EFFICIENT OPTIMIZATION OF COMPUTATIONALLY EXPENSIVE PROBLEMS USING A NEW PARALLEL ALGORITHM AND RESPONSE SURFACE BASED METHODS

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Amandeep Singh

May 2011

EFFICIENT OPTIMIZATION OF COMPUTATIONALLY EXPENSIVE PROBLEMS USING A NEW PARALLEL ALGORITHM AND RESPONSE SURFACE BASED METHODS

Amandeep Singh, Ph.D.

Cornell University 2011

This thesis concerns the development and implementation of efficient optimization algorithms for simulation based functions (real world problems) that are computationally expensive to evaluate.

The first contribution is a new parallel algorithm, RODDS for global optimization. RODDS algorithm is a stochastic heuristic global search algorithm, which effectively uses multi-core computers to reduce the computational expense of an optimization problem. The RODDS algorithm introduces the use of hyperspheres in candidate point generation. The optimization search is based on the concept of dynamically changing the dimensions perturbed to direct the search from a global to a local focus. Hyperspheres are used to prevent clustering of candidate points in optimization process to efficiently search the domain. We present numerical results on test problems as well as real world application problems from environmental engineering (groundwater management and watershed calibration) to document RODDS effectiveness when the computational budget is limited. RODDS algorithm achieves efficiencies greater than 1 for most applications which is very significant since implementation of parallel processing usually results in efficiency well below 1. We also present numerical results to show the efficiency of the use of hyperspheres in candidate point generation in RODDS by comparing with a parallel implementation without the

hyperspheres.

The next contribution is application of Radial basis function (RBF) based methods on computationally expensive optimization problems. We compare the performance of RBF methods with several popular global optimization algorithms (derivative based and heuristic) on two Groundwater superfund remediation sites (Pump and Treat system). These are two field sites Umatilla Chemical Depot (19,728 acres) and Blaine Ammunition Depot (48,800 acres). We present numerical results to indicate that RBF based methods are much more effective algorithms for computationally expensive groundwater problems, followed by a heuristic algorithm DDS. Under limited budget RBF based methods on average outperform traditional methods by an order of 100.

The third contribution is a new methodology of integrating a new integer value optimizer (Search over Integers with Tabu (SIT)) with continuous value optimizer (RBF based method) to solve fixed cost problems (which are Mixed Integer value problems, MIVP). Mixed integer value problems (MIVP) in general have large search domain thus the optimization process is computationally very expensive. This approach tries to take advantage of the fact that SIT is effective for optimizing discrete variables, while response surface method is much more efficient for optimizing continuous value variables. This study tries to limit the computational expense of such kind of problems by implementing a Sequential Response Surface method in conjunction with SIT. We present numerical results to show the effectiveness of integration methodology in comparison to Genetic Algorithm based NSGA-II (Deb et. al., 2003) and the MIVP optimizer, NOMAD (Abramson et. al. 2008). The SIT-RBF methodology is shown to be distinctly better than GA (SIT-RBF resulting in 150 times better solution than GA) under limited computational budget.

BIOGRAPHICAL SKETCH

Amandeep Singh was born in July 1980 in Srinagar, India to a Punjabi Sikh family. He received Bachelor of Technology degree from REC Jalandhar in Civil Engineering in May 2001. In May 2004, he received his Master of Technology from Indian Institute of Technology (IIT) Delhi. He worked as Engineer (Design) in Water Resources Division with RITES India Ltd., A Govt. of India Enterprise until July 2006.

In August 2006, Aman joined Ph.D. program in Civil and Environmental Engineering at Cornell University. During his studies at Cornell University he worked as a teaching assistant for different graduate and undergraduate level courses in the School of Civil and Environmental Engineering.

To my family.

ACKNOWLEDGEMENTS

First and foremost, I want to thank my advisor, Christine Shoemaker for her excellent support throughout my Ph.D. program. Enthusiasm and passion she has for her research was contagious and motivational for me at all times of my Ph.D. pursuit. I also want to acknowledge Philip Liu for his help over the years and for serving on my committee. I am indebted to Peter Diamessis who has served as a great mentor in addition to serving on my committee. I am grateful to Todd Cowan for his valuable suggestions during my stay at Cornell. I consider myself fortunate to join one of the best Environmental and Water Resources Systems group ever: Daniel Pete Loucks, Christine Shoemaker and Jerry Stedinger.

I would like to thank my parents and brother for their constant support and encouragement. My parents and brother Suman have been a never-ending source of motivation and this dissertation is as much a credit to their support as anything academic I picked up along the way. My better half Jyoti's enthusiasm and support helped me stay calm during my final stretch at Cornell.

Antoinne Espinet was not only my colleague during my stay at Cornell but he also became one of closest friends. His enthusiastic attitude towards everything helped me stay calm during my tough times at Cornell. Lastly, I had many fine student mentors while at Cornell, and mention Alex, Swarnavo, Saif, Raffel, Josh, Tom, Sue Nee, Yongsung, Ryan, Dillon and Manuel.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

With our ever increasing understanding of different physical, chemical and biological processes, we are able to devise improved mathematical models that describe these processes. The computational implementations of these mathematical models provide a basis to forecast and simulate different alternatives. The decision makers or engineers can then choose the best possible option or an "optimal policy". The need for the use of an optimization algorithm to choose the "optimal policy" arises when the range of parameter values and the number of parameter combinations is too large for analysts to enumerate or to test all possible alternatives. So for these kinds of problems optimization algorithms are a tool to guide the search to good solutions.

A typical framework used for these kinds of real world optimization problems is "Simulation-Optimization". The simulation model attempts to mimic reality using numerical approximations of process based equations, and the optimization model then tries to find the best set of input parameters (from domain) for the simulation model. Figure 1.1 depicts the framework used to choose an optimal policy. Here the upper box describes a loop that is repeated many times i.e. the simulation model is run many times, each time with a different set of input parameters. These different sets of input parameters are generated by optimization algorithm based on a particular criterion. Each optimization algorithm has its unique way of starting the search (for optimal solution) and generating candidate points for subsequent simulation runs.

The increasing complexity of the environmental models comes at the cost of increased computational expense i.e. CPU units (time) required for one such

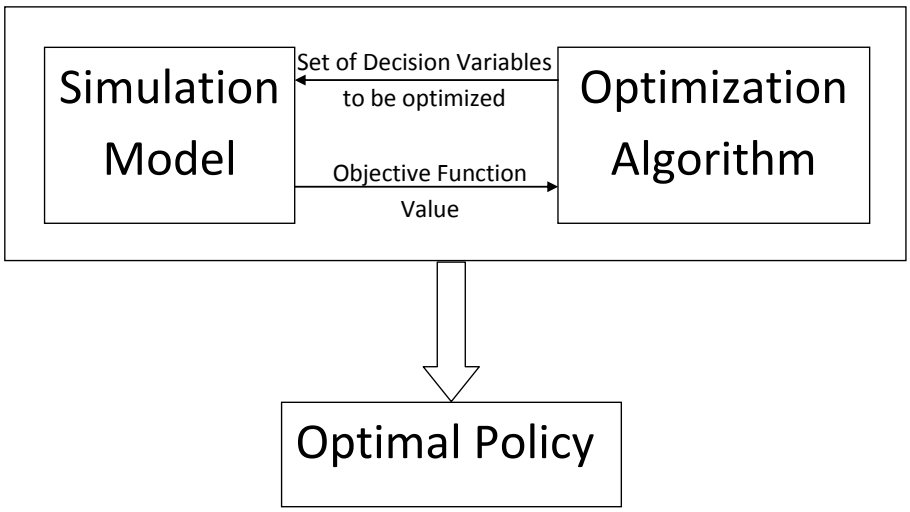


Figure 1.1: Simulation-Optimization for Optimal policy

simulation. The identification of an optimal policy or optimal parameters requires this expensive simulation to be repeated many number of times. Most of the earlier work done for addressing these kinds of simulation-optimization problems are based on the standard linear programming and global optimization tools like Genetic Algorithms (GA) etc. The problem with these methods is that they need simulation model to be run many many times. For a computationally expensive models, it becomes practically impossible to perform a large number of simulations e.g. hundreds or sometimes even thousands. Therefore, it becomes essential to use an optimization algorithm that minimizes the computational expense for an optimization trial by requiring relatively few simulations.

Another feature of many practical optimization problems is that they require "Global Optimization". Global optimization here refers to the class of problems with many local optimal solutions. Thus a good optimization algorithm for

these kind of problems should be able to efficiently explore the search space within a certain time frame. The focus of this thesis is to address the issue of efficient optimization of computationally expensive real world environmental problems. Stochastic RBF (Regis and Shoemaker (2007)), Simulated Annealing (Sadiq and Sait (1999)), Genetic Algorithms (Goldberg (1989)) are some global optimization algorithms, which have been shown to converge to optimal solution. But these proofs are based on the assumption of infinite model simulations, thus making their proof of convergence questionable for most practical applications (i.e. fixed number of model simulations).

To summarize, this thesis focuses on optimization algorithms for problems with the following characteristics

1. Only the objective function values from the simulation model are available to the algorithm i.e. No accurate derivatives are available inexpensively.
2. Function simulation is computationally expensive.
3. Test problems are Global optimization problems i.e. could have more than one local optimum.

The developed algorithms are tested on two groundwater EPA superfund sites. The total cost for cleaning up contaminated groundwater can exceed many millions of dollars and the whole cleanup process can take many years. The total cost and the time to do the cleanup depends on the pumping policy that is chosen for the whole management period. Applying optimization can significantly reduce the cost as compared to remediation plans formulated by trial and error simulations. Another reason to choose groundwater contamination sites for the developed algorithms is that the simulation is computationally expensive.

Hence the whole optimization process becomes computationally very intensive thus motivating the need for development of computationally efficient methods. Although this thesis focuses on groundwater and watershed models, the results are just as relevant to all environmental simulations of a computationally demanding models.

Computational expense of an optimization trial can be minimized in two ways, a) by efficiently using modern parallel computing tools and b) by using algorithms that minimize the number of model simulations to obtain a good solution point. Chapter 2 of this thesis reduces the computational wall clock time of an optimization trial by efficiently using the parallel computing tools. This chapter introduces a stochastic heuristic global search algorithm, Radii based Dynamically Dimensioned Search (RODDS). The RODDS algorithm tries to effectively use the multi-core machines to reduce the computational expense. The algorithm discussed is then applied for an optimal groundwater remediation design using a pump and treat method for determining the optimal pumping strategy. The method is applied to design an optimal pumping strategy for containment of multiple plumes at Umatilla Chemical Depot, Oregon. A simulation-optimization approach was used to determine an optimal policy. The performance of RODDS is then compared with a serial algorithm (DDS) and a parallel version of DDS. The study indicates that relatively good results can be achieved with considerable savings in time by using parallel RODDS implementation.

Chapter 3 focuses on minimizing the number of expensive function evaluations for an optimal solution by using a Response surface based optimization method. For two complex real groundwater sites (which require remediation

using a pump and treat system) the performance of the function approximation based methods is compared to traditional nonlinear and heuristic optimization methods. Independent trials of optimization runs are carried out to provide sufficient data for statistical testing. The algorithms are distinguished using different statistical procedures and metrics. Algorithms are tested on two superfund sites, Umatilla Chemical Depot, Oregon and computationally more expensive Blaine Ammunition Depot Hastings, Nebraska.

Chapter 4 addresses the issue of computational expense of a fixed cost problems (mixed integer nonlinear problem). This kind of problems arise in cases where installation costs must be accounted in addition to Operation and Maintenance cost. The complexity mainly arises due to discrete nature of the installation costs as these costs have a binary variable associated with them i.e. whether a facility should be constructed or not. Based on which of the chosen facilities are to be constructed there is an optimal operation policy. This optimal policy changes when a different set of facilities is chosen to be constructed. The resulting mixed integer nonlinear problem has an extremely large solution space. It becomes practically impossible to enumerate the number of possible installation configurations. This chapter introduces a new methodology which uses a new method SIT (developed in this thesis) with a Response Surface based method (Stochastic RBF) for solution of these kinds of fixed cost problems. The methodology developed for this study uses radial basis function as a surrogate for actual computationally expensive objective function simulation. The main focus of this study is to sequentially use the actual cost function evaluation information across different integer configurations to improve the accuracy of surrogate function approximations. The study compares the new methodology with GA based NSGA (Deb (2003)) and a mixed integer value optimizer, NOMAD on

two test and one real groundwater problem.

The increasing complexity of the environmental models comes at the cost of large number of parameters. Not all of these parameters can be determined by laboratory experiments. Also parameters estimated in the laboratory may not work well in field scale model. Often in such cases Automatic calibration is used. Automatic Calibration refers to use of an optimization algorithm to identify the parameter set that produces the best goodness of fit measure. Often the goodness of fit measure is non-convex function of parameters, thus derivative based methods are not suitable. RODDS algorithm introduced in chapter 2 is implemented for the first time on an calibration problem in chapter 5. RODDS algorithm tries to minimize the computational expense of calibration problem by effectively utilizing the multi-core machines. The algorithm is then tested on two real Townbrook and Cannonsville calibration problems.

We have developed a new parallel global optimization algorithm (RODDS) targeting computationally expensive simulation-based optimization problems in Water Resources. We have compared the performance of Response Surface based methods with the conventional methods. We have used the structure of RBF's to develop global optimization algorithm (SIT-RBF) to address fixed cost problems. We have tested the algorithms on a variety of applications (Groundwater Remediation and Watershed Calibration) and shown that they perform well, when relatively few expensive function evaluations are available.

CHAPTER 2

IMPLEMENTATION OF A NEW PARALLEL ALGORITHM FOR COMPUTATIONALLY EXPENSIVE GROUNDWATER MODELS: UMATILLA ARMY DEPOT, OREGON

2.1 Introduction

A typical framework used to model a real world optimization application is ”Simulation-Optimization”. The simulation model attempts to mimic reality using numerical approximations of partial differential equations and the optimization model then tries to find the best set of decision variables for the simulation model. In other words the ”Simulation-Optimization” approach tries to couple simulation models to optimization algorithms. The purpose of this could be model calibration or the design or management of facilities. In general the optimization part involves running the simulation model many many times, so in cases where one such simulation (e.g. a groundwater transport problem) is computationally expensive, the whole process becomes very CPU intensive perhaps to the point of being infeasible. Sometimes the whole process may run for weeks or even months. In other situations the entire optimization process needs to be repeated many number of times each time with a different set of parameters. This again could run for a long time. It is possible to decrease the computational time by using an highly scalable parallel algorithm. Many serial algorithms are not very efficient in that, there is not much reduction in the ”wall clock” time even when many processors are used. A computationally expensive Groundwater remediation problem is used as an example. In addition to high computational expense, a groundwater remediation problem has high cost so

the optimization is useful (Becker et.al. 2006).

This study introduces a parallel stochastic heuristic global search algorithm, Radii based Dynamically Dimensioned Search (RODDS). RODDS algorithm is developed to take advantage of commonly available parallel facilities ranging from individual multi-core machines to parallel clusters (TERAGRID resources). The new algorithm is inspired by Dynamically Dimensioned Search (DDS) algorithm (Tolson and Shoemaker, 2007). DDS algorithm is based on the concept of dynamically changing dimension perturbations to direct the search from global to local. DDS has worked very well in serial applications, but in this study it will be shown that RODDS works better than DDS as the number of processors increase. RODDS improvement over DDS is due to use of hyperspheres to avoid clustering the search points. The way to define the hyperspheres to be efficient over a range of optimization problems is a major aspect of the development of RODDS as discussed in later sections.

This paper is organized as follows. Section 2.2 introduces/explains the algorithm and its parameters. The optimization test problems and the groundwater bioremediation problem is explained in section 2.3. Section 2.4 goes over the comparison of the algorithms and the results. Results are then discussed in section 2.5. Last section 2.6 highlights the conclusions for this study. Appendix A shows the general structure of RODDS algorithm and explains the candidate point generation procedure (hyperspheres).

2.2 Methodology

The *Radii based Optimization using Dynamically Dimensioned Search* (RODDS) algorithm is a stochastic heuristic global search algorithm that was developed to reduce the computational expense of optimization problems by effectively utilizing the multi processors machines. The algorithm tries to locate good optimal solution points within specified number of function evaluations. The algorithm is inspired by serial algorithm DDS (Tolson and Shoemaker, 2007). As in serial DDS the RODDS algorithm searches globally at the start of the search and becomes a local search as the number of iterations approaches the maximum allowable number of function evaluations. The transition from global to local search is achieved by dynamically and probabilistically limiting the dimensional space of the neighborhood i.e. the neighborhood size for each processor decreases as a function of iteration number (as the search progresses). The candidate points are generated by perturbing the current best solution point in the randomly selected dimensions. The choice of these evaluation points (for all processors) depends on all previously evaluated points and the respective function values i.e. RODDS tries to stay away from all previously generated high-cost points. RODDS differs from DDS in a way that it uses hyperspheres of some radius (r) to prevent search points for being too close to each other. This factor is especially important for parallel optimization. Much of the focus of this research is on defining the radii so that the method is effective on a range of problems. The radius of the hypersphere (discussed in section 2.2.2) depends on the input parameters initial and final radius adjusted by a factor. Immediately below is the algorithm steps followed by an explanation/discussion of each of the step.

2.2.1 Algorithm : RODDS

Step-1. **Define Inputs**

a. Neighborhood perturbation size parameter, r (0.2 is default).

b. Initial and final radii's i.e. R_o and R_l.

c. Maximum number of function evaluations **by each processor**, m.

d. Number of dimensions, D.

e. Number of processors to be used, w.

Step-2. **Define Starting conditions** i.e. choose X_{best} i.e. the current best solution point and the corresponding function value F_{best} from a set of uniformly generated random points:

a. Generate w number of randomly spaced solution points $\{X^k | X^k \in R^n, k = 1, ..., w\}$, and each processor then evaluates the function at these points.

b. Set $F_{best} = Min(F(X^1), F(X^2), ..., F(X^w))$ and $X_{best} = X_{min}$, let X_{best} equal the point such that $F(X_{best}) = F_{best}$. Also set $F_{worst} = Max(F(X^1), F(X^2), ..., F(X^w))$.

Step-3. **Define hypersphere**

a. Calculate $\alpha = \left(\frac{R_l}{R_o}\right)^{\frac{1}{m}}$.

b. Initiate function evaluation counter, $i = 1$.

Each processor $(j = 1, ..., w)$ **performs step 4 through 8**

Step-4. **Define Neighborhood** $N(j)$ **for processor** j

a. Calculate probability of each decision variable being included in neighborhood $N(j)$ as a function of the current iteration count $P = 1 - \frac{Log(w*(i-1))}{Log(m*w)}$

b. Randomly select J of the D decision variables for inclusion in neighborhood i.e. For $d = 1, 2, ..., D$ decision variables, add d to $N(j)$ with probability

$$P' = P * \left(1 - \frac{(k-1)}{w}\right), \tag{2.1}$$

then $|N(j)|$ is the number of decision variables changed by processor j in iteration i.

c. If $N(j)$ empty, select one random d from $(1, ..., D)$ for $N(j)$.

Step-5. **Generate candidate solution point** $X^{new}(j)$ **from** $X^{best} = [x_1^{best}, ..., x_D^{best}]$.

a. Perturb the best solution point by normal random variable with zero mean and standard deviation σ in $|N(j)|$ dimensions i.e. selected dimensions from previous step

$x_c^{new} = x_c^{best} + \sigma * N(0, 1)$ if $c \in N(j)$ otherwise $x_c^{new} = x_c^{best}$

$X^{new}(j) = [x_1^{new}, ..., x_c^{new}, .., x_D^{new}]$

b. For all dimensions check for bound violation and reflect if necessary i.e. if x_c^{max} is the upper bound for c^{th} dimension, x_c^{min} is the lower bound for c^{th} dimension.

- If $x_c^{new} \leq x_c^{min}$; $x_c^{new} = min(x_c^{max}, (x_c^{min} + (x_c^{min} - x_c^{new})))$
- If $x_c^{new} \geq x_c^{min}$; $x_c^{new} = max(x_c^{min}, (x_c^{max} - (x_c^{new} - x_c^{max})))$

Step-6. **Implement hypersphere criteria** i.e. the current point $X^{new}(j)$ should be away from all previously evaluated points.

a. Set $maxdiff = (F_{best} - F_{worst})$.

b. Let Z be set of all previously evaluated points by all processors for $X^s \in Z$.

$$\left(\frac{\|X^s - X^{new}(j)\|}{\sqrt{n}}\right) \leq R_o * \alpha^i * min\left(1, \frac{log(1 + cost(X_s) - cost_{best})}{log(maxdiff)}\right) \tag{2.2}$$

If equation above is true go to step 4 otherwise go to step 7.

Step-7. Each processor evaluates the respective function value ($F(X^{new}(1))$) and passes it to primary processor.

Primary processor performs step 8 through 10.

Step-8. **Update the best solution if required**.

a. Set $F_{new} = Min(F(X^{new}(1)), F(X^{new}(2)), ..., F(X^{new}(j)), ..., F(X^{new}(w)))$.

b. Update current best solution if necessary i.e. If $F_{new} \leq F_{best}$ update the best solution $F_{best} = F_{new}$ and $X_{best} = X^{new}$.

Step-9. Update iteration counter (i) and

$F_{worst} = Max(F(X^{new}(1)), ..., F(X^{new}(j)), ..., F(X^{new}(w)))$.

Step-10. Check stopping criterion i.e. stop when i reaches m otherwise go to step 4.

The RODDS algorithm begins with the master processor executing the first three steps of the algorithm (i.e. basically setting up the run). Step-1 sets up the algorithm parameters and specifies the optimization problem. Algorithm parameters involve initial (R_o) and final radius (R_l), neighborhood perturbation factor (r) and maximum allowed function evaluations (m). Other inputs include the problem size i.e. the number of decision variables (D) and the number of

processors to be used for the run (w). For this study of starting points is an integer function of the number of processors, w for efficiency.

Step 2 involves defining the starting conditions. In step-2a the main processor generates a set of starting points. For this study the algorithm generates a set of randomly generated points, but the algorithm is flexible to start from a user specified set of points. The master processor then assigns the evaluation of the starting points to processors. Once function evaluation is done for the starting points, the control and the respective objective function values are passed on to the main processor. In step 2b the main processor collects all the information and initiates the best cost(F_{best}) to be the point (X_{min}) with minimum cost among the starting points. This least cost point (X_{best}) is used to generate the next set of candidate points in step-5. The initial set of points and the respective cost values are saved for hypersphere radii calculation in step-6. For efficiency in terms of memory used and to limit communication cost, only the main processor saves the solution point data (point and the cost) i.e. instead of saving multiple copies on different processors only one copy of the information is saved.

In step-3a alpha, α is calculated which is the rate at which the hypersphere shrinks. The hypersphere for the best point shrinks exponentially starting from initial radius (R_o) to the final radius (R_l) (discussed in next section). Step-3b sets up the counter (i) which keeps track of the number of function evaluations done by each processor and implements the termination criterion i.e. terminates the algorithm once maximum function (m) evaluations is reached. Later steps will loop back to step 4 since it is the first step not involved in the initial setup part. The execution part gets repeated until allowed number of function evaluations (m) is reached.

Step 4 and 5, deal with candidate point generation. Step 4a calculates the probability of a particular dimension to be included in candidate point generation. Step 4b builds up the neighborhood space (N) i.e. chooses dimensions to be perturbed. Choice for the size of this set ($|N|$) depends on the maximum allowed function evaluations, processor number and the desired efficiency. A factor in Step-4b (explained in next section) for processor number helps in exploring the solution space better i.e. choice of set size is ensured to be different for various processors in any particular iteration. Step-5 generates a set of candidate solution points. The candidate points are generated by perturbing the best solution in randomly selected dimensions (Step-5a). The perturbations are sampled from a Normal distribution of mean zero and unit variance. Step-5b checks for the bounds of the decision variables i.e. the constraints. In case of violation reflection method is used to implement the bound. Step 6 implements the hypersphere criterion (discussed in next section) i.e. the candidate points thus generated have to satisfy the radii criterion i.e. candidate points must lie out the hypersphere whose size depends on iteration number (i), maximum allowed function evaluations (m), current best cost value (F_{best}) and the function values at all previously generated points. If the candidate point happens to lie within the hypersphere, the whole generation procedure (Steps 4-6) is repeated until candidate points satisfy the radii criterion.

In step 7 the computationally expensive function evaluation of $F(X^{new}(j))$ is done for all the candidate points $X^{new}(j)$ $(for j = 1, ..., w)$. At this point each processor, including the main processor, evaluates the function. After function evaluation, for $F(X^{new}(j))$ done by the j^{th} processor, the control and the evaluated objective function value, is passed back to the main processor.

In the final stage (Step 8-10) of any particular iteration the master processor collects all the information i.e. function values at the candidate points and updates the best cost and the respective point if needed. Step 9 updates the counter (i) and the worst cost F_{worst} (if needed). Step 10 then sends the control back to step 4 incase the counter (i) is less then the maximum number of allowed function evaluations (m).

2.2.2 Algorithm Discussion

A parallel algorithm needs to generate more than one candidate points at any step for expensive function evaluation. These points must be systematically generated to explore the search space, otherwise lot of computation time would be wasted. The idea behind using hyperspheres is to keep the newly generated candidate points in step 5 of the pseudocode away from all previously evaluated expensive solution points. This helps in getting to a good solution point by helping the algorithm to either stay away from local minima or escaping local minima, if it is caught in one (local minima). A lot of effort in this study focussed in getting to the current form of hypersphere equation (equation 2.2). Various other forms were tried, the first one for the same purpose was with hypersphere radius starting with some initial radius (R_o) and exponentially going down to (R_l).

$$\left(\frac{||X_s - X_{new}(j)||}{\sqrt{n}}\right) \leq R_o * \alpha^i \tag{2.3}$$

The problem with this form of the equation is that it gives equal weight of all previously evaluated points. As per this equation the newly generated point has

to be equal distance away from a good as well as a bad solution point. This led to algorithm getting stuck in local minima. Various other forms of the equation were implemented and then tested.

$$\left(\frac{||X_s - X_{new}(j)||}{\sqrt{n}}\right) \leq R_o * \alpha^i * min\left(1, \frac{(cost(X_s) - cost_{best})}{maxdiff}\right) \tag{2.4}$$

The equation 2.4 was another form of the equation which performed well for some of the tested functions but it ran into trouble when objective function range ($F_{worst} - F_{best}$) is relatively big i.e. bigger range resulted in bigger $maxdiff$ making whole right side of equation very very small (hypersphere radii reduces to zero for all points). The current form of equation (equation 2.5) takes care of this problem by keeping the right side within reasonable limit.

$$\left(\frac{||X_s - X_{new}(j)||}{\sqrt{n}}\right) \leq R_o * \alpha^i * min\left(1, \frac{log(1 + cost(X_s) - cost_{best})}{log(maxdiff)}\right) \tag{2.5}$$

This hypersphere criterion is checked for all previously evaluated points. The right hand side of the equation goes to zero for the case of current best point i.e. $cost(X_s) = cost_{best}$ i.e. there is no restriction around the current best point. This helps the algorithm freedom to move to a better solution point even if it is close to current best point. Also the equation 2.5 reduces to equation 2.3 for worst or near worst points, assigning bigger hypersphere radii's to these points respectively. This essentially means that the final radii's (near the end of optimization run) for the worst or near worst points is the R_l, one of the input parameters. The figure 2.1 compares the performance of three different criterions explained above for Schoen function. Equation 2.5 outperforms the other two criterions for all the tested functions.

