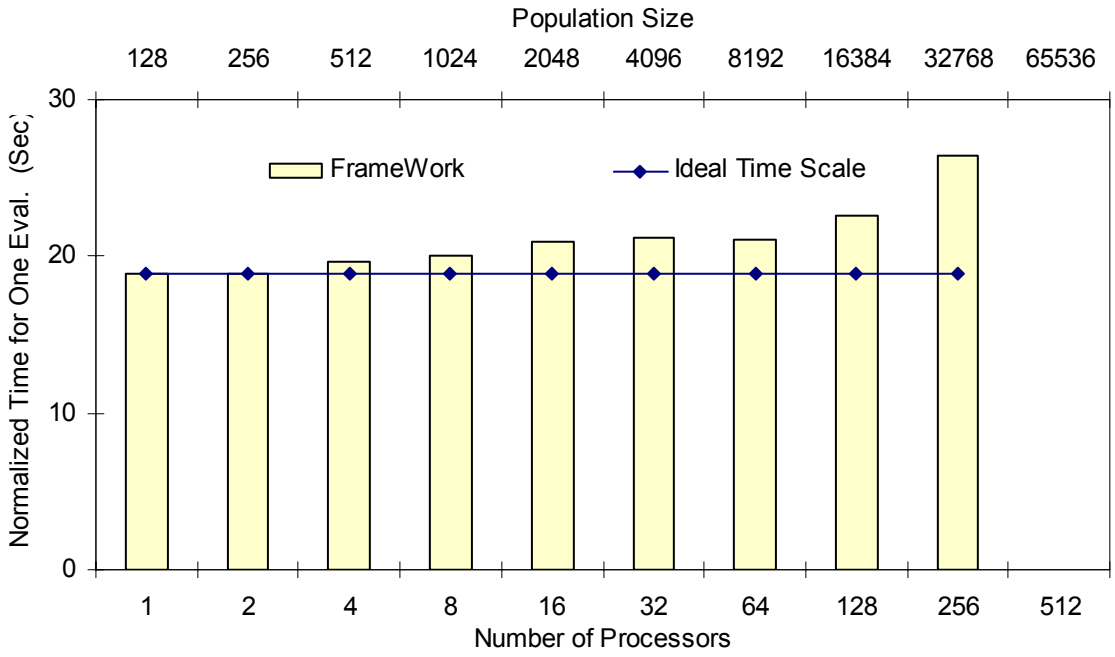

Figure 6-14A. Scalability Results Utilizing SDSC Site, Procs/Group =1:1, Tasks/Procs = 1:1, Number of Servers =1

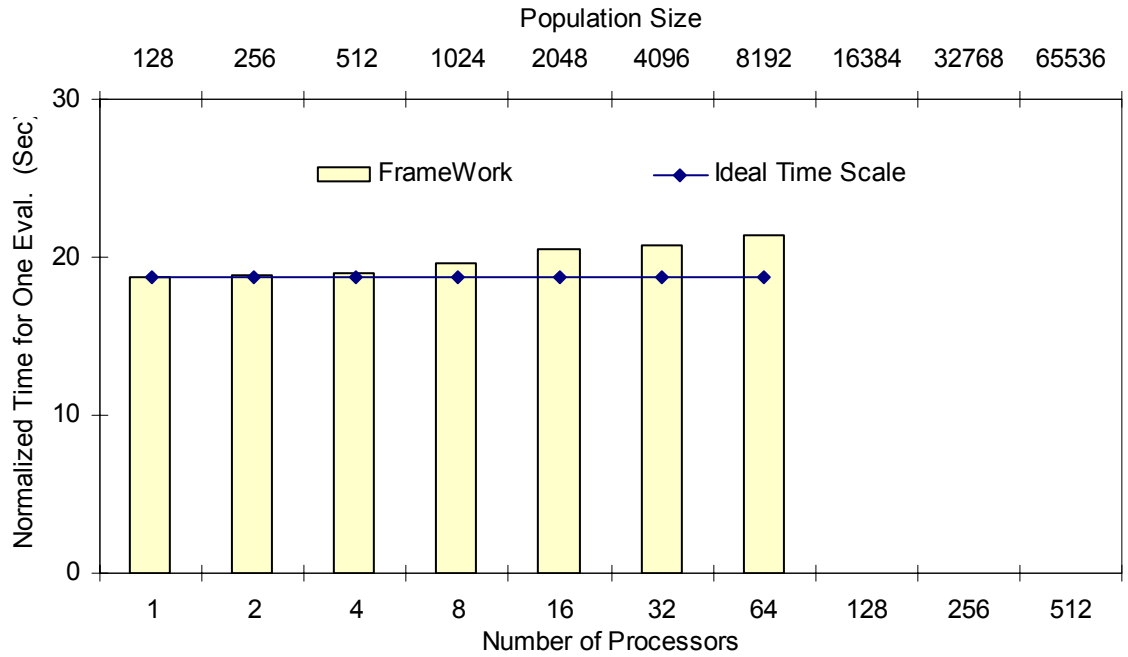

Figure 6-14B. Scalability Results Utilizing UC/ANL Site, Procs/Group =1:1, Tasks/Procs = 1:1, Number of Servers =1