Equation 2.1 tries to ensure that the points generated by different processors at a particular iteration come from different $N(j)$'s. This equation assigns differ-

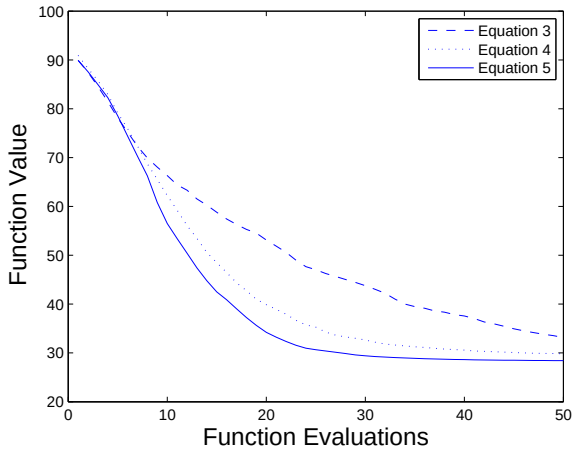


Figure 2.1: RODDS Hypersphere criterion comparison

ent probabilities to respective processors hence making sure that the neighborhood $N(j)$ selected by individual processors are different. This adds to diversity of candidate points thus resulting in better exploration of search space.

2.2.3 Algorithm Parameters

RODDS has three parameters to be tuned and they are hypersphere radii's (at start (R_o) and at the end (R_l) of the search) and the scalar neighborhood size perturbation parameter (r). The initial radius (R_o), multiplied by a factor, defines the minimum Euclidean norm distance between the set of initial points and the set of candidate points generated in first iteration, respectively. Similarly the final radius, multiplied by a factor, defines the minimum Euclidean norm distance between the set of points generated in last iteration and set of all previously generated points. Above mentioned factor (discussed in previous section) tries to assign high radii weight's to high cost points and low radii weight's to the near best solution points,at a particular iteration. All intermedi-

ate radii's are functions of starting radius, ending radius, the evaluation number and the respective objective function values of all previously evaluated points. These radii's (initial and final) depend on dimension of the problem and the expected range of the function value. The recommended values of initial and final radii's are 0.3 and 0.05 respectively, because these values give enough chance to the algorithm to stay away from expensive points. In case range of objective function is relatively small recommended value for the final radius is 0.01. The perturbation factor 'r' comes from the serial version on DDS; it defines the random perturbation size standard deviation as a fraction of the decision variable range. As recommended by Tolson and Shoemaker (2007) value of 0.2 is used. While generation candidate points, upper and lower bounds are imposed by reflection method. As found by Tolson and Shoemaker (2007) the reflection type boundary conditions allow decision variables to approach their optimal values easier and faster than the other methods. Lastly the maximum evaluations 'm' is also an algorithm input. In this study 'm' is maximum number of function evaluations done by each processor i.e. total number of function evaluations is 'm' times the number of processors used.

2.3 Optimization Test Problems

2.3.1 Ground Water

The demonstration site for this study was adapted from NAVFAC (*Naval Facilities Engineering Command*) technical report TR-2237-ENV, NAVFAC ,2004. The Chosen facility "Umatilla Chemical Depot" is located in northeastern Oregon.

Site Description

Umatilla Chemical Depot is a 19,728 acre military reservation established in 1941 as an ordnance depot for storage and handling of munitions. From the 1950s until 1960s the depot was used as an on-site explosives washout plant. The plant processed munitions to remove and recover explosives using a pressurized hot water system. The wash water from the plant was disposed in two unlined lagoons, from where the wash water infiltrated into the soil system. During this time, an estimated 85 million gallons on wash water was discharged to the lagoons.

Two of the most common contaminants, RDX (Hexahydro-1,3,5-trinitro-1,3,4-triazine, and commonly referred to as Royal Demolition Explosive) and TNT (2,4,6-Trinitrotoluene) are used as indicator contaminants. A pump-and-treat system was designed by the U.S. Army Corps of Engineers (USACE, 1996 and 2000) to contain and remove the RDX and TNT plumes. The USACE designed pump-and-treat system consists of three pumping wells and 3 recharge basins (shown in figure 2.2). One of the pumping wells and the infiltration basins were marked as inactive in the report. The cost of activating the inactive well is considerably less than the cost of installing a new well and there is no installation/activation cost associated with any other existing wells. Existing wells/basins and the inactive wells/basins play role in the cost definitions. The contaminated groundwater is extracted from the wells and then sent to GAC units, which remove the contaminants. The treated water is then discharged to the infiltration basins.

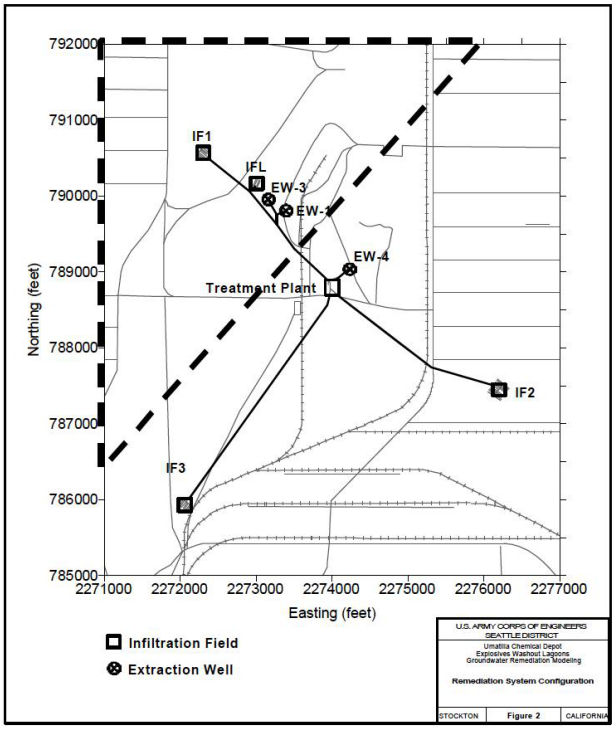


Figure 2.2: Umatilla Site Map : NAVFAC (*Naval Facilities Engineering Command*) technical report TR-2237-ENV,2004 showing location of extraction wells and infiltration fields (recharge basins)

Model Description

Groundwater flow is simulated using the MODFLOW code (Harbaugh, 2000). The MODFLOW-2000 1.18 version was used in Umatilla model. The study model has 125 rows, 132 columns and 5 layers, with variable grid spacing of 24.8ft - 647.9ft along the rows and 21.6ft - 660.7ft along the columns. Models layers are

- Layer 1: Alluvial aquifer, unconfined
- Layer 2: Silt and weathered basalt, convertible (confined/unconfined)
- Layer 3: Silt and weathered basalt, convertible (confined/unconfined)

- Layer 4: Silt and weathered basalt, convertible (confined/unconfined)
- Layer 5: Silt and weathered basalt, convertible (confined/unconfined)

The formulation-1 only focuses on contaminant transport in layer 1 of the model. The model boundary conditions for all four sides of the model domain were simulated as constant head. The Groundwater contaminant transport is simulated with MT3DMS (Zheng (1990)).

The model is structured into three phases i.e. input, simulation and output. The model takes Hydro-geological data, Domain-discretization data and the pumping data as input. The pumping data consists of pumping well locations with the respective pumping rate (to be optimized in this study). The formulation used for this study treats only the pumping rates as the decision variables for fixed well locations. After input phase the simulation is done using MODFLOW and MT3D. The study model simulates TNT and TCE (the two chosen parameters). And in the end objective function is calculated using the pumping data (input decision variables) and the simulated concentrations (C_{RDX}^{max} and C_{TNT}^{max}) at the end of simulation period. The model units are in feet and years.

2.3.2 Optimization Formulation

Objective Function

The objective of this formulation is to minimize the total costs (including both fixed capital costs and operation/maintenance costs) for the entire project duration i.e.

$$\min_{Q}(CCW + VCE(Q) + VCG(Q) + PenaltyCost(Q, C)) \tag{2.6}$$

Subjected to: $C_{RDX}^{max} \leq 2.1 ppb$ and $C_{TNT}^{max} \leq 2.8 ppb$

where,

- CCW (Q): Capital costs of new wells ($75,000 for installing a new well, $25000 for putting an existing unused well into service i.e. well 3 in this study)
- VCE (Q): Variable electric cost of operation
- VCG (Q): Variable costs of GAC units
- Penalty Cost (Q,C): For violating the concentration constraint, pumping constraint

where, Q=(Q_1, Q_2, ..., Q_{10}) is the respective well pumping rate (i=1,...,8 are pumping wells and j=9-10 are recharge basins with the last recharge basin getting a recharge of (Total pumping-Total recharge)) and $C(Q)$ is maximum contaminant concentration of TNT and RDX respectively (C_{RDX}^{max} and C_{TNT}^{max}) .

All the cost terms are computed in net present value (NPV) with the following discount function $NPV = \frac{cost_{iy}}{(1+r)^{iy-1}}$. Where, NPV is the net present value of a cost incurred in year iy with a discount rate of r=5%. The cost term is evaluated at the end of each year to account for annual discounting and to ensure that no costs are incurred after cleanup is achieved.

Constraints

The formulation includes following constraints that must be satisfied while the objective function is minimized

1. The modeling period consists of 1 management period of 4 years, with pumping rates kept constant throughout this period.
2. Cleanup must be achieved at the end of 4 years. In other words, the maximum concentrations (over space) of RDX and TNT ($C^{max}_{RDX}, C^{max}_{TNT}$ respectively) in model layer 1 must be less than their respective cleanup targets by the end of 4 years $C^{max}_{RDX} \leq 2.1ppb$ and $C^{max}_{TNT} \leq 2.8ppb$. C^{max}_{RDX} and C^{max}_{TNT} are computed by the MODFLOW-MT3D model for the given values of the vector Q of pumping and recharge rates.
3. The total pumping rate, after adjustment for the average amount of system uptime, cannot exceed 1300 gpm, i.e. the current maximum capacity of the treatment plant $\frac{1}{\alpha}Q_{total} \leq 1300$, where α is a coefficient representing the average amount of system uptime (α=0.9 for this study)
4. The pumping capacity of individual wells must not exceed 400 gpm in the less permeable portion of the aquifer and 1000 gpm in permeable portion $Q_{total} \leq 400$
5. The total amount of pumping must equal the total amount of injection through the infiltration basins within an error tolerance (implemented in this study by 3^{rd} recharge basin getting the balance of (Total pumping)-(Total recharge))

2.3.3 Optimization-Modeling Approach

The optimization formulation tries to do the cleanup by finding the optimal pumping and recharge rates for fixed locations (8 in number and 3 recharge basins). Figure 2.2 shows the location of existing pumping wells and the in-

filtration basins (Recharge basins). Thus, the optimization goal is to identify a pumping strategy that lowers the C_{max} values of RDX and TNT to their respective cleanup targets of 2.1 and 2.8 ppb in layer 1 within 4 years while satisfying all the pumping constraints. For this study the specific objective is to identify the best pumping rates on eight pumping wells and two recharge basins. The maximum allowed concentration constraint and the total pumping constraint are implemented by using the penalty functions hence the solution points not satisfying either of the two constraints (concentration and pumping) are penalized, which forces the algorithm to look for solution points that satisfy the above-mentioned constraints. The flow and transport model takes approximately 5 mins per simulation on a Pentium 2.2 Ghz computer.

2.3.4 Test Functions

RODDS algorithm runs in this study involved trial runs test functions ranging from 10- to 30- dimension, with 400 to 2000 total function evaluations per optimization trial. Four global optimization test functions (Ackley, Schoen, Griewank and Rastrigin) were chosen. These test functions are not expensive to evaluate but are multi-modal and possess some detectable trends or patterns typical to a global optimization problem. The test functions are not explicitly defined here instead summarized in table 2.1 with respective references.

Table 2.1: Test Functions for RODDS algorithm

Test Function	n	Domain	Reference
Ackley	10/30	[-1,3]	Ackley(1987)
Schoen	17	[-1,3]	Schoen
Griewank	10	[-2,2]	Griewank(1981)
Rastrigin	10/30	[-1,3]	Rastrigin(1974)

Note:-All these domains were normalized to be within [-1 1]

2.4 Results

2.4.1 Outline of Algorithm Comparison

The idea behind developing RODDS was to develop an efficient algorithm, which effectively utilizes available parallel resources, to identify good solution points for a global optimization problems where wall clock time is limited. Thus the experimental runs for the study were designed to test the algorithm with a fixed number of function evaluations under varying number of processors (i.e. as number of processors increase the effective wall clock time decreases). The results presented here compare the performance of RODDS with the serial version of DDS and a straight forward implementation of DDS algorithm in parallel (called DDS-PC). The main difference between the RODDS and the DDS-PC is that RODDS uses hyperspheres to stay away from local minima's for better exploration of search space. In order to compare the effectiveness of this method of candidate point generation, the plots include a version of DDS-PC.

RODDS algorithm was run on four different test problems with workers

varying from 4 to 32. The number of workers used for the trial runs varied from 4 to 32 i.e. 4, 8, 16 and 32 with respective decrease in wall clock time. Table 2.1 lists the test functions along with their respective decision dimensions. For the groundwater problem RODDS was run with 4, 8 and 16 processors. Algorithm comparison for all the plots was limited to RODDS (with n processors), RODDS with 1 processor, Serial DDS and DDS-PC, so as to clearly highlight the comparison between the serial and the parallel version.

To take into account the stochastic nature of the algorithms, we did 30 trials each of the three algorithms RODDS, DDS-PC and the serial DDS. There were 30 sets of starting points $x_{initial}$ values used in step 3 and each algorithm started a trial with each of the sets, to remove any bias from the starting values. The test functions for algorithm trial runs ranged from 6-dimensions to 30 dimensions global optimization problems. The maximum number of model or function evaluations per optimization run were chosen to vary from 400 to 1600, so as to cover the range of maximum ground water model evaluations used to solve an expensive global optimization problem (groundwater problem for this study). To improve computational efficiency, in all the cases the maximum allowed function evaluations were chosen as integer multiple of the respective number of processors used for the run. For algorithm convergence comparison, the best solution found is plotted against the respective number of objective function evaluations for each algorithm. In other words, for a particular algorithm, the average of the best solution found so far across all optimization trials is plotted against the respective number of function evaluations, which takes into account the fact that RODDS does 'w' (i.e. the number of processors) function evaluations per iteration.

The serial DDS is initialized to the best of $0.005 * m$ uniform random solutions (where m is the total number of DDS objective function evaluations to solve the problem) and the neighborhood perturbation size parameter, r, is set to the value of 0.2, as recommended by Tolson and Shoemaker (2007). RODDS algorithm is initialized with of 'w' uniform random solutions, where 'w' is the number of processors used for the trial. The neighborhood perturbation size parameter, r is set to the value of 0.2. Section 2.2.3 suggests a way to choose these radii's. All three algorithms i.e RODDS, DDS-PC and DDS stop only when the maximum function evaluation limit is reached.

Results are presented here in three sections. Section 2.4.2 compares the performance of RODDS algorithm with the serial version of DDS and the DDS-PC for some test problems. This section also compares the algorithm performance with respect to processors used i.e. it compares algorithm performance when the number of processors employed in the optimization run are increased or the wall clock time is decreased. This comparison was limited to RODDS only, to clearly highlight the effect of change in the number of processors used for the run. Section 2.4.4 discusses the performance comparison of RODDS on a ground water flow transport model (Umatilla). The last section 2.4.3 discusses and compares the results in terms of parallel computing metrics i.e. speedup and efficiency.

2.4.2 Test Functions

Figures 2.3 and 2.4 compare the performance of RODDS algorithm with the serial DDS (Tolson and Shoemaker, 2007), DDS-PC and RODDS 1-processor for 30-

dimensional Ackley and 17-dimensional schoen function. Plots here show only the results for two of the tested functions but performance on other tested functions is compared using tables for parallel metrics (2.4.3). Figure 2.3(a) compares the algorithm performance with 4 processors. Similarly figures 2.3(b), 2.3(c) and 2.3(d) compare the algorithm performance with 8, 16 and 32 processors respectively. Plots show the convergence plot for the objective function values with varying number of processors used for the respective runs, with the maximum allowed function evaluations being limited to 400 to 1600. For each plot, the function value (y -axis) is plotted against the specific i^{th} function evaluation (x-axis). This function value is averaged over 30 trial runs for the best solution found on or before the i^{th} function evaluation, respectively. For RODDS algorithm the one iteration equals 'w' function evaluations, where 'w' is the number of processors used for the run. In cases where maximum number is limited to 400, algorithm performance is only compared for 4,8 and 16 processors.

Figure 2.5 summarizes the performance of RODDS with respect to the number of processors used for the run. Each figure plots the results for the different test functions with respect to the respective number of processors used for that particular run. The maximum allowed function evaluations per processor were chosen such that each run does same amount of total function evaluations in total. These plots show how the performance of RODDS is affected by increase in number of processors.

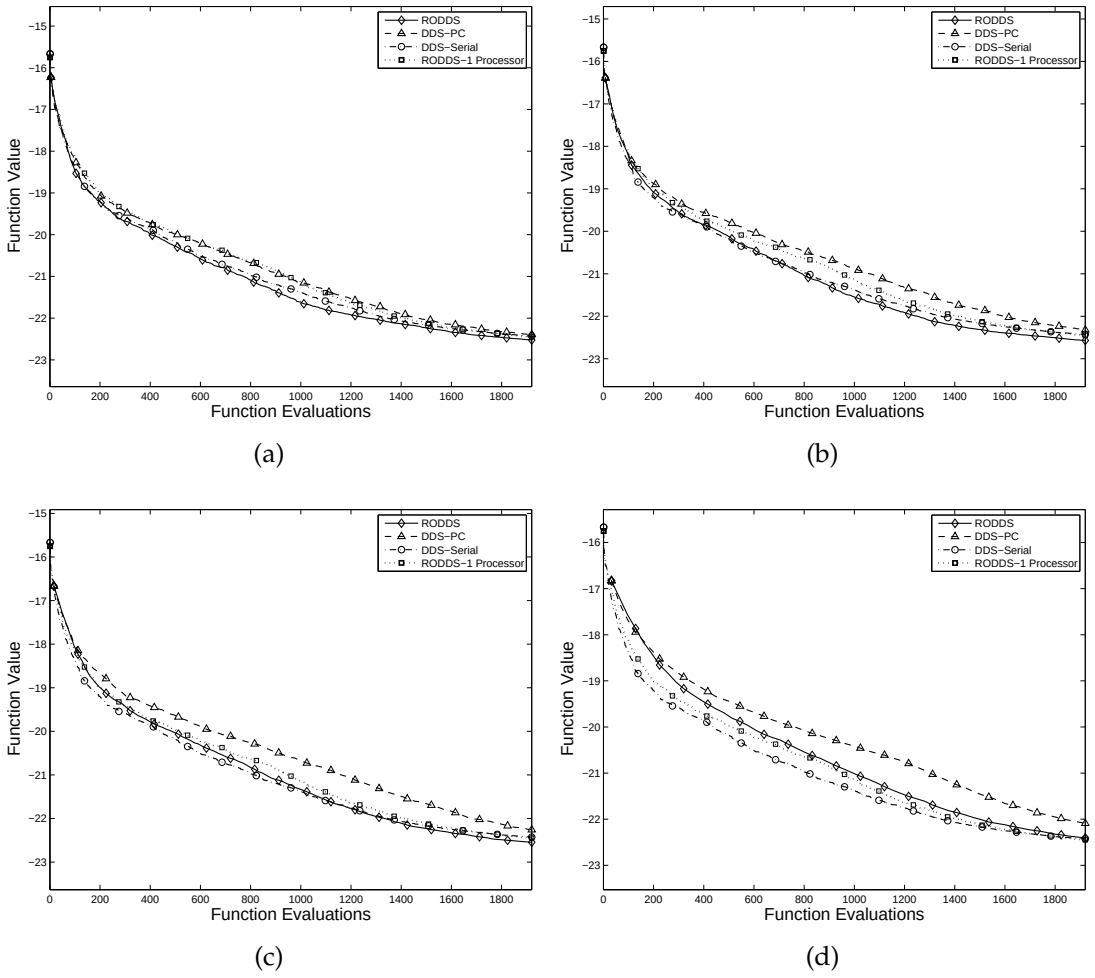


Figure 2.3: RODDS Results on 30-dimensional Ackley function: (a) with 4 Processors; (b) with 8 Processors; (c) with 16 Processors; and, (d) with 32 Processors

2.4.3 Parallel Metrics

The commonly used measures for the goodness of a parallel implementation are speedup and efficiency. Speedup is a measure of time gained i.e. by how much a parallel algorithm is faster than the respective serial algorithm. Efficiency metric reflects the processor utilization i.e. how well the work is distributed among the processors. Since RODDS is a stochastic algorithm and assumes a fixed number of evaluations, the calculation of these metrics is based on the objective function

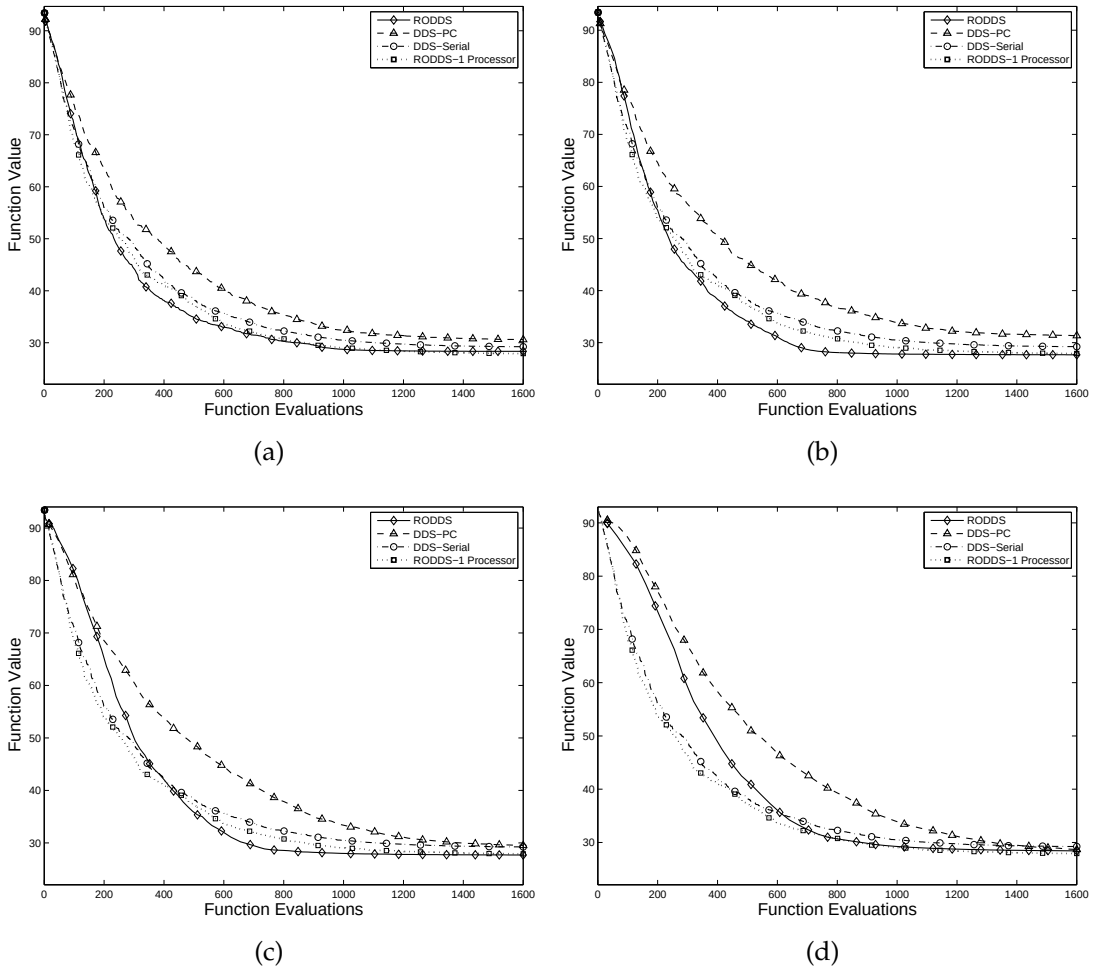


Figure 2.4: RODDS Results on 17-dimensional Schoen function: (a) with 4 Processors; (b) with 8 Processors; (c) with 16 Processors; and, (d) with 32 Processors

values obtained after a fixed number of iterations. So for this study, we modified these criteria's based on the runs it took a particular algorithm to reach within c% of the final answer obtained by DDS-serial averaged over 30 trials (explained below). This 'c' value is chosen based on results of DDS-PC.

Parallel metrics for RODDS and DDS-PC algorithms on the tested functions are listed using tables (Tables 2.8-2.7) respectively. Figure 2.6 explains the relation between wall clock times and modified metrics and the following tables for

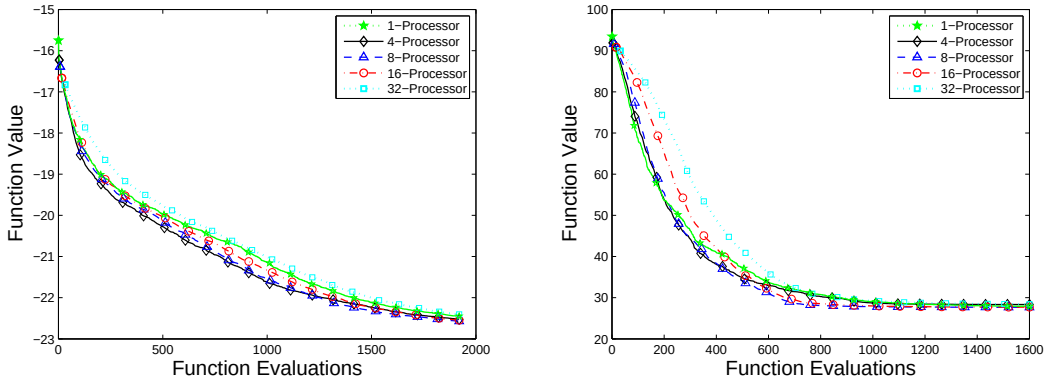


Figure 2.5: RODDS Processor performance comparison: (a) Ackley function; (b) Schoen function;

respective functions list the modified metrics. First column 'np', lists the number of processors used for a particular run. The next column lists the difference between the results of RODDS and DDS-serial at the end of optimization run averaged over 30 trials, D_f. WT_f is the wall clock units to get to Serial answer, whereas T_f is the total CPU units to get to Serial answer i.e. $nprocs * WT_p$. Next column lists the average number of function evaluation to reach within c% of the serial answer, for a respective algorithm. The '$Sp - c\%$', is the ratio of 'serial run to get to within c% of final serial answer' to 'total CPU units to get to serial answer' i.e. $\frac{T_s}{T_p}$, averaged over 30 trials. And the last column lists '$Ef - c\%$', the ratio of '$Sp - c\%$' to the respective number of processors used.

Table 2.8 lists the parallel metrics for Umatilla groundwater problem. Similarly Tables 2.3-2.7 list the parallel metrics for 10-dimensional Ackley, 30-dimensional Ackley, 17-dimensional Schoen, 10-dimensional Rastrigin,30-dimensional Rastrigin and 10-dimensional Griewank function respectively.

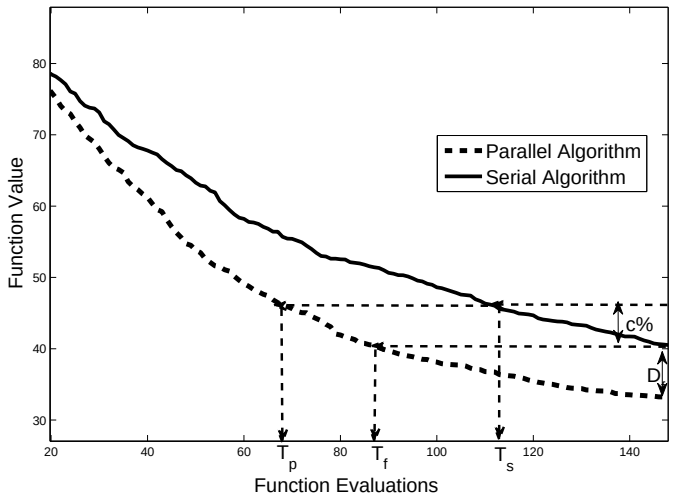


Figure 2.6: RODDS Metric Definitions

Table 2.2: RODDS results for 10-Dimensional Ackley Function. a) is for RODDS and b) is for DDS-PC

np	D_f	WT_f	T_f	$T_p(1\%)$	$Sp(1\%)$	$Ef(1\%)$
1	0.5	411	411			
4	0.5	103	412	81	5.83	1.46
8	0.5	55	440	42	11.24	1.41
16	0.4	33	528	24	19.67	1.23
32	0.2	21	672	16	29.50	0.92

a) RODDS

np	D_f	$T_s(1\%)$	$Sp(1\%)$	$Ef(1\%)$
1	0	472		
4	-0.1	68	3.93	0.98
8	-0.1	36	6.94	0.87
16	-0.8	36	13.11	0.81
32	-1	24	19.67	0.61

b) DDS-PC

Table 2.3: RODDS results for 30-Dimensional Ackley Function. a) is for RODDS and b) is for DDS-PC

np	D_f	WT_f	T_f	$T_p(1\%)$	$Sp(1\%)$	$Ef(1\%)$
1	0.4	1720	1720			
4	0.4	434	1736	369	4.17	1.04
8	0.6	206	1648	173	8.90	1.11
16	0.5	107	1712	91	16.92	1.06
32	-0.1	60	1920	60	25.67	0.802

a) RODDS

np	D_f	$T_s(1\%)$	$Sp(1\%)$	$Ef(1\%)$
1	0	1538		
4	-0.1	415	3.71	0.92
8	-0.4	223	6.91	0.86
16	-0.7	116	13.28	0.83
32	-1.5	Failed	-	-

b) DDS-PC

Table 2.4: RODDS results for 17-Dimensional Schoen Function. a) is for RODDS and b) is for DDS-PC

np	D_f	WT_f	T_f	$T_p(1\%)$	$Sp(1\%)$	$Ef(1\%)$
1	4	964	964			
4	3	230	920	225	5.72	1.43
8	5.3	83	664	82	15.68	1.96
16	5.2	44	704	43	29.91	1.87
32	2.8	31	992	30	42.87	1.34

a) RODDS

np	D_f	$T_s(1\%)$	$Sp(1\%)$	$Ef(1\%)$
1	0	1286		
4	-4	Failed	-	-
8	-7	Failed	-	-
16	-3	Failed	-	-
32	-3	Failed	-	-

b) DDS-PC

Table 2.5: RODDS results for 10-Dimensional Rastrigin Function. a) is for RODDS and b) is for DDS-PC

np	D_f	WT_f	T_f	$T_p(1\%)$	$Sp(1\%)$	$Ef(1\%)$
1	0.9	581	581			
4	0.8	139	556	123	5.03	1.19
8	0.9	67	536	58	10.67	1.33
16	0.6	39	624	33	18.76	1.17
32	-0.1	25	800	23	26.91	0.84

a) RODDS

np	D_f	$T_s(1\%)$	$Sp(1\%)$	$Ef(1\%)$
1	0	619		
4	-0.1	165	3.75	0.94
8	-0.2	90	6.88	0.86
16	-0.8	48	12.90	0.8
32	-2.6	Failed	-	-

b) DDS-PC

Table 2.6: RODDS results for 30-Dimensional Rastrigin Function. a) is for RODDS and b) is for DDS-PC

np	D_f	WT_f	T_f	$T_p(1\%)$	$Sp(1\%)$	$Ef(1\%)$
1	2	1464	1464			
4	1.9	374	1496	366	4.19	1.05
8	1.6	197	1576	185	8.3	1.04
16	0.3	99	1584	96	16	1
32	-1.3	50	1600	Failed	-	-

a) RODDS

np	D_f	$T_s(1\%)$	$Sp(1\%)$	$Ef(1\%)$
1	0	1536		
4	-2	Failed	-	-
8	-3.9	Failed	-	-
16	-7.1	Failed	-	-
32	-15.2	Failed	-	-

b) DDS-PC

2.4.4 Groundwater Problem

The groundwater flow contamination problem in section 2.3.1 is solved here using the RODDS, DDS-PC and DDS-serial. Figure 2.7(a) compares the perfor-

Table 2.7: RODDS results for 10-Dimensional Griewank Function. a) is for RODDS and b) is for DDS-PC

np	D_f	WT_f	T_f	$T_p(1\%)$	$Sp(1\%)$	$Ef(1\%)$
1	15.2	350	350			
4	12.8	81	328	82	4.8	1.2
8	7.4	46	376	47	8.38	1.05
16	5.4	23	384	24	16.4	1.03

a) RODDS

np	D_f	$T_s(1\%)$	$Sp(1\%)$	$Ef(1\%)$
1	0	394		
4	-2	Failed	-	-
8	-10	Failed	-	-
16	-20	Failed	-	-

b) DDS-PC

Table 2.8: RODDS results for 10-Dimensional Umatilla Problem. a) is for RODDS and b) is for DDS-PC

np	D_f	WT_f	T_f	$T_{p(10\%)}$	$S_{p(10\%)}$	$E_{f(10\%)}$
1	7.1	240	240			
4	5.3	76	304	41	6.46	1.6
8	4.8	44	352	33	8.00	1.05
16	6.5	22	352	15	17.67	1.1

a) RODDS

np	D_f	$T_{s(10\%)}$	$S_{p(10\%)}$	$E_{f(10\%)}$
1	0	265		
4	-4.7	73	3.6	0.9
8	-4.5	50	5.3	0.66
16	-5.9	30	8.83	0.55

b) DDS-PC

mance of RODDS with serial DDS (Tolson et. al.) and DDS-PC for 4, 8 and 16 processors respectively. Figure 2.7(d) compares the performance of RODDS with respect to the number of processors used for the run. Similar to test functions for each figure, the function value (y -axis) is plotted against the specific i^{th} function evaluation (x- axis). Each plot lists the wall clock time the respective algorithm took for the serial and the parallel run. Algorithm comparisons for the groundwater problem in Figures were limited to the RODDS, serial DDS and DDS-PC algorithms to in order to clearly highlight some general algorithm performance differences. Due to relatively high wall clock expense of the groundwater problem the number of trials were limited to 10. So the function value on y-axis is the average over 10 trial runs for the best solution found on or before the i^{th} function evaluation, respectively.

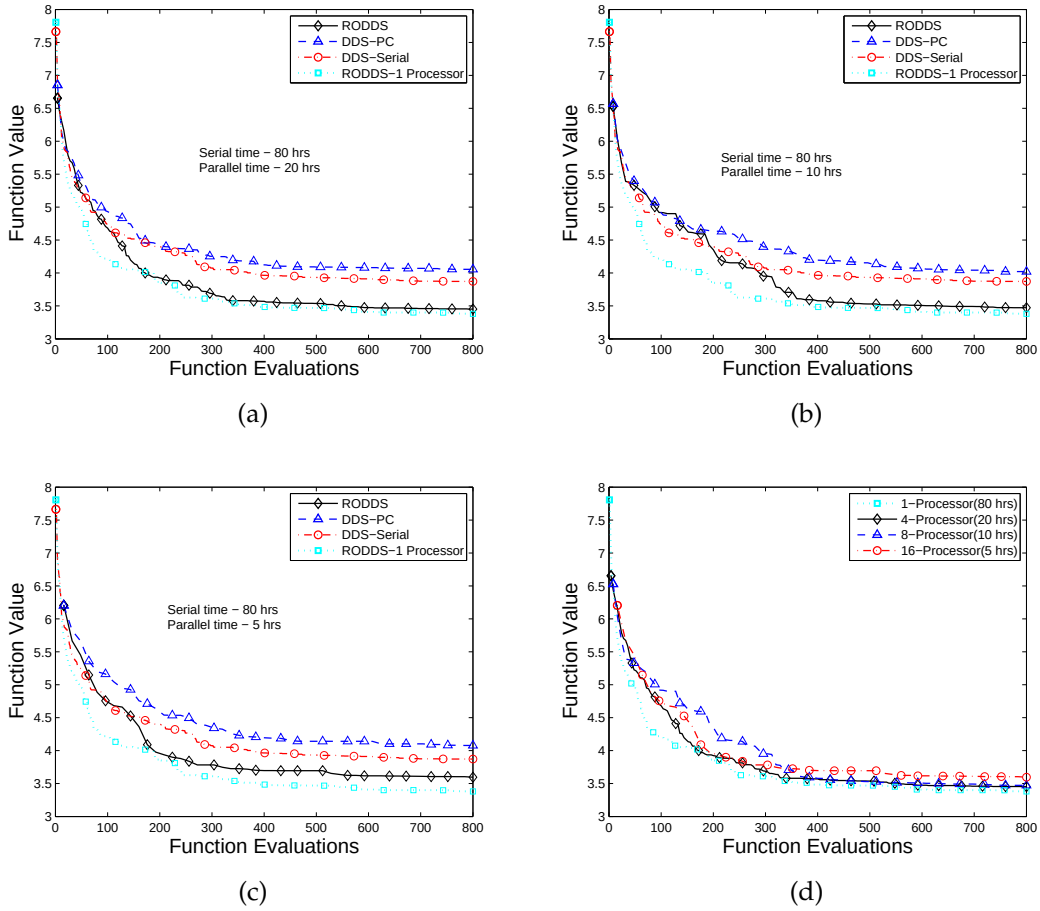


Figure 2.7: RODDS Results on Umatilla Groundwater problem: (a) with 4 Processors; (b) with 8 Processors; (c) with 16 Processors; and, (d) Processor performance comparison

2.5 Discussion

RODDS algorithm tries to find good solution points for global optimization problems within fixed computational budget, by efficiently utilizing the available parallel tools. Initially it is reasonable to expect that in cases where number of processors times maximum allowed function evaluations per processor is constant, as the number of processor increases the quality of results in terms of speedup and efficiency would decrease. But this study tries to stress on the

fact that efficient candidate point selection can greatly help maintaining good speedup and efficiency. This study tries to achieve or maintain good speedups and efficiency by combining the idea of hyperspheres with dynamic dimensioning. This study presents no evidence that RODDS is better algorithm than DDS, when the total available time is essentially unlimited (i.e. no bound for total allowable function evaluations).

Figure 2.3(a) shows that RODDS with 4 workers performs the best of all the compared algorithms on Ackley function. Figures 2.3(b), 2.3(c) and 2.3(d) compare the algorithm performance for Ackley function with 8, 16 and 32 workers respectively. Similarly Figures 2.4(a), 2.4(b), 2.4(c) and 2.4(d) compare the algorithm performance respectively for Schoen function. In general RODDS performed better than the serial DDS with workers up to 16 for all the tested functions. The difference in the average best solutions obtained is much more clear for 4 and 8 workers (i.e. the results were better with 1/4th or 1/8th the serial computational time). For 3 out of 4 test functions chosen the runs with workers up to 16 gave better results and the fourth one gave similar quality results. The results obtained with 32 workers were of the same quality as obtained by the serial DDS for all the test functions. Both the plots (2.4 and 2.3) for DDS-PC show the effect of not using the hyperspheres in point selection. The DDS-PC algorithm is more susceptible to stop (or get stuck) at local minima's. Thus the performance of DDS-PC is much more sensible to the number of processors used.

Tables 2.8-2.7 list the modified speedups and efficiencies for the different functions. Results on all test functions (tables 2.3-2.7) and the groundwater problem (Section 2.8) present evidence that RODDS maintains good efficiency

and speedups. In fact the results for all function including the groundwater problem show that RODDS gets better answers up to 8 processors (efficiencies> 1 and speedups> w). In other words RODDS gets better solution points with significantly less wall clock time e.g. 1/8th the serial time in case with 8 processors. In some cases the results with 16 were better than the ones from the run. And the results with 32 processors were similar to the ones from serial run if not better. The results for the groundwater problem in section 2.3.1 demonstrates (Figure 2.7) that RODDS locates good solution points with significantly less wall clock time. RODDS also tries to uniformly allocate the work among all the workers (processors), thus achieving a good efficiency.

One interesting thing to notice in RODDS results (tables 2.8-2.7) is that for some runs efficiencies with 16 workers is better than 8 workers which is not what is expected. Authors believe that this accounts from the fact that RODDS in parallel has the ability to move in multiple directions from the same point. Thus it is more adaptive to avoid local minima as serial DDS moves only in one direction at a particular iteration.

In comparison to DDS-serial the RODDS requires more tuning i.e. it has two more additional parameters. The additional parameters are initial and the final hypersphere radii's. Both of these parameters depend on the dimensionality of the problem and the expected cost range of the function. In case of high dimensionality and wide range the values of 0.45 and 0.2, for initial and final radii's are recommended respectively. For lower dimensional problems values of 0.25 and 0.1 are recommended. Using these hyperspheres helps RODDS avoid getting stuck in local minimas by forcing it to choose points away from expensive points (in terms of objective function).

2.6 Conclusion

Numerical results demonstrate that the RODDS algorithm is able to computationally efficiently use parallelism to get good results for the range of test functions and groundwater remediation example problem considered in this study. RODDS outperforms DDS-serial with 4,8 and 16 processors (thus resulting in speedups greater than 4, 8 and 16 respectively). For RODDS run with 32 processors, the solutions obtained are very close to the solutions obtained from serial DDS i.e. achieving near perfect speedup of 32. RODDS algorithm in general was able to reach efficiencies of greater than one also all of these efficiencies were much better as compared to DDS-PC, which show that the idea of hyperspheres helped RODDS to escape some local minima points and get to better solution point then DDS-PC. The value of RODDS is greatest for computationally demanding models where there is limited time to get results. In this study, the objective function or model evaluations were mainly limited to 1600 or fewer, in all the runs RODDS produced quality solutions (in terms of objective function value) in comparison with the DDS-serial. RODDS algorithm like DDS is quite simple, thus can be easily coded in whatever programming language of choice.

With the increasing availability of cheap multi-core machines and the simplistic characteristics of RODDS make it an attractive optimization tool for Environmental simulations. The speedups achieved by RODDS are significant as when coupled with a parallelized model (using parallel simulation), the resulting overall speedup can result in big time savings e.g. RODDS with 16 processors on a parallelized function using 10 processors can result in overall speedup of 160. The demonstration runs for this study limited the number of processors to 32 (for test functions with maximum of 1600 total function evaluations

i.e. 50 evaluations per worker) but the results can be generalized for functions which need far more total function evaluations. For these functions the number of processors used can be much greater than 32. Although this study focused on test functions and groundwater model, the results are just as relevant to all environmental simulations of a computationally demanding model.

BIBLIOGRAPHY

[1] Ackley, D.H., 1987. *A Connectionist machine for Genetic Hillclimbing*, Kluwer, Norwell, MA.

[2] Becker, D., Minsker, B., Greenwald, R., Zhang, Y., Harre, K., Yager, K., Zheng, C. and Peralta, R. (2006),*Reducing Long-Term Remedial Costs by Transport Modeling Optimization. Ground Water*, 44: 864875. doi: 10.1111/j.1745-6584.2006.00242.x.

[3] Griewank, A.O., 1981.*Generalized descent for global optimization, journal of Optimization Theory and Applications*, 34:11-39.

[4] Harbaugh, A.W., Banta, E.R., Hill, M.C., and McDonald, M.G., 2000, MODFLOW-2000, *The U.S. Geological Survey modular ground-water model - User guide to modularization concepts and the Ground-Water Flow Process*: U.S. Geological Survey Open-File Report 00-92, 121 p.

[5] The Mathworks, Inc. (2009a) *Parallel Computing Toolbox for Use with MATLAB: Users Guide*, version 1. The Mathworks, Inc., Natick, Massachusetts, USA.

[6] The Mathworks, Inc. (2009a) *Optimization Toolbox for Use with MATLAB: User's Guide*, version 3. The Mathworks, Inc. Natick, Massachusetts, USA.

[7] Naval Facilities Engineering Command (NAVFAC) technical report TR-2237-ENV.

[8] Rastrigin, L.A., 1974. *Systems of Extremal Control. Nauka*, Moscow. in Russian.

[9] Schnabel,R.B., *A view of the limitations, opportunities, and challenges in parallel nonlinear optimization*, Parallel Computing 21 6 (1995), pp. 875905.

[10] Regis, R. G., C. A. Shoemaker. 2007c. *Parallel radial basis function methods for the global optimization of expensive functions.* Eur. J. Oper. Res. 182(2) 514535

[11] Sait, Sadiq M and Youssef, Habib. *Iterative Computer Algorithms with Applications in Engineering IEEE Computer Society*, Los Alamitos, California.

[12] Schoen, F. 1993. *A wide class of test functions for global optimization*. J. Global Optim. 3(2) 133137.

[13] Tolson, B. A., and C. A. Shoemaker (2007), *Dynamically dimensioned search algorithm for computationally efficient watershed model calibration*, Water Resour. Res., 43, W01413, doi:10.1029/2005WR004723. Jan. 2007

[14] Tolson, B. A. (2005) *Automatic Calibration, Management and Uncertainty Analysis: Phosphorous transport in the Cannonsville Watershed*. PhD Thesis, School of Civil and Environmental, Cornell University, Ithaca, New York,USA.

[15] Zheng, C., 1990, *MT3D: A modular three-dimensional transport model for simulation of advection, dispersion, and chemical reactions in groundwater systems*, Report to the U.S. Environmental Protection Agency, Ada, OK, 170p.

[16] Zheng, C. and Wang, P.P., 1998, *MT3DMS, A modular three-dimensional multispecies transport model for simulation of advection, dispersion and chemical reactions of contaminants in groundwater systems*, Vicksburg, Miss., Waterways Experiment Station, U.S. Army Corps of Engineers. 238p.

CHAPTER 3

RESPONSE SURFACE AND HEURISTIC OPTIMIZATION METHODS APPLIED TO COMPUTATIONALLY EXPENSIVE GROUNDWATER CONTAMINANT TRANSPORT MODELS

3.1 Introduction

The total cost of cleaning up contaminated groundwater can exceed many millions of dollars and the cleanup duration can run into years. Two of the most commonly used methods are ”Pump and Treat system (P&T)” and bio-remediation. In pump and treat system extraction wells take out contaminated water from an aquifer and pump in clean water for hydraulic gradient control, while in bio-remediation wells inject electron acceptors. Total cost in both of these methods depends on pumping policy. Pumping policy here refers to the pumping well locations and the respective rates.

Numerical models are used to decide an optimal pumping policy, which minimizes the overall cost of cleanup. ”Simulation-Optimization” is the most common approach used for such kind of applications. Simulation model attempts to mimic reality using numerical approximations of partial differential equations that describe the transport of pollutants in response to pumping then the optimization model tries to find the best set of pumping rates to input to the simulation model. The ”Simulation-Optimization” approach basically tries to couple simulation models to optimization algorithms. The optimization process in general involves repeating the simulation process many many times. Thus if such methods are applied to a large scale problem, the optimization process becomes very computationally demanding. Various types of optimization algo-

rithms such as derivative based algorithms or heuristic algorithms have been used for deciding an optimal policy.

Given a deterministic continuous function $f(Q)$ where $Q = (Q_1, Q_2, ..., Q_n)$ and n is the number of continuous decision variables. The optimization problem is:

$$\min_{Q}(f(Q)) \tag{3.1}$$

Subjected to:-

$Q_{min} \leq Q_j \leq Q_{max}$ for $j = 1, ..., n;$

$G_i(Q) \geq 0 \;\; for\; i = 1, ..., M$

Here $G_i(Y)$ is nonlinear constraint that could possibly have a multi-modal surface. For example if $f(Q)$ is a simulation model and $\theta_i(Q)$ is an output of the simulation (say contaminant concentration) then $G_i(Q) = C^{max} - \theta_i(Q) \geq 0$ i.e. value of $\theta_i(Q)$ is equal to or less than C^{max}. In case when the nonlinear constraint is computationally expensive to evaluate it can be included in objective function (equation 3.1) using penalty function approach.

This paper describes the first comparison of recently developed global optimization models to design remediation of a large groundwater Umatilla site and the even more computationally expensive simulation model for Blaine a 48800 acre facility. Most of the earlier work done for solving these kinds of optimization problems are based on the standard linear programming and global optimization tools like Genetic Algorithms (GA) etc. Optimal control models have been developed by Ahlfeld et. al. (1996), Atwood et.al. (1985) Shoemaker et.al. (1992) that optimize pumping rates at the wells to minimize the overall cost. Several authors have used heuristic global optimization tools: El Har-

rouni et al. (1996), Wang and Zheng (1997), Aly and Peralta (1999b); Becker et.al. (2006); Espinoza et. al. (2005) used genetic algorithms (GA); Karatzas and Pinder,1992; presented an outer approximation method; Aral and Guan (1996) used a differential GA. Several other authors have tried to solve for optimal pumping strategy. The problem with some of these methods is that they need simulation model to be run many many times. In cases where one such simulation is computationally expensive the whole optimization process becomes very expensive. In order to reduce the computational time, several authors have tried to implement hybrid approaches such as combining artificial neural networks (ANNs) with GA's (Solomatine (1998), Dibike et al. (1999) and Rao and Jamieson (1997)). The aim of this study is explore the implementation of Radial Basis Function (RBF) based methods and a heuristic algorithm DDS developed by Tolson and Shoemaker, 2007; for solving these kind of problems.

In this study the performance of Response surface methods (gradient-free methods) is compared against traditional gradient based and heuristic algorithms on computationally expensive groundwater models. The aim is to identify algorithms that consistently find good solution points with modest computational effort. Multiple optimization trials of each algorithm were run to statistically measure the variability of algorithm performance. In practice it is generally not feasible to conduct multiple trials so statistical comparison is used to choose the best algorithm for the problem. The algorithm performance is studied under fixed computational time i.e. by fixing the number of allowed maximum function evaluations.

This study is organized as follows. Section 3.2 introduces/explains the various algorithms and their parameters. The two application problems i.e. the

groundwater remediation problems Umatilla Chemical Depot and Blaine Ammunition Depot are explained in section 3.3. Section 3.4 describes the performance comparison of the various algorithms tested and the statistical results. Results are then discussed in section 3.5. Last section 3.6 highlights the conclusions of the study.

3.2 Algorithm Description

The goal of this paper is to examine a range of global optimization methods on some field scale groundwater remediation problems. The study includes a surrogate model optimization method Stochastic RBF (Regis and Shoemaker, 2007, 2009). It also compares their performance to other commonly used methods. The study also involves a class using a systematic multistart method (Multi-Level Single Linkage, MLSL) in conjunction with local optimization methods. Two classes of evolutionary algorithms population based and non-population based metaheuristics were considered for the optimization run. The heuristic algorithms used for the study were Simulated Annealing, Real-coded Genetic Algorithms and Dynamic Dimensioned search (DDS). For all heuristic algorithms the algorithm parameters were chosen based on previous research or based on the author's experience. No detailed analysis were made to find optimal algorithm parameters since this would require significant amount of computational time and resources in terms of groundwater model simulations. All the listed algorithms were tested on Umatilla model, whereas only selected algorithms were tested on the computationally more expensive Blaine model.

3.2.1 Multi Start Local Optimizers for Global Optimization

A major difference between a local optimization algorithm and a global optimization algorithm is that a local optimization method will stop searching when it finds a local minimum, which might or might not be global minimum. One approach to incorporate a local optimizer into global optimization is to restart the local optimization algorithm at a new starting point, after the algorithm stops at a local minima. This restarting operation is repeated each time an algorithm finds a local minima. The choice of starting points is one of the main criterions for the performance of local search algorithm. So it is important to choose the new starting points effectively. Otherwise the algorithm could reach the same local minima more than once thus resulting in waste of computational effort and time.

Multi-Level Single Linkage (MLSL) developed by Rinnooy Kan & Timmer (1987) is one method for restarting local optimizers, so that a local optimizer can be used for global optimization problems. Multi-Level Single Linkage uses a critical distance criterion for selecting starting points. Each iteration of MLSL involves: (1) generating a uniform random sample of N points from the search space and adding it to the sample from previous iterations; (2) evaluating the objective function at the new sample points; (3) selecting some fraction of the sample points with the best objective function values; and (4) starting a local optimization run at each selected sample point, unless it has been used as a starting point from the previous iteration, or if there is another sample point with a lower function value that is within some critical distance of the selected point. For more details reader is referred is Rinnooy Kan & Timmer, 1987. When used with a optimization that is proven to converge to the global optimum,

MLSL (by Rinnooy Kan & Timmer, 1987) was proven to converge to the global optimum.

3.2.2 Local Methods with MLSL

Derivative based method

Derivative based Optimization methods are the oldest nonlinear methods and will only find local minima. But calculation of derivatives for complex large scale simulation models is time consuming, possible inaccurate, and in some cases impossible (for example if there is no source code available). Finite differences can be used as an alternative to calculate derivatives when well defined analytical structure is not available, but is the computationally very expensive as function needs to be evaluated more than once for one derivative calculation. Sequential Quadratic Programming (SQP) with finite difference derivatives was used in this study as it is a standard and well-known derivative-based optimization method. SQP is implemented in the FMINCON solver in the MATLAB Optimization Toolbox (The Mathworks, 2010), where derivatives are calculated using finite differences.

Implicit Filtering

Implicit Filtering (Gilmore & Kelley, 1995) is a modified form of a derivative-based method that uses finite differences to estimate derivatives. Implicit Filtering uses a unique way to select the step sizes for the finite differences, to accommodate roughness in the objective function (e.g. those arising from numerical

solutions to equations in a simulation model). Implicit Filtering varies the step sizes for the finite difference approximation. MATLAB Implicit Filtering code provided by Kelley (1999) is used.

Pattern Search

Pattern Search (Torczon, 1997) is a derivative-free local optimization method where each iteration consists of searching a set of points called a pattern which expands or shrinks depending on whether any point within the pattern has a lower objective function value than the current point. At each step, the algorithm searches a set of points, called a mesh, around the current best point. The algorithm forms this mesh by adding the current point to a scalar multiple of a fixed set of vectors called a pattern. If the algorithm finds a point in the mesh that improves the objective function at the current point, the new point becomes the current point at the next step of the algorithm. The mesh has a rigid structure at one iteration that could require extra evaluations of computationally expensive functions. Here, we use the implementation of pattern search available in the MATLAB Genetic Algorithm and Direct Search Toolbox (2009).

3.2.3 Global Optimization Methods

Simulated Annealing

Simulated Annealing (SA) is a random-search technique which exploits an analogy of an global optimization problem with the way a metal cools and freezes into a minimum energy state (crystalline structure, the annealing process). A

cooling schedule proposed by Aarts et al.1989, is employed in this study. The schedule uses a statistical analysis of the problem and adjusts the schedule to make the execution time polynomial. For a detailed implementation of SAs the reader is referred to Sadiq and Sait (1999).

Genetic Algorithm

Yoon and Shoemaker (2001) showed that real coded Genetic algorithms (GA) outperformed the binary coded genetic algorithms for optimal policy design of groundwater problems. A real coded GA was used with a population size of 20, one point crossover and mutation probabilities of 0.95 and 0.1 respectively was used in this study. Earlier work on groundwater management optimization (Willis (2001)) showed that GA with elitism is more effective than the conventional GA. For this study two elite members of each population were allowed to survive through to the next generation. For a detailed implementation of GAs the reader is referred to the book by *Goldberg* (1989).

Dynamic Dimensioned Search

Dynamic Dimensioned Search (DDS) algorithm was developed by Tolson and Shoemaker (2007). The DDS algorithm tries to minimize the computational expense of an optimization problem and tries to locate good optimal solution points within specified number of function evaluations. DDS algorithm searches globally at the start of the search and becomes a local search as the number of iterations approaches the maximum allowable number of function evaluations. The transition from global to local search is achieved by dynami-

cally and probabilistically limiting the dimensional space of the neighborhood i.e. the number of dimensions decreases as a function of iteration number (function evaluations), as the search progresses. The candidate points are generated by perturbing the current best solution point in the randomly selected dimensions.

Shuffled Complex Evolution

Shuffled Complex Evolution (Duan et al.1993) has been used widely for watershed optimization and has also been used for groundwater management optimization (Eusuff and Lansey (2004), . Originally SCE is designed to be run for enough model simulations (relatively computationally cheap functions) that the method has converged to the optimal solution. Algorithm developers developed algorithm for cases, that run very quickly so that many thousands of simulations were feasible. The goal of this paper is obtain a close-to-optimal solution using relatively few simulations. Hence, this study tries to accomplish a different goal to that for which SCE was originally developed. In the numerical experiments, we used the Matlab SCE code written by Q. Duan downloaded from MATLAB Central (http://www.mathworks.com/matlabcentral/).

3.2.4 Response Surface based methods

Response Surface based methods use a surrogate mathematical model as an approximation for the computationally expensive optimization objective to guide the search for suitable parameters. The idea is to fit an approximation to the objective function values from prior generations. The function approximation

algorithms used in this paper use Radial Basis Functions (RBF) (Buhmann, 2003; Powell, 1999) to approximate the expensive objective function. The purpose of using this RBF approximation is to reduce the computational expense of an optimization problem by allowing the RBF approximation to screen out candidate points which are unlikely to be highly fit before the actual simulations are done. Four response surface based methods are tested in this study Controlled Gutmann RBF (CGRBF, Gutmann (2001)), Evolution Strategy with Global RBF approximation (ESGRBF, Regis and Shoemaker (2004)), Multistart Local Metric Stochastic RBF (MLMSRBF, Regis and Shoemaker (2007)) and Global Optimization by Radial Basis function Interpolation in Trust Regions (GORBIT, Wild and Shoemaker (2009)).

For an optimization problem, if $x_1, x_2....x_n$ are the previously evaluated set of parameters, a cubic RBF interpolation model $s_n(x)$ that approximates the objective function has the form

$$s_n(x) = \sum_{i=1}^{n} \lambda_i \phi(\|x - x_i\|_2) + b^T x + a \tag{3.2}$$

where, $\lambda_1, \lambda_2, ...\lambda_n \in R, b \in R^d, a \in R, \phi$ is a radial function and $\|.\|$ is the Euclidean norm. The coefficients of the above model are chosen such that the interpolant passes through all the design points. The way to choose the experimental design points $x_1, x_2....x_n$ differentiates the algorithm suggested by *Gutmann* and CGRBF i.e. *Gutmann* (2001) evaluates the costly function at the corners of the domain D, so that there are 2^d points, where d is the dimension. This becomes too expensive for higher dimensional models. *Regis and Shoemaker* (2007) suggested the use of a Latin Hypercube Experimental design (LHD) for fitting the initial response surface. For d decision variables $\frac{(d+1)(d+2)}{2}$ symmetric Latin hypercube design points were used for initial surface. For a detailed mathematical description of the algorithm the reader is referred to the paper

by Gutmann (2001). MLMSRBF (Regis and Shoemaker (2007)) differs from CG-BRF in the way the next evaluation point is chosen. The next evaluation point in this case is chosen to be best point from a set of randomly generated candidate points. For algorithmic details reader is referred to the work by *Regis and Shoemaker, 2007*. ESGRBF algorithm couple RBF surrogate methods with evolutionary strategy. ESGRBF algorithm is a variant of ESBRF by *Regis & Shoemaker (2004)*. The difference between the ESRBF and ESGRBF algorithms is that the former uses local RBF approximations while the latter uses global RBF approximations (i.e. uses all data points to fit the RBF model). In each generation, of the generated offsprings only those are chosen which are most likely to fit (using RBF approximations). For more details on ESGRBF the reader is referred to *Regis and Shoemaker, 2004*. GORBIT (Wild and Shoemaker, 2009) attempts to implement trust regions within the RBF framework.

3.3 Groundwater Remediation Sites Description

The demonstration sites for this study were adapted from NAVFAC (*Naval Facilities Engineering Command*) technical report TR-2237-ENV, NAVFAC 2004. The two chosen sites are Umatilla Chemical Depot and Blaine Ammunition Depot.

3.3.1 Umatilla Chemical Depot

Site History

The Chosen facility "Umatilla Chemical Depot" is located in northeastern Oregon. It is a 19,728 acres military reservation established in 1941 as an ordnance depot for storage and handling of munitions. From the 1950s until 1960s the depot was used as an onsite explosives washout plant. The plant processed munitions to remove and recover explosives using a pressurized hot water system. The wash water from the plant was disposed in two unlined lagoons, from where the wash water infiltrated into the soil system. During this time, an estimated 85 million gallons on wash water was discharged to the lagoons.

Two of the many contaminants, RDX (Hexahydro-1,3,5-trinitro-1,3,4-triazine, and commonly referred to as Royal Demolition Explosive) and TNT (2,4,6-Trinitrotoluene) are used as indicator parameters. A pump-and-treat system was designed by the U.S. Army Corps of Engineers (USACE, 1996 and 2000) to contain and remove the RDX and TNT plumes (Figure 3.1). In this study the objective to find the optimal pumping rates and recharge rates for the locations suggested by NAVFAC (*Naval Facilities Engineering Command*) technical report TR-2237-ENV, NAVFAC 2004. The contaminated groundwater is extracted from the wells and then sent to GAC units, which removes the contaminants. The treated water is then discharged to the infiltration basins.

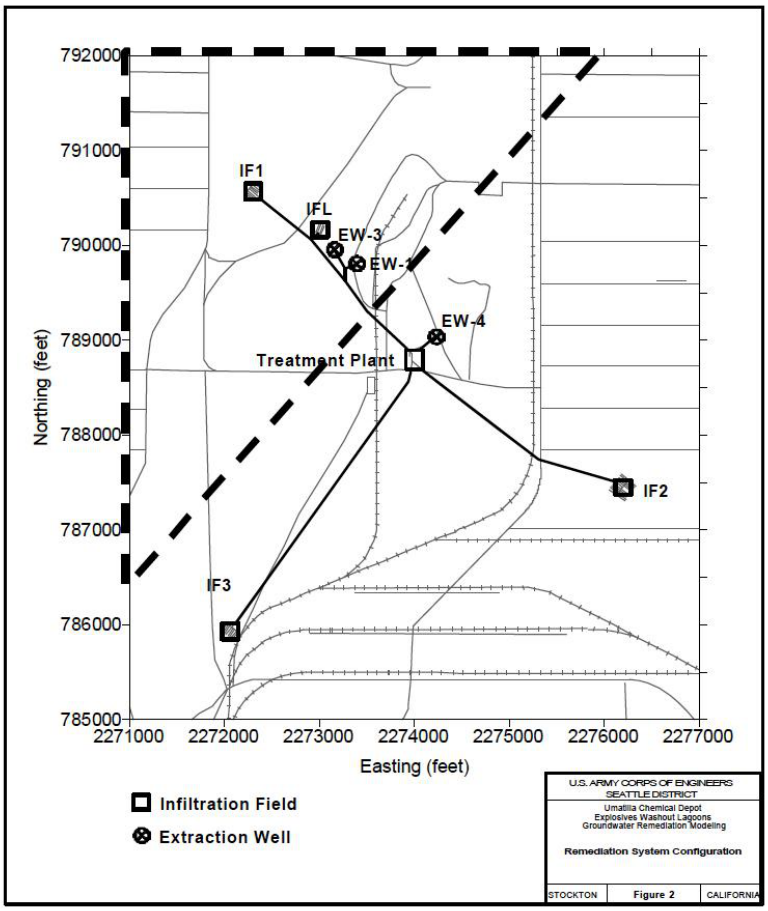


Figure 3.1: Umatilla Site Map : NAVFAC (*Naval Facilities Engineering Command*) technical report TR-2237-ENV, 2004 showing location of extraction wells and infiltration fields(recharge basins)

Model Description

The study simulates the groundwater flow using the MODFLOW (Harbaugh et.al., 2000) code. The MODFLOW (Harbaugh et.al., 2000) code is maintained by U.S. Geological Survey. The study model has 125 rows, 132 columns and 5 layers, with variable grid spacing of 24.8ft - 647.9ft along the rows and 21.6ft - 660.7ft along the columns. Models layers are

1. Layer 1: Alluvial aquifer, unconfined

2. Layer 2: Silt and weathered basalt, convertible (confined/unconfined)
3. Layer 3: Silt and weathered basalt, convertible (confined/unconfined)
4. Layer 4: Silt and weathered basalt, convertible (confined/unconfined)
5. Layer 5: Silt and weathered basalt, convertible (confined/unconfined)

The formulation considered in this study only focuses on contaminant transport in layer 1 of the model. The model boundary conditions for all four sides of the model domain were simulated as constant head. The Groundwater contaminant transport is simulated with MT3DMS (Ver 5.2) (Zheng, 1990).

The model is structured into three phases i.e. input, simulation and output. The model takes Hydro-geological data, Domain-discretization data and the pumping data as input. The pumping data consists of pumping well locations with the respective pumping rate (to be optimized in this study). The formulation used for this study treats only the pumping rates as the decision variables for fixed well locations. After input phase the simulation is done using MODFLOW and MT3D. The study model simulates TNT and TCE (the two chosen parameters). And in the end objective function is calculated using the pumping data (input decision variables) and the simulated concentrations (C_{RDX}^{max} and C_{TNT}^{max}) at the end of simulation period. The model units are in feet and years.

Objective Function

The objective of this formulation is to minimize the total operation costs (i.e. continuous value problem) for the entire project duration i.e.

$$\min_{Q}(VCE(Q) + VCG(Q) + PenaltyCost(Q, C)) \tag{3.3}$$

where,

VCE (Q): Variable electric cost of operation wells.

VCG (Q): Variable costs of GAC units.

Penalty Cost (Q,C): For violating the concentration constraint, pumping constraint.

where, Q=($Q_1, Q_2....Q_{10}$) is the respective well pumping rate ($i = 1, ..., 8$ are pumping wells and $j = 9 - 10$ are recharge basins with the last recharge basin getting a recharge as per constraint 6). $C(Q)$ is maximum contaminant concentration of TNT and RDX respectively (C_{RDX}^{max} and C_{TNT}^{max}) simulated using MODFLOW-MT3D model.

All the cost terms are computed in net present value (NPV) with the following discount function $NPV = \frac{cost_{iy}}{(1+r)^{iy-1}}$. Where, NPV is the net present value of a cost incurred in year iy with a discount rate of r=5%. The cost term is evaluated at the end of each year to account for annual discounting and to ensure that no costs are incurred after cleanup is achieved.

Constraints

The formulation includes the following constraints that must be satisfied while the objective function is minimized

1. The modeling period consists of 1 management period of 4 years, with pumping rates kept throughout this period.
2. Cleanup must be achieved at the end of 4 years. In other words, the max-

imum concentrations of RDX and TNT in model layer 1 must be less than their respective cleanup targets by the end of 4 years $C_{RDX}^{max} \leq 2.1 ppb$ and $C_{TNT}^{max} \leq 2.8 ppb$

3. The total pumping rate, after adjustment for the average amount of system uptime, cannot exceed 1300 gpm. Hence the current maximum capacity of the treatment plant $\frac{1}{\alpha} Q_{total} \leq 1300$, where α is a coefficient representing the average amount of system uptime (α=0.9 for this study)

4. RDX and TNT concentrations must not exceed their respective cleanup levels beyond a specified area when evaluated at the end of each management period.

5. The total amount of pumping must equal the total amount of injection through the infiltration basins within an error tolerance. This is insured by setting the total recharge to the third basin to be equal to the sum of pumping for all extraction wells minus the recharge to the first two recharge basins.

Optimization-Modeling Approach

The optimization formulation tries to do the cleanup by finding the optimal pumping and recharge rates for fixed locations (8 in number and 3 recharge basins). Figure 3.1 shows the location of existing pumping wells and the infiltration basins (Recharge basins). Thus, the optimization goal is to identify a pumping strategy that lowers the C_{max} values of RDX and TNT to their respective cleanup targets of 2.1 and 2.8 ppb in layer 1 within 4 years while satisfying all the pumping constraints. For this study the specific objective is to identify the best pumping rates on eight pumping wells and two recharge basins. The max-

imum allowed concentration constraint and the total pumping constraint are implemented by using the penalty functions hence the solution points not satisfying either of the two constraints (concentration and pumping) are penalized, which forces the algorithm to look for solution points that satisfy the above-mentioned constraints. The flow and transport model takes approximately 5 mins per simulation on a Pentium 2.2 Ghz computer.

3.3.2 Blaine Ammunition Depot

Site History

The Blaine site covers significantly larger area than Umatilla and its model is much more computationally expensive to simulate than Umatilla model. The Blaine Naval Ammunition Depot (NAD) comprises 48,800 acres just east of Hastings,Nebraska. It was build in early 1940s as an active ammunition facility during World War II and the Korean Conflict. While producing nearly half of the Naval ammunition used in World War II, this facility generated that was disposed of on the site through surface impoundments and natural drainage areas of the facility, and disposal of solid waste and explosives. Large tracts of the former Blaine NAD were sold to various individuals, businesses and municipalities in mid-1960s, as a result there are over 100 irrigation wells in the area.

Groundwater contamination at this site is primarily due to chemical spills and discharge of wastewater to surface impoundments, wastewater systems and natural drainages. The contaminants of concern are VOCs and explosives. Two contaminants, Trichloroethylene (TCE), a probable carcinogen, and

Trinitrotoluene (TNT), a possible carcinogen, are used as indicator parameters. Groundwater is encountered approximately 100 feet below ground surface. The semi-confined layer is the major water supply aquifer in the region. The groundwater flow directions are altered in irrigation seasons due to heavy pumping.

Model Description

Groundwater flow is simulated using MODFLOW (Harbaugh et.al., 2000) code. The study model has 136 rows, 82 columns and 6 layers, with variable grid spacing of 400 ft by 400 ft in the center of the model to 2000 ft to 2000 ft near the model edges. Three hydrogeologic units in the saturated zone of interest are

1. The unconfined aquifer (model layer 1)
2. The upper confining layer (model layer 2)
3. The semi-confined aquifer (model layers 3-6)

Transport of the contaminants TNT and TCE is modeled using MT3DMS (Zheng, 1990).

Objective Function

The objective of this formulation is to minimize the total operation costs (i.e. continuous value problem) with respect to fixed well locations for the entire project duration. The Objective function is

$$\min_{Q}(VCE(Q) + VCG(Q) + PenaltyCost(Q, C)) \tag{3.4}$$

where,

1. VCE (Q): Variable electric cost of operation wells
2. VCG (Q): Variable costs of GAC units
3. Penalty Cost (Q,C): For violating the concentration constraint, pumping constraint

where, Q=($Q_1, Q_2....Q_{15}$) is the respective well pumping rate and $C(Q)$ is maximum contaminant concentration of TCE and TNT respectively (C_{TCE}^{max} and C_{TNT}^{max}) simulated by MODFLOW-MT3D model.

Constraints

The formulation includes following constraints that must be satisfied while the objective function is minimized. The biggest difference between the constraints for Blaine problem and Umatilla problem is that there is no constraint on total pumping (Umatilla 3^{rd} constraint) and also the total pumping need not to be equal to total recharge (Umatilla 5^{th} Constraint).

1. The modeling period consists of 6 management period of 5 years, with pumping rates kept throughout this period.
2. Cleanup must be achieved at the end of 30years. In other words, the maximum concentrations of TCE and TNT in model layer 1 must be less than their respective cleanup targets by the end of 30 years $C_{TCE}^{max} \leq 5.0 ppb$ and $C_{TNT}^{max} \leq 2.8 ppb$
3. The pumping capacity of individual wells must not exceed 350 gpm in the less permeable portion of the aquifer.

4. TCE and TNT concentrations must not exceed their respective cleanup levels beyond a specified area when evaluated at the end of each management period.

Optimization-Modeling Approach

The optimization goal for the Blaine NAD site is to identify a pumping strategy that lowers the C_{max} values of TCE and TNT to their respective cleanup targets of 5.0 and 2.8 ppb within 4 years while satisfying the pumping constraints for individual well. For this study the specific objective is to identify the best pumping rates on 15 pumping wells over six management periods. The maximum allowed concentration constraint is implemented by using the penalty functions hence the solution points not satisfying any of the two constraints (concentration and pumping) are penalized. The flow and transport model takes approximately 45 mins per simulation on a Pentium 2.2 Ghz PC.

3.4 Results

3.4.1 Umatilla Results

Convergence plot comparison

The performance of algorithms tested on Umatilla function is shown in figure 3.2. The average best value of objective function (y -axis (log scale)) is plotted against the specific i^{th} function evaluation (x- axis). Average best value indicates

the average objective value for the best solution obtained until that i^{th} function evaluation over 10 trials.

For this formulation the Radial Basis Function methods seems to outperform all other algorithms. Specifically the stochastic radial basis function method (MLMSRBF) outperforms all other optimization methods. The ESGRBF and CGRBF also performed well followed by a heuristic algorithm (DDS). Other traditional heuristic algorithms Simulated Annealing (SA)/Genetic Algorithms (GA), Shuffled Complex Evolution (SCE) and derivative based FMINCON methods are inferior in performance to the RBF methods as evidenced by on average higher objective function values with increase in function evaluations. The quality of results obtained by different algorithms is compared in section 3.5. (Description and references of all the algorithms are given in Section 3.2).

Empirical CDF Plots

Empirical Cumulative Density Functions (CDF) plots are another measure used to compare the reliability of algorithm performance. These plots indicate if any one of the algorithms stochastically dominates some other algorithm. Empirical CDF plots are constructed from probabilistic weights assigned to ordered statistics. Figure 3.3 compares the empirical CDF plots of all the algorithms for Umatilla groundwater contamination transport model. Weibull's plotting positions were used to assign weights to ordered best-objective values from the 10 trials of each algorithm. The weights were assigned using $i/(n+1)$, where i is the rank of the ordered best objective value and n is the total number of data points. This study focuses on minimization problem, hence the empirical CDF

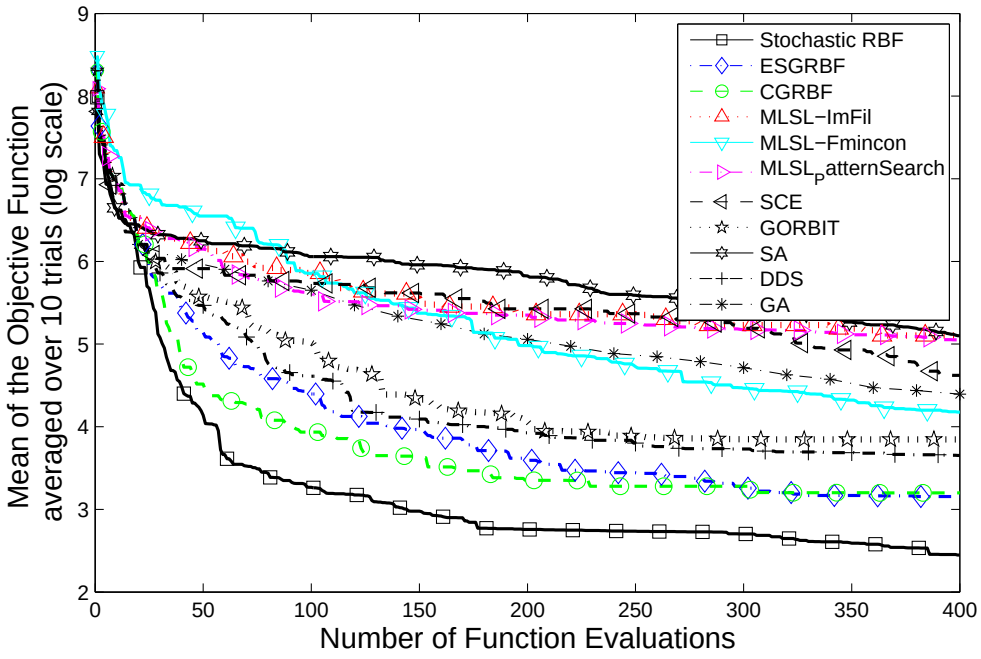


Figure 3.2: Global Optimization Methods on 10 Dimensional Umatilla Function. References of all methods given in section 3.2

that has a high probability for a lower objective value should be preferred. So the CDF plot for a good algorithm should be to the far left. If algorithm A's CDF is always to the left of algorithm B's CDF, the algorithm A is said to dominate algorithm B. Figure 3.3 shows that MLMSRBF (Stochastic RBF) stochastically dominates all other tested algorithms. The two RBF methods i.e. ESGRBF and CGRBF also have excellent CDF. It can be noted that Multi-start Fmincon found two very good solution points but also got stuck in local minima at other times thus suggesting that a high reliability cannot be expected. Of all the heuristic algorithms tested DDS performed the best and SA was the worst. GA results are well spread out thus suggesting that they are less reliable and less effective than other algorithms. SCE and DDS perform worse than RBF methods but better than methods SA, Pattern Search and Implicit Filtering. GORBIT another

RBF method based on trust regions did not perform good results on the tested Groundwater function.

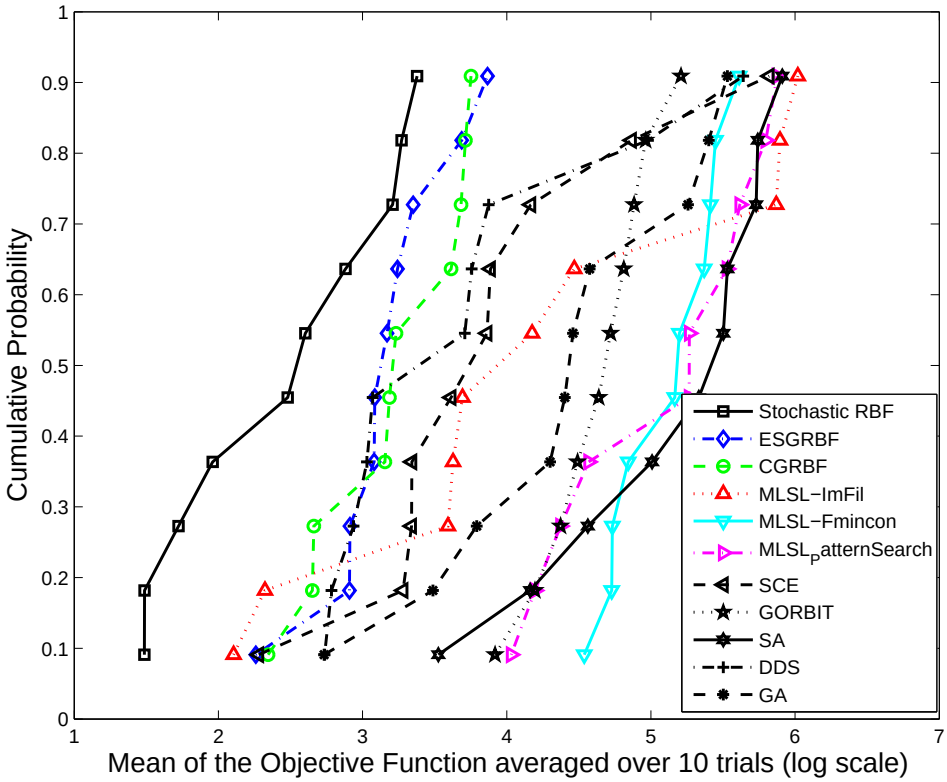


Figure 3.3: Comparison of Empirical Cumulative Density Functions for tested Algorithms on Umatilla Function after 400 simulations. References of all methods given in section 3.2

Box plots

In practice under time constraints with long simulation times it is very unlikely that multiple optimization trials will be performed for optimization. Therefore an algorithm which produces good solutions consistently becomes a superior choice over another algorithm which is as likely to produce an excellent solution

as a poor solution. Figure 3.2 only shows the mean values, they do not show the variability in the solutions produced by each algorithm. The variability in the solutions of the algorithm are compared using box plots as in Figure 3.4. The box plots show the median, inter quartile range (as signified by the ends of the box), whiskers (for data that extend beyond the quartiles) and outliers (>1.5 times the inter-quartile range beyond each quartile) based on a specified number of trials (10 in this case). ESGRBF method produces the results with least variability. MLMSRBF finds two very good solution points thus making it slightly more variable that the other two RBF methods but with much lower mean. Multi-Start Fmincon has some excellent solutions but also has very poor solutions thus resulting in high variability. All other methods have either more variability and/or have far worse means.

Statistical Testing

In order to ensure a fair comparison between the tested algorithms this study tries to initiate all the algorithms from same set points (set to take into account the population and non-population based methods). Pairwise two sample statistical tests were performed for significant difference in the means of the objective function values for the best objective function value by each algorithm. Table 3.1 shows the p-values for each pairwise test. A p-value is the smallest value of the type-I error (i.e. incorrectly rejecting the null hypothesis when it is true) such that the observed results would be sufficient to reject the null hypothesis (which in this case is that the algorithms are same). A low p-value suggests a strong evidence for rejection of null hypothesis (i.e. that one algorithm is better than another). Two types of statistical tests were used to decide about the

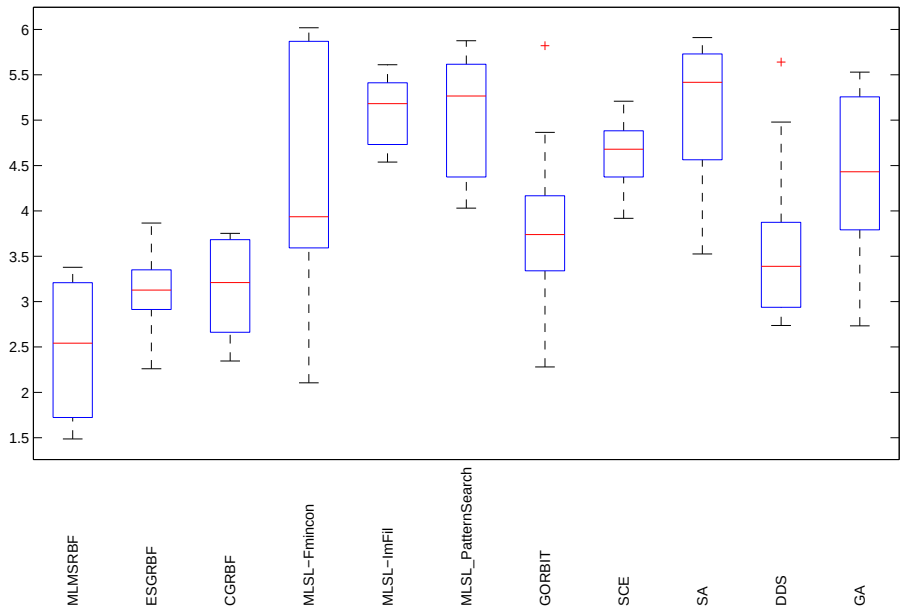


Figure 3.4: Box plot of Best Solution found for Each Algorithm based on 10 trials on Umatilla Function after 400 simulations. References of all methods given in section 3.2

algorithm differences i.e two-sample $t - tests$ and two-sample Wilcoxon rank sum test. The $t - test$ assumes underlying distribution to be normal, which is justifiable when a large number of samples are used in calculating the means. If the number of samples available are less than 30 (approximately), the normality assumption for the means may not hold to justify a $t - test$. In such cases a non-parametric procedure (i.e. Wilcoxon rank sum test) which does not assume normality of means is more reasonable. Wilcoxon rank sum test still requires that the two populations have the same shape and spread. Statistical $t - test$ and Wilcoxon rank sum test results for Umatilla function indicate that MLMSRBF algorithm produces significantly different (lower) mean at a 5% significance level from all the other tested algorithms, thereby providing strong evidence of supe-

rior algorithm performance. Test results for comparison of ESGRBF and CGRBF fails to indicate any significant difference at 5% significance level between the results produced by these two algorithms. Test results for comparison of DDS with ESGRBF or with CGRBF also fail to decide which of these algorithms is better at 5% level, but RBF based methods (MLMSRBF, ESGRBF or CGRBF) clearly are superior at 10% significance level.

The comparisons in the previous section indicate clearly that the Response Surface based methods perform well with a limited computational budget.These methods are relatively better suited to handle global optimization problems with multiple local minima. The failure of some of these methods can be accounted to the fact that in this study as the objective function is expensive, the number of simulations that can be performed is relatively small. Based on these results only some algorithms were chosen to be run on computationally more expensive Blaine function.

3.4.2 Blaine results

The algorithms chosen to be applied on Blaine included the best algorithms MLMSRBF and the other RBF algorithm ESGRBF, that did well. We also included the heuristic's DDS and GA and the most common type of multistart method, which is Mulistart-Fmincon. The performance of selected algorithms chosen to be run on blaine function (i.e. MLMSRBF, ESGRBF, DDS, Multistart-Fmincon and GA) is shown is figure 3.5. The figure 3.5 shows the convergence plot for the respective algorithms (averaged objective function value v/s respective function evaluations). The response surface method MLMSRBF out-

Table 3.1: Statistical comparison of Global optimization methods on Umatilla test function for results after 400 simulations: p-values for 2 Sample t-test

MLMSRBF	ESGRBF	CGRBF	DDS	GORBIT	Fmincon	GA	SCE	PS	ImFil	SA
MLMSRBF	0.015	0.016	0.006	0.001	0.003	<0.001	<0.001	<0.001	<0.001	<0.001
ESGRBF		0.84	0.16	0.05	0.04	<0.001	<0.001	<0.001	<0.001	<0.001
CGRBF			0.21	0.07	0.05	0.001	<0.001	<0.001	<0.001	<0.001
DDS				0.66	0.35	0.09	0.009	0.001	<0.001	<0.001
GORBIT					0.5	0.2	0.03	0.005	0.001	0.005
Fmincon						0.68	0.35	0.09	0.06	0.09
GA							0.47	0.08	0.03	0.07
SCE								0.1	0.01	0.09
PS									0.83	0.88
ImFil										0.99

where, PS-Pattern Search and ImFil-Implicit Filtering

Low p-value indicates the strongest evidence that one algorithm is better than other whereas high p-value indicates that the results suggest that algorithms are essentially producing similar results

Table 3.2: Statistical comparison of Global optimization methods on Umatilla test function for results after 400 simulations:p-Values for Wilcoxon rank-sum test

MLMSRBF	ESGRBF	CGRBF	DDS	GORBIT	Fmincon	GA	SCE	PS	ImFil	SA
MLMSRBF	0.051	0.03	0.01	0.002	0.004	<0.001	<0.001	<0.001	<0.001	<0.001
ESGRBF		0.79	0.52	0.03	0.05	0.004	<0.001	<0.001	<0.001	<0.001
CGRBF			0.34	0.07	0.12	0.003	<0.001	<0.001	<0.001	<0.001
DDS				0.43	0.43	0.12	0.02	0.005	0.007	0.006
GORBIT					0.43	0.16	0.02	0.007	0.007	0.014
Fmincon						0.73	0.24	0.21	0.14	0.27
GA							0.57	0.12	0.04	0.08
SCE								0.14	0.02	0.07
PS									0.9	0.85
ImFil										0.52

where, PS-Pattern Search and ImFil-Implicit Filtering

Low p-value indicates the strongest evidence that one algorithm is better than other whereas high p-value indicates that the results suggest that algorithms are essentially producing similar results

performs all other methods and GA performs the worst. For Blaine function the other heuristic algorithm DDS outperforms the ESGRBF.

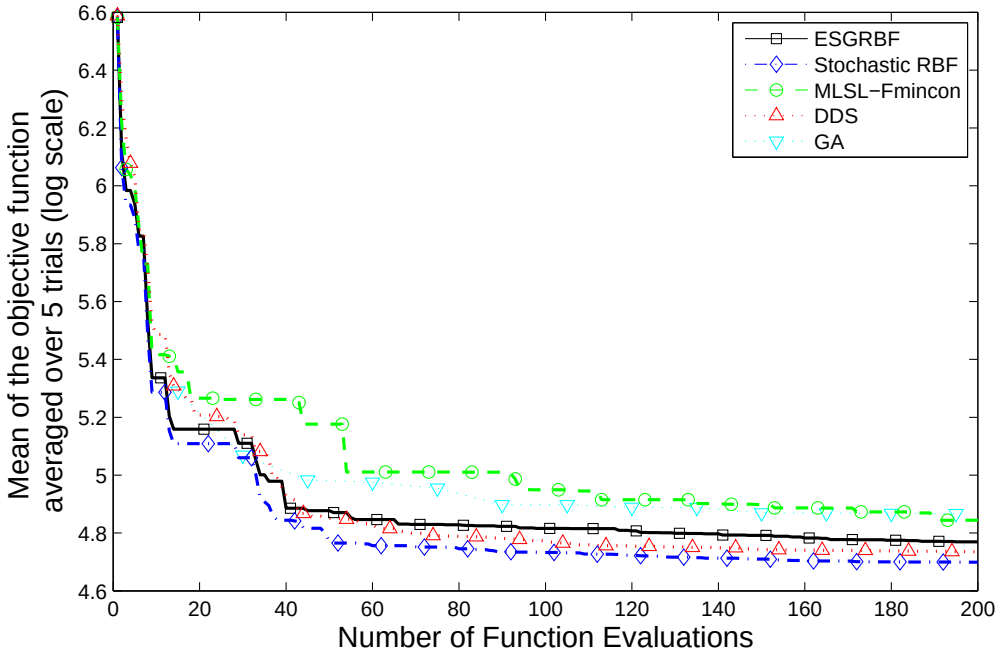


Figure 3.5: Global Optimization Methods on 15 Dimensional Blaine Function. References of all methods given in section 3.2

3.5 Discussion

In this section we attempt to investigate the failure of some of the tested algorithms. The failure of different algorithms is mainly due to nonlinearity (hence multimodal characteristic) of objective function. The non linearity in Umatilla objective function arises mainly due to two penalty functions i.e. two constraints. The first forces the formulation to pump (across all wells) less than certain fixed amount and the second constraint forces the concentration at the

end of the simulation period to be less than the standard for the two contaminants respectively. Whereas for Blaine function the nonlinearity is mainly due to concentration constraint. Failure of some of the tested algorithms is mainly due to their failure in satisfying these nonlinear constraints. Table 3.3 lists the constraints behavior (selected algorithms) for Umatilla function at the end of simulation period for the optimization trial that gives the median objective function for the algorithm. Table 3.4 lists the respective pumping rates. The table also lists the respective objective function values with constraint violations and corresponding penalties for the respective optimal solution points. For the median trial MLMSRBF is the only algorithm that satisfies all the constraints, hence resulting in minimum objective function value. Median solution for ESGRBF fails to satisfy RDX constraint whereas FMINCON fails to satisfy both RDX and TNT constraint. Genetic Algorithms (GA) perform the worst and it fails to satisfy all three constraints. The purpose of tables 3.3 and 3.4 is show the difference in quality of results obtained by different algorithms in terms of constraints and objective function values. Similarly table 3.5 list the pumping rates for respective wells for Blaine function. The table also list the objective function values obtained by respective algorithms after 200 function simulations. The table again shows that MLMSRBF performs the best (lowest objective function value) whereas GA's perform the worst (highest objective function value).

The MLMSRBF, which has stochastic component in addition to RBF to reduce the number of simulations performs the best of all the tested algorithms. ESGRBF, which is an evolutionary algorithm coupled with a function approximation to reduce the number of simulations also gave good results for Umatilla function. Controlled Gutmann (CGRBF) which tries to evaluate the costly function only at selected points instead of all corners also gave very good results. For

Table 3.3: Algorithm Comparison in terms of Constraints satisfied for Umatilla test function for an optimization trial (median for objective function values among 10 trials) after 400 simulations. References of all methods given in section 3.2

Algorithm	**RDX constraint**[a]	**TNT constraint**[b]	**Pumping constraint**[c]
MLMSRBF	Satisfied	Satisfied	Satisfied
ESGRBF	Not satisfied	Satisfied	Satisfied
Fmincon	Not satisfied	Not satisfied	Satisfied
GA	Not satisfied	Not satisfied	Not satisfied

[a] refers to $C_{RDX}^{max} \leq 2.1 ppb$ at the end of simulation period
[b] refers to $C_{TNT}^{max} \leq 2.8 ppb$ at the end of simulation period
[c] refers to $\alpha Q_{total} \leq 1300$

these types of constrained optimization problems i.e. with multiple local minima and a rough surface (due to numerical approximations/limitations) that are computationally expensive to evaluate the function approximations appear to be quite effective. The MLMSRBF, ESGRBF and CGRBF (not done for Blaine) drop more quickly in objective function value on both the Umatilla and Blaine problems than all the other methods i.e. they are able to avoid long periods of function evaluations without any improvement. One other attractive feature about RBF methods was no parameter tuning.

The two derivative-based methods, the SQP-Fmincon and the Implicit Filtering Method coupled with MLSL did not perform well on the two problems. The failure of these methods can be accounted to the limited computational budget, dimensionality and the rough surface of the objective function (due to numerical approximations/limitations). The derivative based methods converged to local minima not the global minima. The convergence plot shows that for both

Table 3.4: Algorithm Comparison in terms of pumping rates for Umatilla test function for an optimization trial (median for objective function values among 10 trials) after 400 simulations. References of all methods given in section 3.2

Well index	**Pumping Rates (GPM)**			
	MLMSRBF	**ESGRBF**	**Fmincon**	**GA**
Pumping Well 1	40	0	0	256
Pumping Well 2	111	257	193	47
Pumping Well 3	320	385	356	11
Pumping Well 4	10	28	0	24
Pumping Well 5	244	179	399	351
Pumping Well 6	398	294	46	338
Pumping Well 7	0	5	169	56
Pumping Well 8	47	0	0	96
Recharge Basin 1	0	0	62	59
Recharge Basin 2	561	543	0	781
Recharge Basin 3	609	605	1101	341
Objective Function	891	1585	2520	2.1×10^5
Penalty Function	0	680	1500	2×10^5
*Max(0,$C_{RDX}^{max} - 2.1ppb$)	0	0.3	1.1	2.1
*Max(0,$C_{TNT}^{max} - 2.8ppb$)	0	0	0.3	2.3

**Represents the two concentration constraints $C_{RDX}^{max} \leq 2.1ppb$ and $C_{TNT}^{max} \leq 2.8ppb$ at the end of simulation*

Table 3.5: Algorithm Comparison in terms of pumping rates for Blaine test function for an optimization trial (median for objective function values among 10 trials) after 200 simulations. References of all methods given in section 3.2

Well index	**Pumping Rates (GPM)**			
	MLMSRBF	**ESGRBF**	**Fmincon**	**GA**
Pumping Well 1	1	161	115	181
Pumping Well 2	141	99	281	156
Pumping Well 3	83	283	97	113
Pumping Well 4	143	234	176	194
Pumping Well 5	100	190	122	115
Pumping Well 6	228	218	314	255
Pumping Well 7	159	91	222	46
Pumping Well 8	24	149	223	114
Pumping Well 9	228	342	325	301
Pumping Well 10	249	229	137	379
Pumping Well 11	209	286	670	261
Pumping Well 12	162	167	335	106
Pumping Well 13	147	103	98	49
Pumping Well 14	240	243	269	302
Pumping Well 15	27	0	96	107
Total Pumping	2150	2803	2884	2686
Objective Function	5×10^4	5.8×10^4	7.2×10^4	6.8×10^4

these algorithms the best objective function value is improving but at a much slower rate than the RBF based methods i.e. if given enough function evaluations (increased computational budget) algorithms will be able to find optimal solution points. It is surprising to note that multi start Fmincon outperforms multi start Implicit filtering. The results show that Fmincon was able to find two good solution points but also some worst solution points, thus making this an unreliable method.

Pattern Search (Torczon, 1997) is similar to the classical simplex reflection method by Nelder & Mead (1965), which is incorporated in the SCE algorithm. Pattern Search methods is local optimizer. MLSL-Pattern Search uses multi-level single linkage to convert the local optimizer into a global optimizer,whereas SCE has its own unique method for developing a global optimizer from the simplex reflection procedure. Both of these methods were developed for models, that run very quickly so that many thousands of simulations were feasible. The aim of this study to suggest algorithms that can be used to find good solution (optimal) points for an optimization problem (computationally expensive simulation) under limited computational budget. These methods again can find good solution points if given enough computational budget.

Of all the heuristic algorithms tested, Simulated Annealing and Genetic Algorithms did not perform well, whereas a new heuristic algorithm DDS performed well. In case of GA and SA, as the objective function is expensive to compute, the number of simulations that were performed for identification of algorithm parameters such as population size, crossover and mutation probabilities were limited. It was observed that the GA population nearly converged to the elite member (the best solution) after first few generations, which then

placed a heavy emphasis on mutation to produce any improvement. Since a larger population size results in fewer iterations, it is not likely that increasing the population size would produce an improvement when the number of function evaluations is fixed. It is worth discussing the success of DDS. DDS is new global optimization algorithm that is very simple. DDS was the most successful method that did not use a response surface as a surrogate during the optimization search. Other attractive features of DDS is that it has only one parameter to tune and is very easy to implement.

3.6 Conclusion

The results of optimal policy design for pump and treat system (i.e. continuous value pumping rate variables) illustrate the efficacy of Response surface based methods. The study compared some Response surface based methods with heuristic and derivative based methods. The study coupled some local optimizers with Multi-Level Single Linkage (MLSL) multistart procedure to use a local optimization algorithm for solving global optimization problems. The results indicated that under limited computational budget (i.e. limit on the number of computationally expensive simulations/function evaluations), the Response surface based methods (MLMSRBF, ESGRBF and CGRBF) were the most effective algorithms with DDS being second best to these RBF methods. These results also show that Response surface based methods can replace traditionally used methods for optimal policy design over multiple management periods and also can be coupled with some integer programming solver (such as Tabu Search) to solve mixed integer programming models.

These results are based on groundwater pump and treat system problems and do not prove that RBF methods will always be better than other algorithms such as SCE, Implicit filtering on water resource problems. However the results still suggest that some of these Response Surface based methods and DDS, a simple new heuristic algorithm, should be considered as alternatives to widely used methods such as SCE and evolutionary algorithms.

BIBLIOGRAPHY

[1] Aarts E. and Korst.J., *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing*. John Wiley & Sons, 1989.

[2] Ahlfeld, D.P., Mulvey, J.M. and Pinder, G.F., 1986, *Designing optimal strategies for contaminated groundwater remediation*, Advanced Water Resources, 9, 7784.

[3] Aly, A. H., and Peralta, R. C. (1999b). *Optimal design of aquifer cleanup systems under uncertainty using a neural network and a genetic algorithm.* Water Resour. Res., 35(8), 25232532.

[4] Aral, M. M., and Guan (1996), *Optimal groundwater remediation design using differential genetic algorithm*, J. Computational Methods in Water Resources XI (1), 349-357, Computational Mechanics Publications.

[5] Atwood, D., & Gorelick, S. (1985) *Hydraulic Gradient Control for Groundwater Contaminant Removal*. J. of Hydraul. 76, 85-106.

[6] Becker, D., Minsker, B., Greenwald, R., Zhang, Y., Harre, K., Yager, K., Zheng, C. and Peralta, R. (2006),*Reducing Long-Term Remedial Costs by Transport Modeling Optimization. Ground Water*, 44: 864875. doi: 10.1111/j.1745-6584.2006.00242.x.

[7] Buhmann, M. D. (2003) *Radial Basis Functions. Cambridge University Press* , Cambridge, UK

[8] Chang, L.-C., Shoemaker, C. A., and Liu(1992), P. L.-F., *Optimal time-varying pumping rates for groundwater remediation: Application of a constrained optimal control algorithm*. Water Resources Research, 28(12), 3157-3173.

[9] Dibike, Y.B., Solomatine, D., and Abbott, M.B. *On the encapsulation of numerical hydraulic models in artificial neural networks*. J. Hydraulic Research, 37, 147-161,1999.

[10] Duan, Q. , Gupta, V. K. and Sorooshian, S. (1993) *Shuffled complex evolution approach for effective and efficient global minimization*. J. Optimization Theory and Applications 76:3 , pp. 501-521.

[11] Devore J. L., *Probability and Statistics for Engineering and the Sciences*, Kluwer Academic Publishers, Boston, MA, 1997.

[12] El Harrouni K., Ouazar, D., Walters, G. A., and Cheng, A. H.-D. *Groundwater optimization and parameter estimation by genetic algorithm and dual reciprocity boundary element method*. Engineering Analysis with Boundary Elements, 18(4), 287-296, 1996.

[13] Espinoza, F. P., Minsker, B. S., and Goldberg, D. E. (2005). *Adaptive hybrid genetic algorithm for groundwater remediation design.* J. Water Resour. Plann. Manage., 131(1), 1424. [ISI]

[14] Eusuff Muzaffar M. and Lansey Kevin E., *Optimal Operation of Artificial Groundwater Recharge Systems Considering Water Quality Transformations* Water Resources Management Volume 18, Number 4, 379-405, 2004.

[15] Harbaugh, A.W., Banta, E.R., Hill, M.C., and McDonald, M.G., 2000, MODFLOW-2000, *The U.S. Geological Survey modular ground-water model – User guide to modularization concepts and the Ground-Water Flow Process*: U.S. Geological Survey Open-File Report 00-92, 121 p.

[16] Gilmore, P. & Kelley, C. T. (1995) *An implicit filtering algorithm for optimization of functions with many local minima*.SIAM J. Optimization 5, 269285.

[17] Goldberg, D.E., 1989, *Genetic algorithms in search, optimization, and machine learning*, Addison-Wesley publishing company inc., Massachusetts, 432p.

[18] Gorelick, S.M., Voss, C.I., Gill, P.E., Murry, W., Saunders, M.A. and Wright, M.H., 1984, *Aquifer reclamation design: The use of contaminant transport simulation combined with nonlinear programming*, Water Resources Research, 20, 415427.

[19] Gutmann, H.M. 2001b. *A radial basis function method for global optimization.* Journal of Global Optimization 19 201.227.

[20] Kelley, C. T. (1999) *Iterative Methods for Optimization*. SIAM, Philadelphia, Pennsylvania, USA.

[21] Karatzas, G. P., and Pinder, G. F. *Groundwater management using numerical simulation and the outer approximation method for global optimization*. Water Resources Research, 29(10), 3371-3378, 1993.

[22] Karatzas, G. P., and Pinder, G. F. *Combination of groundwater simulation with an outer approximation methods for global optimization*. Computational Methods in Water Resources IX (1), 337-344, Computational Mechanics Publications, 1992.

[23] Maskey, S., Jonoski, A., and Solomatine, D. P. *Groundwater Remediation Strategy Using Global Optimisation Algorithms*. J. Water Resources Planning and Management, ASCE, 1999.

[24] Naval Facilities Engineering Command (NAVFAC) technical report TR-2237-ENV, 2004.

[25] Mugunthan P., Shoemaker C., and Regis R.,*Comparison of function approximation,heuristic and derivative-based methods for automatic calibration of computationally expensive groundwater bioremediation models*, Water Resour. Res.,

41 (2005).

[26] Nelder, John A.; R. Mead. *A simplex method for function minimization* . Computer Journal 7: 308313. doi:10.1093,comjnl,7.4.308, 1965.

[27] Powell, M. J. D. Muller, M. , Buhmann, M. , Mache, D. and Felten, M. (eds) (1999) *Recent research at Cambridge on radial basis functions*. New Developments in Approximation Theory 132 , pp. 215-232. International Series of Numerical Mathematics , Birkhauser Verlag, Basel, Switzerland

[28] Rao, Z.-F., and Jamieson, D. G. *The use of neural networks and genetic algorithms for design of groundwater remediation schemes*. Hydrology and Earth System Sciences,1(2), 345-356, 1997.

[29] Reed P.,Minsker B., and Goldberg D. E., *Designing a competent simple genetic algorithm for search and optimization*, Water Resour. Res., vol. 36, no. 12, pp. 3757-3761, 2000.

[30] Regis, R. G. & Shoemaker, C. A. (2004) *Local function approximation in evolutionary algorithms for the optimization of costly functions*. IEEE Trans. Evolutionary Computation 8(5), 490505

[31] Regis, R. G. and Shoemaker C. A., *A Stochastic Radial Basis Function Method for the Global Optimization of Expensive Functions*. INFORMS Journal on Computing, Vol. 19, No. 4, pp. 497-509, Fall 2007.

[32] Regis, R. G. and Shoemaker C. A., *Improved Strategies for Radial Basis Function Methods for Global Optimization*. Journal of Global Optimization, Vol. 37, No. 1, pp. 113-135, 2007.

[33] Rinnooy Kan, A. H. G. & Timmer, G. T. (1987) *Stochastic global optimization methods*. Part II: Multi level methods. Mathematical Programming 39, 5778.

[34] Sait Sadiq M and Youssef,Habib(1999) *Iterative Computer Algorithms with Applications in Engineering IEEE Computer Society*, Los Alamitos, California.

[35] Solomatine, D.P. *Genetic and other global optimization algorithms comparison and use in calibration problems*. Proc., 3rd Int. Conf. on Hydroinformatics, 1021-1027,1998.

[36] S. Wild, R. G. Regis, and C. A. Shoemaker. *ORBIT: Optimization by Radial Basis Function Interpolation in Trust-Regions*. SIAM Journal on Scientific Computing, Vol. 30, No. 6, pp. 3197-3219, 2008.

[37] S. Wild, and C. A. Shoemaker (2009) *GORBIT: Global Optimization by Multistart Radial Basis Function Algorithms* submitted

[38] Tolson, B. A., and C. A. Shoemaker (2007), *Dynamically dimensioned search algorithm for computationally efficient watershed model calibration*, Water Resour. Res., 43, W01413, doi:10.1029/2005WR004723. Jan. 2007

[39] Tolson, B. A. (2005) *Automatic Calibration, Management and Uncertainty Analysis: Phosphorous transport in the Cannonsville Watershed*. PhD Thesis, School of Civil and Environmental, Cornell University, Ithaca, New York,USA.

[40] The Mathworks, Inc. (2009a) *Genetic Algorithm and Direct Search Toolbox for Use with MATLAB: Users Guide*, version 1. The Mathworks, Inc., Natick, Massachusetts, USA.

[41] The Mathworks, Inc. (2009a) *Optimization Toolbox for Use with MATLAB: User's Guide*, version 3. The Mathworks, Inc. Natick, Massachusetts, USA

[42] Torczon, V. (1997) *On the convergence of pattern search algorithms*. SIAM J. Optimization 7, 125.

[43] Yoon, J.-H., C. A. Shoemaker. *Comparison of optimization methods for groundwater bioremediation*. J. Water Resources Planning Management 125(1) 5463, 1999.

[44] Yoon J.H and Shoemaker C.A. , *Improved real-coded GA for groundwater bioremediation*, J. Water Resour. Plan Management, vol. 125, no. 1, pp.54-63,1999.

[45] Wang, M., and Zheng, C. *Optimal remediation policy selection under general conditions*. Ground Water, 35(5), 757-764, 1997.

[46] Willis M. B. , Modeling,*Optimization and Sensitivity Snalysis of Reductive Dechlroination of Chlorinated Ethenes with Microbial Competition in Groundwater*, Ph.D. dissertation, Cornell University, Ithaca, NY, 2001

[47] Zheng, C., 1990, *MT3D: A modular three-dimensional transport model for simulation of advection, dispersion, and chemical reactions in groundwater systems*, Report to the U.S. Environmental Protection Agency, Ada, OK, 170p.

[48] Zheng, C. and Wang, P.P., 1998, *MT3DMS, A modular three-dimensional multispecies transport model for simulation of advection, dispersion and chemical reactions of contaminants in groundwater systems*, Vicksburg, Miss., Waterways Experiment Station, U.S. Army Corps of Engineers. 238p.

CHAPTER 4

GLOBAL OPTIMIZATION FOR FIXED COST PROBLEMS WITH APPLICATION TO LARGE GROUNDWATER PROBLEM

4.1 Introduction

This study integrates our method *Search over Integers with Tabu List (SIT)* with a Response surface based global optimization method (Stochastic RBF) to solve fixed cost global optimization problems, i.e. a problem with both discrete and continuous-value decision variables. An important area where this problem arises is for an objective function that minimizes the sum of installation cost for a facility and the operation-maintenance cost for that facility over time. We will consider a case where N facilities can be constructed and as a result of the construction of a particular facility a fixed cost of F_j must be paid. Then there are continuous decision variables $y_j; j = 1, ..., N_I$ which are zero if the facility is not built and otherwise can be any real number $\in [o, y^{max}]$. Associated with the j^{th} facility is integer variable $I_j = 1$ if the facility is built and otherwise $I_j = 0$. Sometimes there might be some pre-existing facilities thus there is no fixed cost associated with these facilities but they still have continuous variables associated with them (i.e. additional continuous value variables). Let N_C be the total continuous value variables thus $N_C \geq N_I$. The optimization problem is then expressed as

$$\min_{Y}(\sum_{j=1}^{N_I} I_j F_j + V(Y)) \tag{4.1}$$

where,

$$y_j \leq I_j y^{max} \quad for\ j = 1, ..., N_I \tag{4.2}$$

and

$$G_i(Y) \geq 0 \quad for\ i = 1, ..., M \tag{4.3}$$

Here Y is a vector of all continuous value variables $y_j; j = 1, ..., N_C$. $G_i(Y)$ is nonlinear constraint that could possibly have a multi-modal surface. Equation 4.2 is used to ensure that if there is no facility at location j then y_j must be zero for all $j = 1, ..., N_I$. y_i^{max} is a constant that is the maximum value of y_j. Equation 4.3 represents a series of other constraints on the 'Y' vector. For example if $\theta(Y)$ is a simulation model and $f_i(Y)$ is an output of the simulation (say contaminant concentration) then $G_i(Y) = C^{max} - f_i(Y) \geq 0$ i.e. value of $f_i(Y)$ is equal to or less than C^{max}.

There are generally two types of decision variables (integer and continuous value variables) that need to be considered in a typical practical water resources management problem (i.e groundwater remediation design problem for this study). During a particular run once the integer variable configuration (well locations) is fixed the installation cost is no longer a variable i.e. the objective function is then to minimize the operation and maintenance cost for that configuration. Such kind of Mixed-Integer Variable Problems (MIVP) problems in water resources management are difficult to optimize especially if $V(Y)$ or $G_i(Y)$ are multi-modal. The inherent relation between the two decision variables makes this a problem with extremely large search domain hence computationally very expensive for real world problems.

There have been a number of papers on for these kinds of problems with continuous value variables including linear approaches [Gorelick et.al. (1979), Willis (1979), Remson and Gorelick (1980), Yoon and Shoemaker (1999)] and nonlinear local optimization methods [Ahlfeld (1997), Ahlfeld and Mulligan

(2000)]. Mixed integer applications for these kind of problems can be broadly categorized according to different optimization approaches: a) Mixed Integer Linear Management modeling approach, b) Mixed Integer Nonlinear local optimization methods c) with Global heuristic methods (e.g. Genetic Algorithms (GA), Simulated Annealing (SA)) and d) Mixed integer with Response Surface. The linear management model fails to take into account the nonlinear behavior of contaminant transport modeling.

Nonlinear methods were used by Gorelick et. al. (1984), Ahfeld (1987) in their work incorporating nonlinear techniques into groundwater management process but these analysis did not include integer variables or fixed cost. Various researchers used MINOS (Murtagh and Saunders (1980), McKinney and lin (1995)) or NPSOL (Gill et al., 1986) to solve these problems. Both MINOS and NPSOL are local optimizers thus cannot be used for problems with integer variables and objective functions with multiple local minima. Hemker et. al. (2006,2008) implement a branch-and-bound approach that uses surrogate functions for a hydraulic capture problem, which is linear. Their study does not include contamination transport which the main reason for multi-modal behavior of remediation problem. Holmstrom et. al. (2008) developed a Response surface method based on Kriging and Radial Basis Function (RBF) interpolation to solve global mixed variable optimization problems. This methodology ARBFMIP (TOMLAB implementation) when applied (in this study) to large scaled problems (equal number of binary/integer and continuous value variables) generated ill conditioned matrices and did not run. The example problem in this paper had 3 integer variables and 5 continuous variables whereas our problem has 10 integer and 10 continuous variables. NOMAD (Abramson et. al., 2008) is a new MIVP optimizer which to the authors knowledge hasn't been applied in

Water resources. Zheng and Wang (1999) developed an integrated approach for remediation design using Tabu Search and Genetic Algorithms. Aly and Peralta (1999) integrated neural networks with Genetic Algorithms. Becker et. al. (2006) used two simulation-optimization packages: SOMOS, developed at Utah State University (USU) (Peralta 2003), and MGO, (Zheng and Wang 2002a). Both of the packages used in this project implement heuristic algorithms: genetic algorithms (Holland 1975; Goldberg 1989), simulated annealing (Metropolis et al. 1953), and tabu search (Glover 1986, 1989). In their study authors acknowledge the fact that "these global methods often require intensive computational effort but have become more practical for application on personal computers as computer speeds have increased". Various researchers have implemented hybrid heuristic methods for these problems (Peralta et. al. (2005)). Jin et. al. (2009) implement parallel hybrid optimization framework that uses genetic algorithms (GA) coupled with local search approaches (GA-LS) to solve groundwater inverse problems. Such approach becomes computationally very expensive for large scale models where one model simulation takes significant time i.e. sometimes one such simulation may run for weeks or months.

The methodology developed here (Mixed integer with Response Surface) suggests a two layered system i.e. Response surface based method for continuous value variables and SIT for integer value variables. For computationally expensive functions, a sensible approach is to use response surface models (also known as surrogate models) for the expensive function. Examples of response surface models Radial Basis Functions (RBF) (Buhmann 2003, Powell 1992), Kriging models (Cressie 1993), and neural networks. Response surface models have been used in the optimization of expensive functions. One of the main attractive features of response surface based methods is that the meth-

ods are ”gradient free” i.e. the methods do not need actual gradient information for optimization run. This feature makes this kind of method very suitable for environmental problems as they generally involve a complex simulation model. None of the response surface methods (including the method used) needs any additional information other than function values for minimization. The methodology implements Radial Basis function based method to reduce the computational time i.e. radial basis function is used as surrogate for simulation. This initial surrogate function is built by evaluating expensive function at some initially generated points (Latin hypercube points for this study). This quality of results from the surrogate function improves as the search progresses i.e. more and more actual (expensive) cost function evaluations are done. The main focus of this study is to sequentially use the actual cost function evaluation information across different integer configurations to improve the accuracy of surrogate function approximations.

This study uses Multistart Local Metric Stochastic RBF (MLMSRBF) called here Stochastic RBF, developed by Regis and Shoemaker (2007). The integration of Stochastic RBF with SIT methodology is first applied on a test problem and hypothetical aquifer, then is tested for a real groundwater aquifer. The methodology we develop for the integration of SIT and the Stochastic RBF is novel and is described in section 4.3.3. The results are then compared to GA, since GA's are widely used for these kind of problems and a mixed variable optimization tool NOMAD (Nonsmooth Optimization by Mesh Adaptive Direct Search), implementation of the Mesh Adaptive Direct Search (MADS) algorithm (http://www.gerad.ca/nomad/Project/Home.html).

This paper is organized as follows. Section 4.2 introduces/explains the ap-

plication problems with details of objective function and constraints. The algorithm and its parameters are explained in section 4.3.3. Results are then discussed in section 4.6, last section 4.7 highlights the conclusions.

4.2 Model Description

The newly developed SIT-RBF methodology is tested on two test problems (one test problem and one hypothetical groundwater problem) and one real groundwater remediation demonstration site. The demonstration site, Umatilla Chemical Depot,is adapted from NAVFAC (*Naval Facilities Engineering Command*) technical report TR-2237-ENV.

4.2.1 Test Problems

The SIT-RBF algorithm is first tested on a test problem and hypothetical groundwater test problem. The objective function for both the test problems is to minimize the sum of fixed (O&M cost) and the variable cost subjected to some constraints. The fixed cost depends on the integer variable configuration, whereas the variable cost depends on the continuous variables. The integer variables are essentially binary variables (0 or 1) and the continuous variables are bounded with upper and lower limit.

The test problem is modified form of 17-dimensional continuous value Schoen problem (1993). The Schoen function is modified into a fixed value problem such that the total cost is minimum when the five of the decision variables are zero i.e. no installation cost for these variables. Hence the objective is to find

this optimal configuration in terms of integer variables (17 in number) and the respective continuous variable values (17 in number).

$$\min_{Y}\left(\sum_{j=1}^{N_I} I_j F_j + VC(Y)\right) \tag{4.4}$$

Subjected to:-

$y_{min} \leq y_j \leq y_{max}$ for $j = 1, ..., N_C$ and

$y_j \leq I_j * y_{max}$*(Fixed cost constraint)* for $j = 1, ..., N_I$

where, N_I is the number of integer variables and $Y = [y_1, ..., y_{N_C}]$ is a set of N_C continuous variables, I_j is the binary variable (0 or 1) associated with j^{th} continuous variable (y_j), F_j is fixed cost associated with j^{th} variable (I_j) and $VC(Y)$ is the variable cost which depends on vector Y.

The objective function for 32-dimensional hypothetical groundwater test problem is to minimize the O&M cost such that concentration constraint is satisfied at the end of simulation period. Each of the 32 continuous variables has a fixed cost attached with it. The total cost is minimum when some of these continuous variables are zero. Hence the objective is to minimize the total cost in terms of integer (32 in number) and binary variables (32 in number) subjected to concentration constraint being satisfied at the end of simulation period. Formulation for this problem has one additional constraint from the test problem i.e. concentration constraint must be satisfied at the end of simulation period, hence the objective function is modified to incorporate penalty when concentration constraint is violated.

$$\min_{Y}\left(\sum_{j=1}^{N_I} I_j F_j + VC(Y) + penaltyfunction(C)\right) \tag{4.5}$$

Subjected to:-

$y_{min} \leq y_j \leq y_{max}$ for $j = 1, ..., N_C$;

$y_j \leq I_j * y_{max}$*(Fixed cost constraint)* for $j = 1, ..., N_I$ and

$C \leq C_{max}$*(Incorporated using penalty function)*

where, N_I is the number of integer variables and $Y = [y_1, ..., y_{N_C}]$ is a set of N_C continuous variables, C is the maximum concentration at end of simulation period measured at observation wells, I_j is the binary variable (0 or 1) associated with j^{th} continuous variable (y_j), F_j is fixed cost associated with j^{th} variable (y_j) and $VC(Y)$ is the variable cost which depends on Y.

4.2.2 EPA Groundwater Site:Umatilla Chemical Depot

Site History

Umatilla Chemical Depot, located in northeastern Oregon is a 19,728 acre military reservation established in 1941 as an ordinance depot for storage and handling of munitions. From the 1950s until 1960s the depot was used as an on-site explosives washout plant. The plant processed munitions to remove and recover explosives using a pressurized hot water system. The wash water from the plant was disposed in two unlined lagoons, from which the wash water infiltrated into the soil system. During this time, an estimated 85 million gallons of wash water was discharged to the lagoons.

Two contaminants, RDX (Hexahydro-1,3,5-trinitro-1,3,4-triazine, and commonly referred to as Royal Demolition Explosive) and TNT (2,4,6-Trinitrotoluene) were used as indicator parameters. U.S. Army Corps of En-

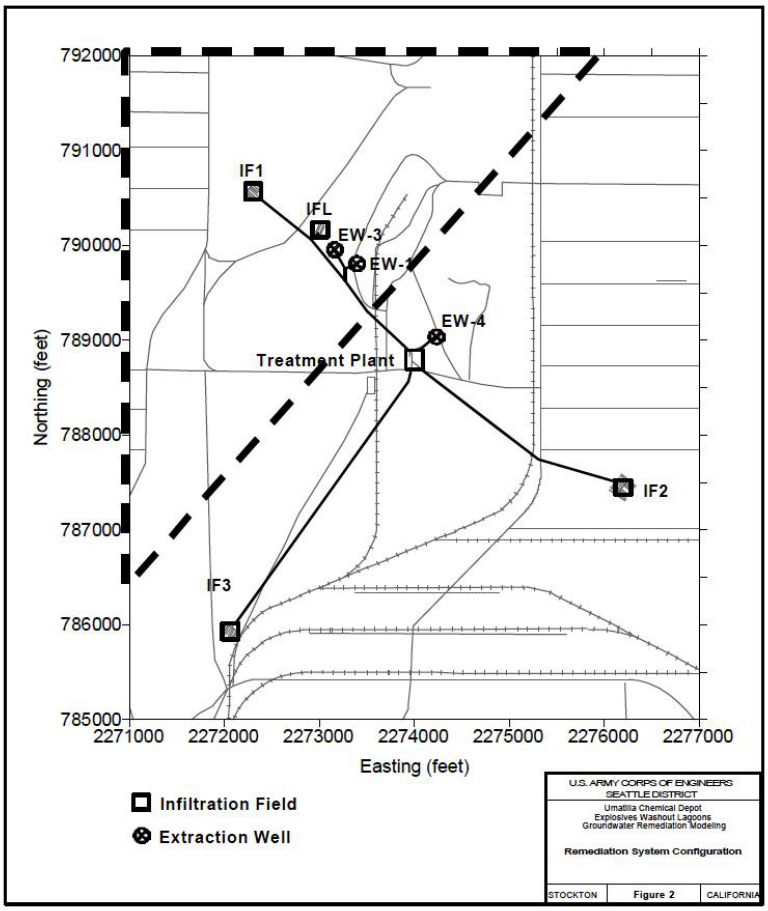


Figure 4.1: Site Map : NAVFAC (*Naval Facilities Engineering Command*) technical report TR-2237-ENV

gineers designed a pump-and-treat system (USACE, 1996 and 2000) to contain and remove the RDX and TNT plumes (Figure 4.1). USACE designed pump-and-treat system consisted of three pumping wells and 2 recharge basins (shown in figure). One of the pumping wells and the infiltration basins are marked as inactive for this study. The cost of activating the inactive well is considerably less than the cost of installing a new well whereas there is no installation/activation cost associated with any other existing wells. Existing wells/basins and the inactive wells/basins play role in the cost definitions. This study uses binary variables associated with all of the pumping well locations i.e.

'1' if active and '0' if inactive. Existing USACE design sets up the initial conditions (initial hydraulic heads and contaminant concentrations) for the model used in this study. The contaminated groundwater is extracted from the wells and then sent to GAC units, which remove the contaminants. The treated water is then discharged to the infiltration basins.

Model Description

Groundwater flow is simulated using the MODFLOW code. MODFLOW is a three-dimensional finite-difference groundwater model that was first published in 1984 (Harbaugh et.al., 2000) and has been updated several times. The study model has 125 rows, 132 columns and 5 layers, with variable grid spacing of 24.8ft - 647.9ft along the rows and 21.6ft - 660.7ft along the columns. The formulation only focuses on contaminant transport in layer 1 of the model. The model boundary conditions for all four sides of the model domain were simulated as constant head. The Groundwater contaminant transport is simulated with MT3DMS (version 5.2) which is the latest (1998) version of MT3D. MT3D was developed by Chunmiao Zheng (1990).

The study model is structured into three phases i.e. input, simulation and output. The model takes Hydro-geological data, Domain-discretization data and the pumping data as input. The pumping data consists of pumping well locations with the respective pumping rate. The formulation used for this study treats only the pumping rates as the decision variables for fixed well locations. After input phase the simulation is done using MODFLOW and MT3D. The study model simulates TNT and TCE (the two chosen parameters). And in the end objective function is calculated using the pumping data (input) and the

simulated concentrations at the end of simulation period. The model units are in feet and years.

Optimization Formulation

Decision Variables

The overall objective for this function to choose optimal well parameters i.e. well locations and the respective extraction/recharge rate of the wells. The integer problem formulation is designed to choose optimal set of pumping wells from all possible well locations and the continuous value formulation finds the respective optimal pumping rates. Each pumping well has a binary variable (location) associated with it, i.e '1' if active or '0' if inactive and the respective pumping rates (continuous variable). For this formulation there were 8 binary variables and 10 continuous variables i.e. 8 pumping wells to choose from and 2 infiltration basins with fixed locations.

Objective Function

The objective of this formulation is to minimize the total costs (installation and operation i.e. mixed value problem) for the project duration (Formulation 1 from NAVFAC report). i.e.

$$\min_{Y,I}\left(\sum_{j=1}^{nwells} I_j F_j + VCE(y) + VCG(y) + penaltyCost(Y, CC)\right) \tag{4.6}$$

where, I_j is the binary variable associated with j^{th} well and *nwells* is the number of wells, F_j is the fixed cost associated with the respective well ($75,000 for installing a new well, $25000 for putting an existing unused well into service

i.e. well 3 in this study), Y is vector of pumping rates for all wells and CC is maximum Contaminant Concentrations of the respective indicator parameters (C_{RDX}^{max} and C_{TNT}^{max}). The cost term $VCE(Y)$ is variable electric cost of operation, $VCG(Y)$ is variable cost of GAC units and $PenaltyCost(Y, CC)$ is for violating the concentration constraint, pumping constraint.

All the cost terms are computed in net present value (NPV) with the following discount function $NPV = \frac{cost_{iy}}{(1+r)^{iy-1}}$. where NPV is the net present value of a cost incurred in year iy with a discount rate of r=5%. The cost term is evaluated at the end of each year to account for annual discounting and to ensure that no costs are incurred after cleanup is achieved.

Constraints

The formulation includes model constraints that must be satisfied while the objective function is minimized. The modeling period consists of 1 management period of 4 years, with pumping rates kept throughout this period. The maximum concentrations of RDX and TNT in model layer 1 must be less than their respective cleanup targets by the end of 4 years $C_{RDX}^{max} \leq 2.1 ppb$ and $C_{TNT}^{max} \leq 2.8 ppb$. The total pumping rate cannot exceed 1100 gpm and the pumping capacity of individual wells must not exceed 400 gpm. RDX and TNT concentrations must not exceed their respective cleanup levels beyond a specified area when evaluated at the end of each management period. The total amount of pumping must equal the total amount of injection through the infiltration basins within an error tolerance (implemented in this study by 3^{rd} recharge basin getting the balance of (Total pumping)-(Total recharge)). The two additional formulation constraints are I is binary variable (0 or 1) and the fixed cost

constraint on all wells i.e. pumping capacity is only optimized if the respective well is chosen to be installed ($y_j \leq I_j * y_{max}$ for all wells).

Optimization-Modeling Approach

Figure 4.1 shows the location of existing extraction wells and the infiltration fields. Thus, the optimization goal is to identify a pumping strategy that lowers the C_{max} values of RDX and TNT to their respective cleanup targets of 2.1 and 2.8 ppb within 4 years while satisfying all the pumping constraints. The maximum allowed concentration constraint and the total pumping constraint are implemented by using the penalty functions hence the solution points not satisfying any of the two constraints (concentration and pumping) are penalized, which forces the algorithm to look for solution points that satisfy the above-mentioned constraints. The flow and transport model takes approximately 5 mins per simulation on a Pentium 2.2 Ghz PC.

4.3 New Algorithm : SIT-RBF Description

The goal of this study is to integrate an integer value optimization solver i.e. SIT with a continuous value solver to solve a fixed cost problem i.e. Response surface based, Stochastic RBF to solve a practical mixed value optimization problem. Response Surface based methods use a mathematical model as a surrogate for the optimization objective to guide the search for suitable parameters. The idea behind these methods is to fit an approximation to the objective function values from prior generations. A better initial fitting process in case of a computationally expensive functions sometimes can save a significant amount

of computational time by reducing the number of actual function evaluations needed during the run. The methodology suggested here tries to save on computational time by using the information (a response surface for $VC(Y)$ based on earlier simulations done) from one configuration (i.e. a particular configuration of integer values) for subsequent configurations of integer value variables (which are subset of initial, explained in 4.3.3). The integration methodology is described in detail after discussing the individual algorithms.

4.3.1 Search over Integers with Tabu (SIT)

The Search over integers with Tabu (SIT) algorithm is heuristic that incorporates a TAbu List as used in Tabu Search but does not use other aspects of Tabu Search. SIT uses *Tabu list* to prevent cycling back to some solution points related to previously generated value. The tabu status of a solution point is overridden if that point has an objective value that exceeds the best solution cost.

4.3.2 Response Surface based method

Response Surface based methods use a mathematical model as a surrogate for the optimization objective function to guide the search for suitable parameters.The idea is to fit an approximation to the objective function values from prior generations. The function approximation algorithms used in this paper use Radial Basis Functions (RBF)(Buhmann, 2003; Powell, 1999) to approximate the expensive objective. The purpose of using this RBF approximation is to reduce the computational expense of an optimization problem by allowing

the RBF approximation to screen out candidate points which are unlikely to be highly fit before the actual simulations are done. The response surface method used for this study is Multistart Local Metric Stochastic RBF (MLMSRBF) developed by Regis and Shoemaker (2007) which is called "Stochastic RBF" in this paper.

For an optimization problem, if $x_1, x_2....x_n\ x_k \in R^n$ are the previously evaluated set of parameter, a cubic RBF interpolation model (*Gutmann*) that approximates the objective function has the form

$$s_n(x) = \sum_{i=1}^{n} \lambda_i \phi(\|x - x_i\|_2) + b^T x + a \tag{4.7}$$

where, $\lambda_1, \lambda_2, ...\lambda_n \in R, b \in R^d, a \in R, \phi$ is a radial function and $\|.\|$ is the Euclidean norm. The coefficients of the above model are chosen such that the interpolant passes through all the design points. *Gutmann* (2001) evaluates the costly function at the corners of the domain d, so that there are 2^d points, where d is the dimension. This becomes too expensive for higher dimensional models, *Regis and Shoemaker* (2007) suggested the use of a Latin Hypercube Experimental design (LHD) for fitting the initial response surface. For d decision variables $(2 * d + 1)$ symmetric Latin hypercube design points were used for initial surface. In this study this initial fit for any function is done only once i.e. at the start of optimization run, but is used for all other possible integer variable configurations. For a detailed mathematical description of the algorithm the reader is referred to the paper by Regis and Shoemaker (2007). Stochastic RBF chooses the next evaluation point as the best point from a set of randomly generated candidate points. For algorithmic details, the reader is referred to the work by *Regis and Shoemaker, 2007*.

4.3.3 Description of SIT-RBF Algorithm

Flowchart 4.2 shows the general structure, while the steps below describe the integration methodology between the SIT and Stochastic RBF. The flowchart is shown here to give the reader a general idea about the integration methodology (explained in detail with Steps 1-6 below). The idea behind the integration is to sequentially use the information from a run with a particular set of integers to guide the search for a solution in a specified sub-set.

Let N_C be the number of continuous variables and N_I be the number of integer variables to be optimized. The variables N_C and N_I are different for the cases when there is no fixed cost associated with some of continuous variables (so $N_I \leq N_C$).

We will define D^k as the set of all binary vectors with N_I elements. We will define D^k as the set of all binary vectors in which exactly k of the elements are zero. The members of D^k are denoted as I^k. There are many possible I^k so we will denote them as $I^k(m)$, $m = 1, ..., |D^k|$ if we want to consider all members of D^k. Hence $I^0 = (1, 1, ..., 1) \in D^0$ and $I^k = (I^k_1, ..., I^k_{N_I}) \in D^k$.

Let $\Phi(I^k)$ be the set of all real vectors of length N_C where $y_i \leq I^k_j * y^{max}$, $j = 1, ..., N_I$.

The objective function for the algorithm SIT (Search over integers with Tabu) at a particular k is to minimize

$$\min_{I^k \in D^k} H(I^k) = \sum_j I^k_j F_j + C(I^k) \tag{4.8}$$

where I^k is a vector and I^k_j is its j^{th} component. Let $I^k_{best} \in D^k$ be the best integer configuration found for a particular k that minimizes equation 4.8. The SIT

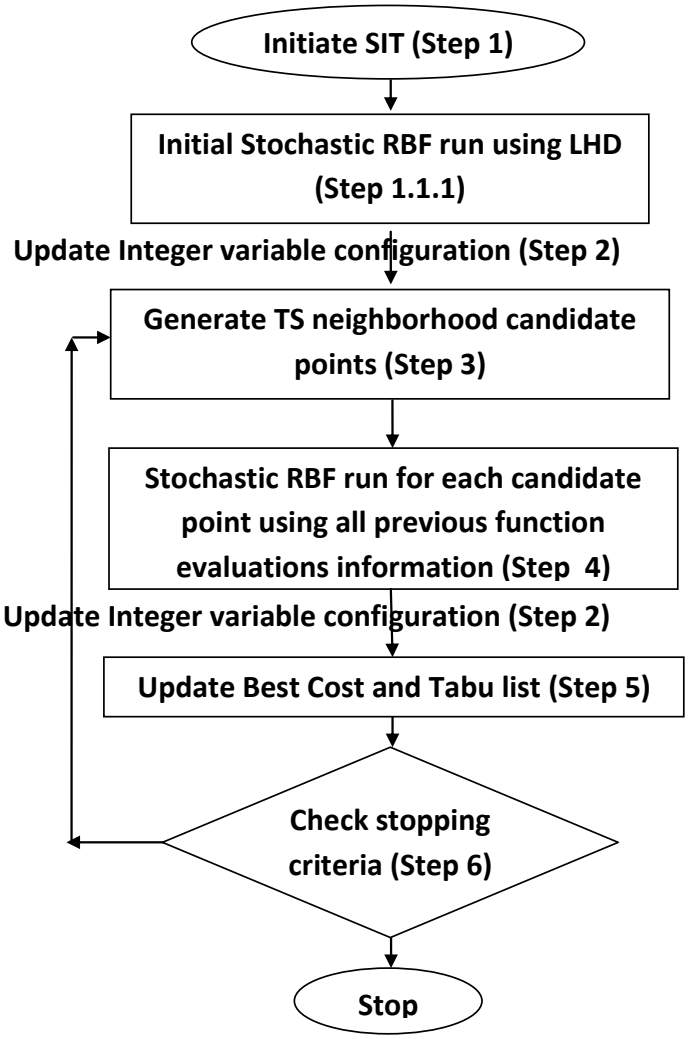


Figure 4.2: SIT-RBF methodology Flowchart with references to Algorithm Steps. Step 5 is the most computationally expensive. Step 3 to Step 5 can be done in parallel

evaluates $H(I^k)$ for all I^k in a neighborhood of current best solution I^k_{best}. $C(I^k)$ in equation 4.8 is computed by continuous global optimization with response surface for optimization problem from equation 4.9 and 4.10 (below).

$$C(I^k) = \min_{Y^k} \theta(Y^k) \tag{4.9}$$

subject to

$$0 \leq y_i^k \leq I_i^k y^{max} \; for \; (i = 1, ..., N_C) \tag{4.10}$$

where, $\theta(Y^k)$ is the cost associated with $Y^k = (y_1^k, y_2^k, ..., y_{N_C}^k)$.

Evaluating θ(Y) in equation 4.10 requires a computationally expensive simulation and can include penalty functions to incorporate constraints. We use Stochastic RBF (Regis and Shoemaker, 2007) as the continuous global optimization method. Stochastic RBF here uses an RBF approximation $Q(Y)$ for all $Y \in R^{N_C}$. $\mathcal{A}$ contains all the points $(Y, \theta(Y))$ that have been evaluated by costly simulation of θ and $Q(Y)$ is a RBF spline surface that has interpolated all the points in the current set $\mathcal{A}$. Recall that I^k refers to a binary vector $I = (I_1^k, ..., I_n^k)$ where exactly k of the $I_i^k = 0$. Hence in the neighborhood of the current best solution in Step 3, any flip of a vector bit from 0 to 1 must be matched by a flip of another bit from 1 to 0.

Let $y_{best}(I^k)$ be the vector of continuous variables that gives the minimum of equation 4.9 and 4.10 (from Stochastic RBF subjected to tabu constraints), so

$$C(I^k) = \theta(Y_{best}(I^k)) \tag{4.11}$$

where $Y_{best}(I^k)$ satisfies equation 4.10. The SIT search continues to find the best solution (I_{best}^k, Y_{best}^k) within *maxevals* function evaluations. Other inputs for the algorithm include N_{nei}, the neighborhood size for SIT and tenure length i.e tabu list length. The Tabu list (ϕ) prevents acceptance of an $I^k(m)$ that involves a swap of 0 and 1 between elements (j, k) that are on a tabu list unless $H(I^k(m))$ is better than the best solution so far.

Steps in SIT-RBF Algorithm

The following gives the exact steps. Followed by steps is a discussion/explanation of them.

Step-1. SIT Initialization :

1.1. Global optimization algorithm (Stochastic RBF) is run in R^{N_C} dimensional continuous space with $k = 0$, $I_i^0 = 1$ for $i = 1, ..., N_I$.

1.1.1. Compute the expensive simulation function $\theta(Z_i)$ $i = 1, ..., (2N_C + 1)$ where Z_i are points from a symmetric Latin hypercube design points. Construct the initial RBF surface $Q(Y)$ that interpolates the points $(Z_i, \theta(Z_i))$.

1.1.2. Save all function evaluations and implement Stochastic RBF to find $Y_{best}^0(I^0) = (Y_{best,1}, ...Y_{best,N_C})$, which is best solution given $I^0 = [1, 1, ...1]$. The continuous cost is $C(I_{best}^0)$ (equation 4.9 with k=0).

1.2. Add respective fixed cost to all evaluated points to compute $H(I^0) = \sum_{i=1}^{N_I} I_i^0 F_i + C(I^0)$ (equation 4.8).

1.3. Set $H_{best} = H(I^0)$ and initialize Tabu list, ϕ (empty set).

Step-2. Update k.

2.1. Set $l = 0$;

2.2. for $i = (1, ..., N_I)$

If $Y_{best,i}^k(I_{best}^k) = 0$ then $I_i^{new} = 0$ and $l = l + 1$.

If $Y^k_{best,i}(I^k_{best}) > 0$ then $I^{new}_i = 1$.

2.3. If $l \neq k$ go to step 2.4, Otherwise, if $l = k$ choose a random integer q from the set $T = \{J|J \text{ is integer and } I^{new}_J \neq 0\}$. Then set $I^{new}_q = 0, l = l + 1$ and go to step 2.4 .

2.4. Assign $k = l$.

Step-3. Generate the neighborhood configurations in integer space i.e. $I^k(1), ..., I^k(N_{nei})$; where N_{nei} is the number of individuals in the neighborhood.

Step-4. Do steps 4.1 − 4.2 for each neighborhood configuration, $I^k(m)$ for $m = (1, ..., N_{nei})$ (As discussed in section 4.3.4 this step can be done in parallel on N_{nei} processors).

4.1. Implement Stochastic RBF to find $Y^k_{best}(I^k(m))$, which is best solution given $I^k(m)$ i.e. $C(I^k(m))$ from equations 4.9-4.10. During this step Stochastic RBF is using a response surface $Q(Y)$ that interpolates all previous simulations of $(y_i, \theta(y_i))$ (stored in set $\mathcal{A}$) to solve equation 4.9. Add all costly simulation results $(y_i, \theta(y_i))$ during the stochastic RBF search to the set $\mathcal{A}$.

4.2. Add respective fixed cost to all evaluated points at the end of optimization run to compute $H(I^k) = \sum_{(j=1)}^{N_I} I^k_j F_j + C(I^k)$ (equation 4.8).

Step-5. Among the best solution $H^k(I^k(m))$ evaluated, pick the best $I^k(m)$ and denote it I^k_{min} and $H^k_{min} = H^k(I^k_{min})$. In this step $I^k(m)$ is tabu of swapping of 0 and 1 occurring at locations that are on the tabu list.

5.1. If the minimum cost configuration(element swap) is not in the tabu list (ϕ) update the best solution point $Y^k_{best}(I^k_{best}) = Y^k_{min}(I^k_{min})$. Otherwise update only if minimum cost is less than current best cost (H_{best})

5.2. If $H^k_{min} \leq H_{best}$ update H_{best}=H^k_{min}.

5.3. Update the tabu list (ϕ) i.e. add the indices of elements swapped($Step - 3$) if the tabu list is not full; otherwise, delete the oldest entry in tabu list and record the swap.

Step-6. Check Stopping criteria i.e *maxevals* otherwise go to step 2.

4.3.4 Discussion of SIT-RBF Algorithm

Step 1 runs the Stochastic RBF without any prior information to find optimal continuous variable values (Y^0_{best}). The initial RBF setup is done using Symmetric Latin Hypercube Sampling (*Step 1.1.1*). *Step 1.1.2* starts with a configuration which has maximum number of integer value variables (N_I) needed to solve the continuous value optimization problem i.e. the configuration with maximum fixed cost. This is done by setting all integer variables (I^k), to 1 (i.e. all facilities on and $k = 0$). Then Stochastic RBF optimization is run to find optimal solution of N_C-dimensional continuous value optimization problem in Section 4.2. Stochastic RBF runs only on continuous value decision variables i.e. objective function ($\theta(Y^k)$) from equation 4.9. In other words this step tries to find optimal pumping rates (Y^0_{best}) for $I^0 = 1; k = 1, ...N_I$. Once optimal continuous variable values (Y^0_{best}) are determined installation cost based on the integer variable configuration is added to the cost values (*Step 1.2*) from the Stochastic RBF (equa-

tion 4.8). *Step 1.3* initiates the H_{best} and tabu list (ϕ) , which saves the indices of elements swapped for neighborhood generation in *Step 3*.

In *Step 2* the variable k is updated. In *Step 2.1 and Step 2.2*, l is used to count the number of zeros in $Y^k_{best}(I^k_{best})$. If there are only k zeros in $Y^k_{best}(I^k_{best})$ then *Step 2.3* randomly selects a component q where I^k_q will be zero in starting the next *Step 3*. In other words if the optimal solution point ($Y_{best,i}$ from *Step 1*) to the continuous value problem $\theta(Y)$ has a zero value in particular dimension i.e $Y_{best,i} = 0; i = 1, ..., N_I$, then the i^{th} element of integer vector is updated $I^{new}_i = 0$. If there is no change in integer configuration ($k = l$), a random zero is introduced in integer space to choose a random integer q from the set of non-zeros in I^k_{best} and set $I^{new}_q = 0$. For example the solution obtained from Stochastic RBF might be of two forms i.e. some continuous value variable values of zero (e.g. with integer space [1,0,1,0,1]) or without any zeros (e.g. with integer space [1,1,1,1,1]). In case the optimal solution from Stochastic RBF looks like [1,1,1,1,1], the methodology introduces a zero in a randomly picked dimension from integer space, otherwise (i.e. for something like [1,0,1,0,1]) this step is skipped.

Step 3 then generates a set of possible neighborhood configurations by swapping 0 and 1 respectively. Each of these configurations $I^k(1), ..., I^k(N_{nei})$ has its own respective installation cost based on the integer configuration ($I^k(m)$) selected for that particular case. In this study the neighborhood size N_{nei} was chosen such that *step 4* can be done in parallel (done in this study) i.e. N_{nei} equals the number of processors used (limited to 8-16 for all tested functions).

The neighborhood generated ($I^k(1), ..., I^k(N_{nei})$) in *Step 3* is then explored in *Step 4*. Here in *Step 4.1* Stochastic RBF is run for each integer configuration ($I^k(j), j = 1, ..., N_{nei}$) with all the information in set $\mathcal{A}$ as input data for the initial

Radial Basis Function (RBF) fit and subsequent RBF function value approximations. This step finds the optimal solution to the continuous value problem (Y^k_{best}) for the neighborhood configurations $I^k(m)$; $m = (1, ..., N_{nei})$ generated in *Step 3*. The number of actual function evaluations done in this step are considerably less than *Step 2* as the prior information (set $\mathcal{A}$) helps in fitting and getting the estimates of actual function value from the RBF. Once optimal continuous variable values (Y^k_{best}) are determined installation cost based on the integer variable configuration is added to the cost values (*Step 4.2*).

Step 5 updates the best cost (H_{best}) and tabu list (ϕ). If the minimum solution in the neighborhood I^k_{min} is not in the tabu list then the best solution point (I^k_{best}, Y^k_{best} is updated to equal I^k_{min}, Y^k_{min}. Otherwise it is updated only if minimum cost H^k_{min} is less than H_{best}. *Step 5.2* updates the best cost (H_{best}) if $H(I^k_{min})$ is less than best solution point. In *Step 5.3* for tabu list updating, the indices of elements swapped in *Step 3* for neighborhood generation are added to ϕ if the tabu list is not full. Otherwise, oldest entry in tabu list is replaced with the newest. *Step 6* controls the termination criteria, which in this study was the total number of function evaluations (*maxevals*).

4.4 Alternative Algorithms for Fixed Cost Problems

4.4.1 Genetic Algorithm

The results from the suggested methodology (SIT-RBF) are compared with Genetic algorithms as GA's are quite popular for these type of problems. This study uses Genetic Algorithm (NSGA-II) for comparison with Tabu-Stochastic

RBF integration. NSGA is a multi-objective Genetic algorithm based on non-dominated sorting algorithm developed by Deb et. al. (2002). Deb et.al. in their study demonstrated that the NSGA performed better than various other GA based algorithms. Also NSGA is designed to solve mixed value (integer variables and continuous value variables) problems. In this study NSGA is used on a single objective problem formulation with constraints in order to do the comparison with the suggested methodology. In other terms NSGA was implemented to solve a mixed integer valued optimization problem (binary and continuous decision variables). The NSGA-II algorithm includes elitism and uses a binary tournament operator for selection. The crossover and mutation operations are performed independently on both real and binary coded variables. Box constraints are enforced in generation of candidate design variables whereas the constraints on binary/integer variables are implemented as non-negative functions. Numerical values of the various parameters i.e. population size, number of generations, crossover/mutation probability were chosen by taking into account considerations from (Reed et. al. 2000; Mayer et. al. 2002).

4.5 NOMAD(Nonsmooth Optimization by Mesh Adaptive Direct Search)

NOMAD a C++ implementation of MADS algorithm is used in this study for comparison of results with the SIT-RBF methodology. It is a freely available at http://www.gerad.ca/nomad/Project/Home.html (under General Public License) constrained optimization tool for black-box functions developed by Abramson et.al.(2009). Mesh Adaptive Direct Search (MADS) algorithm

for non-linear optimization is based on Generalized Pattern Search algorithms. These algorithms are iterative with each iteration consisting of two phases: an Search and a local Poll phase. In search phase the objective function is evaluated over a finite number of mesh points to find a new point with lower objective function value, if algorithm fails to find an improved mesh point it calls the POLL procedure where a barrier objective function (at neighboring mesh points) is evaluated to find a lower function value. If POLL also fails to find an improved point then the mesh is refined and the procedure is repeated. NOMAD implementation can be parameterized to use Response Surface Surrogate approximations for continuous value decision variables but not for mixed integer value problems. For more details reader is referred to Abramson et. al. (2009).

4.6 Results

The goal of this study is to integrate a integer variable optimizer SIT with global continuous value optimizer (Stochastic RBF). Thus the experimental runs were designed to test the suggested integration methodology (SIT-RBF) against two open source codes for Mixed Integer value problem (MIVP) optimizers, NSGA (Deb et. al. (2002)) and NOMAD (Abramson et.al.(2009)). Because of the stochastic nature of suggested methodology (SIT-RBF) and NSGA, 10 trials were run for both the hypothetical test problems and 5 for the Umatilla test problem. The results presented are averaged over all trials. The next section presents the results for the algorithm comparison with discussion about the algorithm.

4.6.1 Algorithm Comparison

The results for this study uses convergence plots for the objective function values to compare the three algorithms. On the plots the points (e, v) correspond to e=number of function values that have been evaluated and v is the average over all trials of the best objective function value obtained in e. Figure 4.3 and 4.4 compare the performance of algorithms on test function and hypothetical groundwater test problem. Figure 4.5 compares the algorithm performance for Umatilla groundwater problem. The lowest curves are best since they indicate the algorithm got closer to the minimum solution in fewer evaluations than the other algorithms.

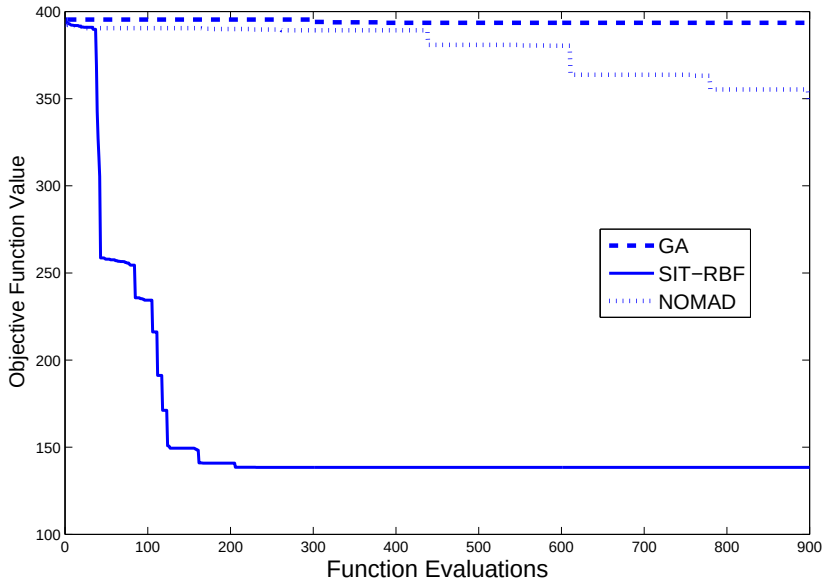


Figure 4.3: SIT-RBF Algorithm performance comparison for test function (Averaged over 10 trials). Lowest Curve is best

For all functions, for a fixed number of total allowed function evaluations the SIT-RBF integrated methodology significantly outperforms the other two

methods (NOMAD and NSGA). NOMAD and NSGA performed relatively better for Umatilla as compared to test problems i.e. the best cost solution seems to improving. An explanation of success of SIT-RBF methodology in locating a good solution with specified function evaluations can be attributed to two reasons. First one the sequential use of previously evaluated points to construct RBF surfaces i.e. relatively better RBF surfaces lead to improved RBF approximations. The second reason is the underlying global nature of Stochastic RBF i.e. Stochastic RBF is a global optimizer thus it is better equipped to solve global optimization problems. The two other tested algorithms need significant number of additional function evaluations to locate a good solution point.

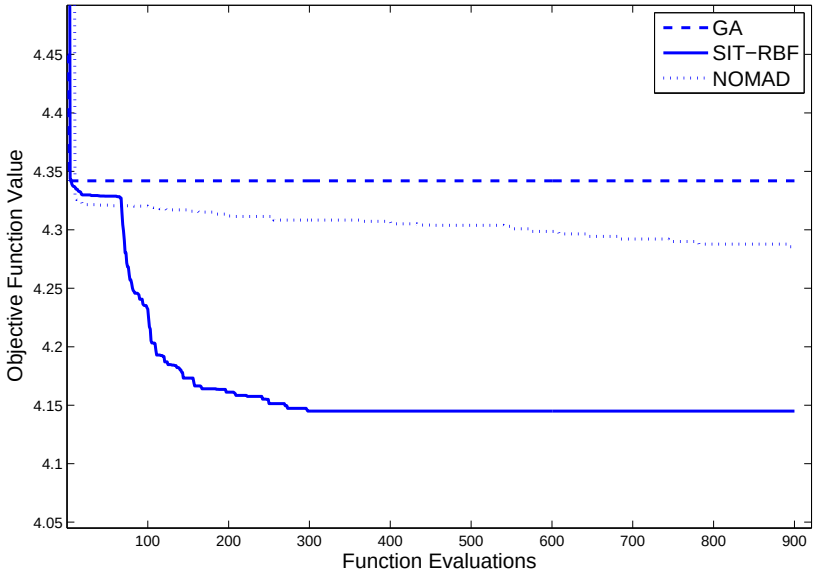


Figure 4.4: SIT-RBF Algorithm performance comparison for groundwater test problem (Averaged over 10 trials). Lowest Curve is best

In this section we try to analyze the success of integrated methodology using a hypothetical problem (5 continuous and 5 integer variables). The idea behind of whole methodology is based on using the function evaluation data for one

configuration for fitting a response surface with a different configuration. Another important factor to be taken into account is that the Stochastic RBF method never deals or sees any fixed cost. Fixed costs are added once the Stochastic RBF run is finished and subtracted when inputting the prior information Stochastic RBF for next configuration. When initial Stochastic RBF is run on this hypothetical problem ($N_C = N_I = 5$) the resulting solution (Y^k_{min}) could have between 0 and k zeros. Since all the solutions for Y^k have N_C elements (zero or non zero), the points Y^k can contribute to the building of the N_C dimensional response surface $Q(Y)$ that approximates $\theta(Y)$ in equation 4.9. Thus the previously computed expensive function evaluations ($\theta(Y)$) for $k < K$ (where K is any index) can be used to approximate the Response surface resulting in the step with $k = K$. Thus with prior information of actual function evaluations, the subsequent optimization runs for higher values of k require relatively many fewer function evaluations. Other reasons for the success of this methodology can be attributed to success of Stochastic RBF as a global optimizer of $\theta(Y)$. It has been tested successfully on various different types of problems (Regis and Shoemaker, 2007)

Umatilla Groundwater problem

This part discusses the quality of results obtained for Umatilla groundwater site for the three optimization methods Genetic Algorithm (GA), NOMAD and the new method SIT-RBF described in this paper. Table 4.1 lists the pumping rates for one optimization trial. The table 4.1 also lists the respective objective function value for the respective optimal solution points.

The table 4.1 highlights the difference in quality of results obtained by differ-

ent algorithms in terms of wells chosen and objective function values at the end of optimization trial (median). The results obtained from SIT-RBF methodology have much lower objective function values as compared to GA and NOMAD. The difference in magnitudes of objective function value is due to penalty function i.e. $C_{RDX}^{max} \leq 2.1ppb$ and $C_{TNT}^{max} \leq 2.8ppb$ at the end of simulation. The constraint violations and corresponding penalties are also listed in the table. The median trial solution (table 4.1) for SIT-RBF methodology chooses 4 pumping wells and two of these wells have no fixed cost (pre-existing wells). SIT chooses two new wells where as NOMAD chooses 4 new pumping wells thus more fixed cost. Thus in comparison to NOMAD, SIT-RBF chooses an option with less fixed cost while satisfying all the constraints (concentration constraints on RDX and TNT). Genetic algorithm performs the worst it uses all but one well and also the total pumping it suggests is considerably more than the pumping policy suggested by SIT-RBF and NOMAD methods.

Figures 4.6, 4.7, 4.8 and 4.9 show the contour plots for the two contaminants. The color bar on all plots shows the concentrations for the two contaminants RDX and TNT, respectively. Figure 4.6 plots the initial contour plot for contaminants TNT and RDX, here the maximum concentration of RDX and TNT are 25ppb and 80ppb, respectively (as shown by color bar). The figure 4.7 shows the contour plots in case when no management (pumping) policy is implemented. Here contaminant concentration decreases slightly (concentration of 20ppb and 70ppb for RDX and TNT) mainly due to dispersion of contaminant. Figure 4.8 shows the contour plot at the end of simulation period for the optimal solution found (median trial) by SIT-RBF methodology (with maximum concentration of 2ppb and 2.5ppb for RDX and TNT, respectively). Figure 4.9 shows the same for optimal policy by NOMAD with maximum concentration of 3ppb and 2ppb for

Table 4.1: Algorithm Comparison in terms of pumping rates for Umatilla problem for an optimization (median for each algorithm with 1600 simulations) trial. Only pumping wells have fixed cost so $N_C = 11$ and $N_I = 8$. The Objective function is from equation 4.6.

Well index	**Pumping Rates (GPM)**		
	SIT-RBF	**NOMAD**	**GA**
Pumping Well 1	-	-	-
Pumping Well 2	395	80	164
Pumping Well 3	329	23	368
Pumping Well 4	-	-	246
Pumping Well 5	348	399	132
Pumping Well 6	95	399	26
Pumping Well 7	-	218.9	154
Pumping Well 8	-	65	234
Recharge Basin 1	-	-	399
Recharge Basin 2	399	100	646
Recharge Basin 3	768	1065	279
Total pumping	1167	1165	1324
Objective Function	977	6309	1.5×10^5
Penalty Function	0	4500	1.49×10^5
*Max$(0, C_{RDX}^{max} - 2.1ppb)$	0	1.1	4.2
*Max$(0, C_{TNT}^{max} - 2.8ppb)$	0	0	3.4

**Represents the two concentration constraints $C_{RDX}^{max} \leq 2.1ppb$ and $C_{TNT}^{max} \leq 2.8ppb$ at the end of simulation*

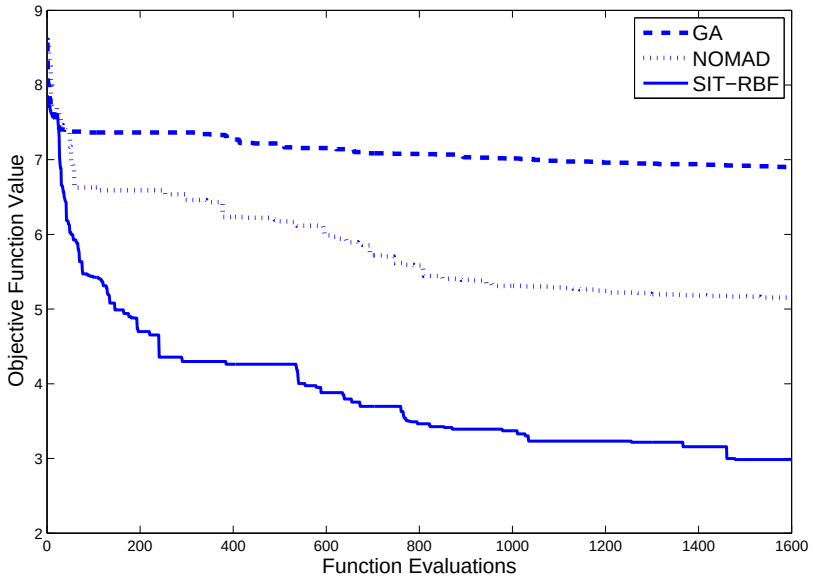
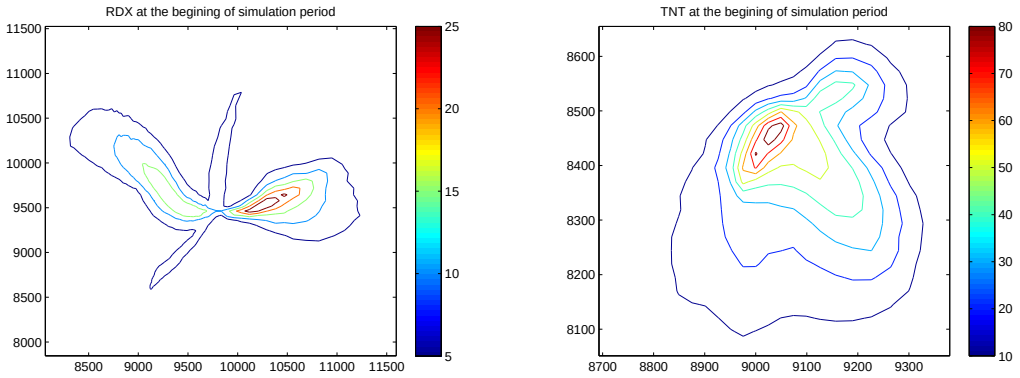


Figure 4.5: SIT-RBF Algorithm performance comparison for Umatilla groundwater problem (Averaged over 5 trials). Lowest Curve is best

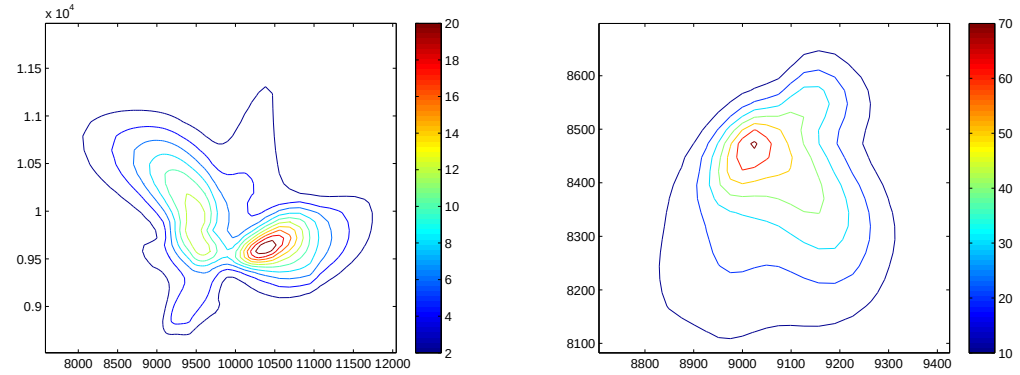
RDX and TNT, respectively. Comparison of figure 4.7 with 4.8 and 4.9 emphasizes on the need of a management policy for remediation. The contour plots shows that the optimal policy from NOMAD run fails to satisfy the concentration constraints for RDX i.e. $C_{RDX}^{max} \leq 2.1 ppb$ as compared to SIT-RBF methodology which satisfies both constraints. This constraint violation results in higher objective function value for NOMAD solution. The GA solution violates both the constraints i.e for RDX and TNT, thus resulting in very high objective function value.

Becker et.al. (2006) in their study emphasize on the need of optimization algorithms for groundwater remediation problems. Detailed computational results are not provided by the analysis of the same Umatilla problem by Becker et.al. (2006). Their paper implies that up to 8000 objective function simulations



(a) (b)

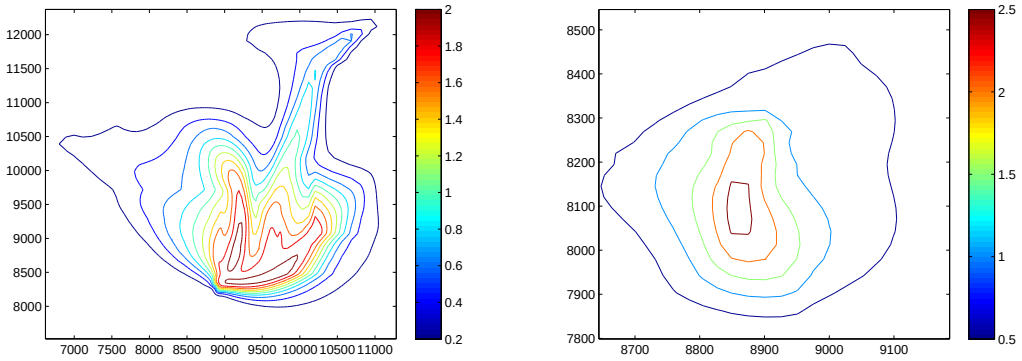
Figure 4.6: Initial Contamination Contour Plot (Color bar represents the respective concentration): (a) RDX ; (b) TNT;



(a) (b)

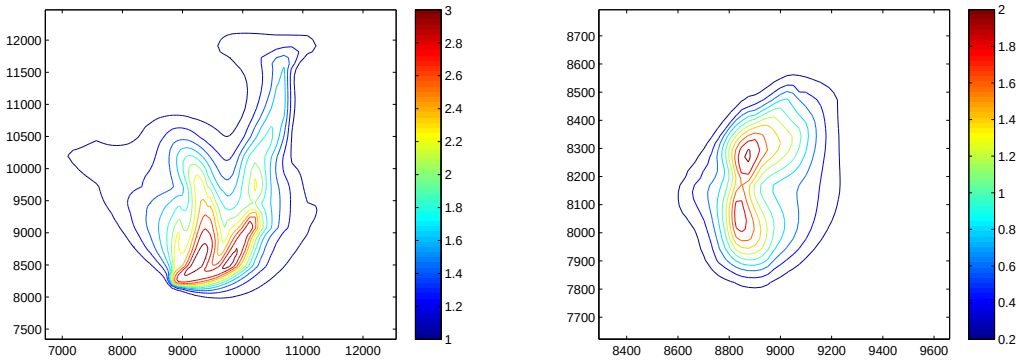
Figure 4.7: Contamination Contour Plot without any management policy (Color bar represents the respective concentration): (a) RDX ; (b) TNT;

were done to reach the objective function value of same order as obtained by SIT-RBF. The optimization approaches ((Peralta (2003) and Zheng and Wang 2002a) used by Becker et. al. (2006) were designed specifically to run with older versions of MODFLOW (Harbaugh et.al., 2000) and MT3D (Chunmiao Zheng (1990)). Hence could not be implemented for this study. They do not report the results of multiple trials or the relationship between the objective function and



(a) (b)

Figure 4.8: Contamination Contour Plot for optimal solution from SIT-RBF methodology (Color bar represents the respective concentration): (a) RDX ; (b) TNT;



(a) (b)

Figure 4.9: Contamination Contour Plot for optimal solution from NOMAD (Color bar represents the respective concentration): (a) RDX ; (b) TNT;

the number of evaluations.

4.7 Conclusion

The results of optimal policy design for pump and treat system (i.e. continuous value pumping rate variables) illustrate the success of SIT-RBF integrated methodology. The SIT-RBF tries to use sequentially the expensive function evaluation information (candidate points with respective function values) from different integer variables configuration to build RBF surfaces for new configuration. This prior information improves RBF approximations hence reduces the amount of function evaluations to be done to find optimal value of continuous value variables corresponding to new configuration. The study compared the suggested SIT-RBF methodology with NSGA and NOMAD for a mixed integer value problems. NOMAD got to solution point of same magnitude as by SIT-RBF with 5000 function evaluations. Whereas our suggested methodology SIT-RBF used 1600 (1600/5000=32%) function evaluations. The results presented indicate that under limited computational budget, the integrated methodology was much more effective than the using any stand-alone mixed integer value problem solver.

These results are based on groundwater pump and treat system problems and some test problems do not prove that integrated methodology method will always be better than other algorithms MIVP problems. However the results still suggest that innovative integration with recent methods such as RBF based global optimization methods should be considered as alternatives to widely used methods such as evolutionary algorithms and mixed value nonlinear methods.

BIBLIOGRAPHY

[1] Abramson M. A. and C. Audet. *Convergence of mesh adaptive direct search to second-order stationary points*. SIAM Journal on Optimization, 17(2):606619, 2006.

[2] Abramson M. A., C. Audet, G. Couture, J. E. Dennis, Jr., and S. Le Digabel. *The NOMAD project*. Software available at http://www.gerad.ca/nomad.

[3] Ahlfeld,D. (1987). *Designing contaminated groundwater remediation systems using numerical simulation and nonlinear optimization*, Ph.D. dissertation, Princenton Univ., Princeton, N.J.

[4] Ahlfeld, D. P. and A. E. Mulligan: 2000, *Optimal Design of Flow in Groundwater Systems*. San Diego: Academic Press.

[5] Aly, A. H., and Peralta, R. C. (1999b). *Optimal design of aquifer cleanup systems under uncertainty using a neural network and a genetic algorithm.* Water Resour. Res., 35(8), 25232532.

[6] Aral, M. M., and Guan (1996), *Optimal groundwater remediation design using differential genetic algorithm*, J. Computational Methods in Water Resources XI (1), 349-357, Computational Mechanics Publications.

[7] Atwood, D., & Gorelick, S. (1985) *Hydraulic Gradient Control for Groundwater Contaminant Removal*. J. of Hydraul. 76, 85-106.

[8] Becker, D., Minsker, B., Greenwald, R., Zhang, Y., Harre, K., Yager, K., Zheng, C. and Peralta, R. (2006),Reducing Long-Term Remedial Costs by Transport Modeling Optimization. Ground Water, 44: 864875. doi: 10.1111/j.1745-6584.2006.00242.x.

[9] Buhmann, M. D. (2003) *Radial Basis Functions. Cambridge University Press* , Cambridge, UK

[10] Chang, L.-C., Shoemaker, C. A., and Liu(1992), P. L.-F., *Optimal time-varying pumping rates for groundwater remediation: Application of a constrained optimal control algorithm*. Water Resources Research, 28(12), 3157-3173.

[11] Cressie, N. 1993. *Statistics for Spatial Data* . John Wiley & Sons,New York.

[12] Dibike, Y.B., Solomatine, D., and Abbott, M.B. *On the encapsulation of numerical hydraulic models in artificial neural networks*. J. Hydraulic Research, 37, 147-161,1999.

[13] Deb K (2000) An efficient constraint handling method for genetic algorithms. Comput Methods Appl Mech Eng 186(24):311338

[14] Deb K (2003) KanGal Homepage, Indian Institute of Technology Kanpur. http://www.iitk.ac.in/kangal

[15] Deb K, Agrawal RB (1995) Simulated binary crossover for continuous search space. Complex Syst 9:115148

[16] Deb K, Goel T (2001) Controlled Elitist Non-dominated sorting genetic algorithms for better convergence.In: Zitler E, Deb K, Thiele L, Coello-Coello C, Corne D (eds.) Proceedings of the first international conference on evolutionary multi-criterion optimization EMO 2001, pp 6781

[17] Deb K, Pratap A, Agarwal S, Meyarivan T (2000) A fast and elitist multi-objective genetic algorithm:NSGA-II, KanGal Report 200001, Indian Institute of Technology Kanpur, Kanpur, India

[18] Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multi-objective genetic algorithm:NSGA-II. IEEE Trans Evol Comput 6(2):182197

[19] Gill,P.D., Murray,W., Saunders,M.A. and Wright,W.H. (1986). Users guide for NPSOL

[20] Gorelick,S.M., Remson,I. and Cottle,R.W. (1979). *Management model of a groundwater system with a transient pollutant source*, Water Resour. Res., Vol.15, No.15, 1243-1249.

[21] Gorelick,S.M. (1982). *A model for managing sources of groundwater pollution*. Water Resour. Res. Vol.18, No. 4, 773-781.

[22] Gorelick,S.M., and Remson,I. (1982). Optimal location and management of waste disposal facilities affecting groundwater quality, Water Resources Bulletin, Vol.18, No.1, 43-51, 1982.

[23] Gorelick, S.M., Voss, C.I., Gill, P.E., Murray, W., Saunders, M.A. and Wright, M.M. (1984). 'Aquifer reclamation Design: The use of contaminant transport simulation combined with nonlinear programming', Water Resour. Res., Vol.20, No.4, pp.415-427.

[24] El Harrouni, K., Ouazar, D., Walters, G. A., and Cheng, A. H.-D. *Groundwater optimization and parameter estimation by genetic algorithm and dual reciprocity boundary element method*. Engineering Analysis with Boundary Elements, 18(4), 287-296, 1996.

[25] Harbaugh, A.W., Banta, E.R., Hill, M.C., and McDonald, M.G., 2000, MODFLOW-2000, *The U.S. Geological Survey modular ground-water model – User guide to modularization concepts and the Ground-Water Flow Process*: U.S. Geological Survey Open-File Report 00-92, 121 p.

[26] Hemker Thomas,Fowler Kathleen R.,Farthing Matthew W. and Stryk Oskar von *A mixed-integer simulation-based optimization approach with surrogate functions in water resources management*. Optimization and Engineering Volume 9, Number 4, 341-360, 2008.

[27] Hemker, T., K. R. Fowler, and O. von Stryk: 2006a, *Derivative-Free Optimization Methods for Handling Fixed Costs in Optimal Groundwater Remedi-*

ation Design. In: Proc. of the CMWR XVI - Computational Methods in Water Resources.

[28] Holmstrm Kenneth, Quttineh Nils-Hassan, Edvall Marcus M., *An adaptive radial basis algorithm (ARBF) for expensive black-box mixed-integer constrained global optimization*. Optim Eng (2008) 9: 311339 DOI 10.1007 /s11081-008-9037-3.

[29] Holland, J.H. 1975. *Adaptation in Natural and Artificial Systems*. Ann Arbor, Michigan: University of Michigan Press.

[30] Huang, C. and A. S. Mayer: 1997, *Pump-and-treat optimization using well locations and pumping rates as decision variables*. Water Resources Research 33(5), 10011012.

[31] Glover, F. 1989. *Tabu SearchPart I*, ORSA. Journal of Computing 1, no. 3: 190206.

[32] Glover, F. 1986. *Future paths for integer programming and links to artificial intelligence*. Computation and Operations Research 13, no. 5: 533549.

[33] Goldberg, D.E., 1989, *Genetic algorithms in search, optimization, and machine learning*, Addison-Wesley publishing company inc., Massachusetts, 432p.

[34] Gorelick, S.M., Voss, C.I., Gill, P.E., Murry, W., Saunders, M.A. and Wright, M.H., 1984, *Aquifer reclamation design: The use of contaminant transport simulation combined with nonlinear programming*, Water Resources Research, 20, 415427.

[35] Jin, X., G. Mahinthakumar, E. M. Zechman, and S. Ranjithan, *A Genetic Algorithm-based Procedure for 3D Source Idenitification at the Borden Emplacement Site*, Journal of Hydroinformatics, 11(1), pp. 51-64, 2009 Math. Programming 79(3) 397414.

[36] Kelley, C. T. (1999) *Iterative Methods for Optimization*. SIAM, Philadelphia, Pennsylvania, USA.

[37] Mayer, A. S., C. T. Kelley, and C. T. Miller: 2002b, *Optimal Design for Problems Involving Flow and Transport in Saturated Porous Media*. Advances in Water Resources 12, 12331256.

[38] McKinney, D. C., and Lin, M.-D. (1995). *Approximate mixed-integer nonlinear programming methods for optimal aquifer remediation design.* Water Resour. Res., 31(3), 731740.

[39] Metropolis, N., A.W. Rosenbluth, M.N. Rosenbluth, A.H. Telle and E. Teller. 1953. *Equations of state calculations by fast computing machines*. Journal of Chemistry and Physics 21 no. 6: 10871091.

[40] Murtagh,B.A., and Saunders,M.A. (1980). MINOS/AUGMENTED users manual, syst.optimiz.Lab.Tech.Rep. 80-14, pp., Dep. Of Oper. Res., Stanford, Calif.

[41] Mugunthan P., Shoemaker C., and Regis R.,*Comparison of function approximation,heuristic and derivative-based methods for automatic calibration of computationally expensive groundwater bioremediation models*, Water Resour. Res., 41 (2005).

[42] Naval Facilities Engineering Command (NAVFAC) technical report TR-2237-ENV, 2004.

[43] Peralta, R.C. 2003. *SOMOS Simulation/Optimization Modeling System*. In Proceedings, MODFLOW and More 2003: Understanding through Modeling, 2003, ed. E.P. Poeter, C.Zheng, M.C. Hill, and J. Doherty, 819823. Golden, Colorado: International Groundwater Modeling Center.

[44] Powell, M. J. D. Muller, M. , Buhmann, M. , Mache, D. and Felten, M. (eds) (1999) *Recent research at Cambridge on radial basis functions*. New Developments in Approximation Theory 132 , pp. 215-232. International Series of Numerical Mathematics , Birkhauser Verlag, Basel, Switzerland

[45] Peralta Richard C., Kalwij Ineke M., and Shengjun Wu, *Practical Remedial Design Optimization for Large Complex Plumes* J. Water Resour. Plng. and Mgmt. 134, 422 (2008), DOI:10.1061/(ASCE)0733-9496(2008)134:5(422).

[46] Reed, P., B. Minsker, and D. E. Goldberg: 2000, *Designing a competent simple genetic algorithm for search and optimization*. Water Resources Research 36(12), 37573761.

[47] Regis, R. G. & Shoemaker, C. A. (2004) *Local function approximation in evolutionary algorithms for the optimization of costly functions*. IEEE Trans. Evolutionary Computation 8(5), 490505

[48] Regis, R. G. & Shoemaker, C. A. *A Stochastic Radial Basis Function Method for the Global Optimization of Expensive Functions*. INFORMS Journal on Computing, Vol. 19, No. 4, pp. 497-509, Fall 2007.

[49] Regis, R. G. & Shoemaker, C. A. *Improved Strategies for Radial Basis Function Methods for Global Optimization*. Journal of Global Optimization, Vol. 37, No. 1, pp. 113-135, 2007.

[50] Remson, I. and Gorelick, S.M. (1980). *Management models incorporating groundwater variables, in Operations Research in Agriculture and Water Resources*, edited by D.Yaron and C.S.Tapiero, North-Holland, Amsterdam, pp.42-75.

[51] Rinnooy Kan, A. H. G. & Timmer, G. T. (1987) *Stochastic global optimization methods*. Part II: Multi level methods. Mathematical Programming 39, 5778.

[52] Sadiq M Sait and Habib Youssef.(1999), *Iterative Computer Algorithms with Applications in Engineering IEEE Computer Society*, Los Alamitos, California.

[53] Solomatine, D.P. *Genetic and other global optimization algorithms comparison and use in calibration problems*. Proc., 3rd Int. Conf. on Hydroinformatics, 1021-1027,1998.

[54] Tolson, B. A., and C. A. Shoemaker (2007), *Dynamically dimensioned search algorithm for computationally efficient watershed model calibration*, Water Resour. Res., 43, W01413, doi:10.1029/2005WR004723. Jan. 2007

[55] The Mathworks, Inc. (2009a) *Genetic Algorithm and Direct Search Toolbox for Use with MATLAB: Users Guide*, version 1. The Mathworks, Inc., Natick, Massachusetts, USA.

[56] The Mathworks, Inc. (2009a) *Optimization Toolbox for Use with MATLAB: User's Guide*, version 3. The Mathworks, Inc. Natick, Massachusetts, USA

[57] Yoon, J.-H., C. A. Shoemaker. 1999. *Comparison of optimization methods for ground-water bioremediation*. J. Water Resources Planning Management 125(1) 5463

[58] YoonJ .H and Shoemaker C.A., *Improved real-coded GA for groundwater bioremediation*, J. Water Resour. Plan Management, vol. 125, no. 1, pp.54-63,1999.

[59] Wang, M., and Zheng, C. *Optimal remediation policy selection under general conditions*. Ground Water, 35(5), 757-764, 1997.

[60] Watkins Jr, D. and D. C. McKinney: 1998, *Decomposition methods for water resources optimization models with fixed costs*. Advances in Water Resources 21(4), 261324.

[61] Willis, R. (1976). *Optimal groundwater quality management: Well injection of waste water*, Water Resour.Res., Vol.12, No.1, pp.47-53.

[62] Willis,R. (1979). *A planning model for the management of groundwater quanlity*, Water Resour. Res. Vol.15, No.6, 1305-1312.

[63] Zheng, C., 1990, *MT3D: A modular three-dimensional transport model for simulation of advection, dispersion, and chemical reactions in groundwater systems*, Report to the U.S. Environmental Protection Agency, Ada, OK, 170p.

[64] Zheng, C. and Wang, P.P., 1998, *MT3DMS, A modular three-dimensional multispecies transport model for simulation of advection, dispersion and chemical reactions of contaminants in groundwater systems*, Vicksburg, Miss., Waterways Experiment Station, U.S. Army Corps of Engineers. 238p.

[65] Zheng, C., and P.P. Wang. 2002a. MGOA Modular Groundwater Optimizer Incorporating MODFLOW/MT3DMS, Documentation and Users Guide. Draft. University of Alabama, Tuscaloosa, Alabama.

CHAPTER 5

PARALLEL CALIBRATION OF COMPUTATIONALLY EXPENSIVE WATERSHED MODEL WITH APPLICATION TO CANNONSVILLE WATERSHED

5.1 Introduction

Our increasing understanding of natural systems enables us to implement larger scaled complex mathematical and numerical models. This increasing complexity of these numerical models comes at the cost of increase in the number of effective physical and/or conceptual model parameters. Not all of these parameters can be determined by laboratory experiments and often the laboratory estimated parameters don't work well in field scale model. These model parameters then need to be adjusted so that model predictions closely replicate the observed environmental system response (field measurements). This process of model parameter-adjustment to the observed data is called calibration. The traditional approach i.e. manual calibration by trial-and-error can be extremely labor intensive and difficult/sometimes even impossible to implement for complex model calibration situations where models are calibrated to large measured system over long time durations.

Automatic calibration involves the use of an optimization algorithm to search through the possible values of these parameters to obtain a set of respective parameters based on a specified goodness-of-fit. A typical goodness of fit measure (objective function) is the sum of squared errors (SSE). These kinds of problems are also sometimes referred to as "inverse problems". In general any kind of optimization process involves running the simulation model

many many times, so for cases where one such simulation is computationally expensive (i.e. long computational time) the whole process becomes very expensive, sometimes the whole process may run for weeks or months. Gupta et. al (1999) and Singh and Woolhiser (2002) list three factors influencing the whole calibration process: calibration data, objective function formulation and the optimization algorithm. This study focuses on the third factor by implementing a parallel optimization algorithm for computationally expensive calibration problems. Parallel algorithms can significantly reduce the computational burden for a calibration problem. This study will focus on the automatic calibration of one particular type of environmental models i.e. watershed simulation models however the results are relevant to all types of environmental simulations.

Watershed calibration problem tends to have multiple local minima so they require global optimization methods. Shoemaker et al. (2007) demonstrate the necessity of global optimization methods for watershed models by showing that local optimization methods fail to find the best solutions. A variety of global optimization methods have been used for watershed calibration, including genetic algorithms (e.g. Franchini, 1996; Franchini et al., 1998; Wang, 1997), Adaptive Cluster Covering (ACCO) (Solomatine, 1998; Solomatine et al., 2000), and Shuffled Complex Evolution (SCE) (Duan et al., 1993). Most of these algorithms were based on large number of function evaluations and hence have high computational cost for computationally expensive complex hydrological models.

This study investigates the effectiveness of new parallel algorithm, Radii based Dynamically Dimensioned Search (RODDS) for watershed calibration. These automatic calibration methods will be applied to a SWAT model of single sub-basin and multiple sub-basin models of the Cannonsville Reservoir wa-

tershed in the Catskills region of New York, USA. The RODDS algorithm is designed to be a scalable parallel algorithm that minimizes the computational expense of an optimization problem (one simulation is computationally expensive) to locate good optimal solution points. The algorithm is inspired from serial algorithm DDS (Tolson and Shoemaker, 2007). Tolson and Shoemaker (2007) in their study compare the performance of serial algorithm DDS with SCE, so this study focuses on comparison of RODDS results against the results obtained by serial DDS. This study focuses on comparison the performance of parallel RODDS algorithm with the serial DDS and straightforward parallel implementation of DDS (which is different from RODDS). Details of the algorithm are given in Singh, 2011.

5.2 Model Description

RODDS algorithm is tested on two watershed models for Cannonsville Reservoir in Upstate New York. Tolson & Shoemaker (2007) applied a modified version of the Soil and Water Assessment Tool version 2000 (Neitsch et al., 2001) watershed simulation model to predict flow, sediment and phosphorus for the above-mentioned watershed. The Soil and Water Assessment Tool (SWAT) is maintained by USDA Agricultural Research Service at the Grassland, Soil and Water Research Laboratory in Temple, Texas, USA.

The SWAT simulation model is used to predict the impact of land management practices on water, sediment and agricultural chemical yields in large complex watersheds with varying soils, land use and management conditions over long periods of time. Benaman et al. (2005) and Benaman (2003) used the SWAT

model to simulate for flow and sediment in the Cannonsville watershed. This model was then extended to include particulate and dissolved phosphorous by Tolson & Shoemaker (2007). In this study a parallel algorithm RODDS is used for solving the automatic calibration problems and the results are then compared with the serial algorithm, DDS (Tolson and Shoemaker, 2007).

The Cannonsville Reservoir is one of the main sources of drinking water to New York City. It is located in Delaware County in the Catskill region of Upstate New York. The Watershed is approximately 1200 km^2 in area, most of which is dominated by forests and agricultural lands. Agricultural practices in the watershed are monitored for controlling the phosphorous loading to the reservoir i.e. if not monitored it can result in eutrophication. A treatment plant for removal of algae from eutrophication is estimated to cost NY city over US $8 billion. Further details about the SWAT2005 model application to the Cannonsville Reservoir Watershed are provided in Tolson and Shoemaker (2005).

The goal for this study is to implement the parallel algorithm RODDS on a real world calibration problem. For this purpose two scales of SWAT2005 models within the Cannonsville Reservoir Watershed were selected for this study. The smaller single basin Town Brook (37 km^2) watershed is inside the larger multiple sub-basin Cannonsville (1200 km^2) watershed. The New York State Department of Environmental Conservation ($NYSDEC$) provided daily TSS and total phosphorus loads calculated at each monitoring location. The $NYSDEC$ monitoring program is described in Longobucco and Rafferty (1998).

Four calibration problems were extracted from Tolson and Shoemaker (2007) for this study i.e. two formulations each from Townbrook and Cannonsville watershed. First set of formulation calibrates only the flow parameters whereas

the second set simultaneously calibrates flow, sediment, dissolved phosphorous and other parameters respectively. The parameter ranges were based on ranges in the SWAT2005 model documentation (Neitsch et al., 2005) to replicate the calibration process for the selected study area. Following subsections describe the formulations used for automatic calibration.

5.2.1 Optimization Formulation

For automatic calibration, a calibration problem is formulated as a box-constrained minimization problem, where objective function is some measure of error in calibration. For this study the objective was to calibrate the SWAT2005 model for input against real measured data. The algorithm comparison was done for four different scenarios i.e. Formulation-1 (described below) with two scenarios (Scenario 1-1 for Townbrook and Scenario 1-2 for Cannonsville) and Formulation-2 (described below) with two scenarios (Scenario 2-1 for Townbrook and Scenario 2-2 for Cannonsville).

Formulation-1: Flow Calibration (Scenario 1-1 and Scenario 1-2)

For single basin Townbrook (Scenario 1-1) and multi basin Cannonsville watersheds (Scenario 1-2), SWAT2005 models were calibrated respectively for flow against real measured flow data using the optimization model:-

$$\min_{x} SSE^{Q}(x) = \sum_{t=1}^{T} (Qmeas_t - Qsim_t(x))^2 \tag{5.1}$$

subject to

$$x_i^{min} \leq x_i \leq x_i^{max}, \qquad i = 1, ..., D \tag{5.2}$$

where SSE^Q is the sum of squared error for daily flows, x is a vector of D model parameters that are each subject to bound constraints, $Qmeas_t$ and $Qsim_t$ are the measured and simulated flows (model output) on day t, and T is the total number of days in the calibration period. D is the number of parameters (15 for flow calibration for scenarios 1-1 and 1-2). For Townbrook (Scenario 1-1) in this T is 2192 days (October 1998-September 2004) and for Cannonville (Scenario 1-2), T is 2192 days (January 1994- December 1999). Formulations are discussed further in Tolson and Shoemaker (2007). Table 5.1 lists the calibrated parameters with the respective description and bounds.

Table 5.1: SWAT2005 flow related parameters used in Formulation 1 and Formulation 2

Parameter No. (i)	Parameter Name	Brief Description	Lower Bound (x_i^{min})	Upper Bound (x_i^{max})
1.	$SFTMP$	Snow fall temperature (^{o}C)	-5	5
2.	$SMTMP$	Snow melt base temperature threshold (^{o}C)	-5	5
3.	$SMFMX$	Melt factor for snow (mm H_2O/ C-day)	1.5	8
4.	$TIMP$	Snow pack temperature lag factor	0.01	1
5.	$SURLAG$	Surface runoff lag coefficient	1	24

Table 5.1: (continued)

Parameter No. (i)	Parameter Name	Brief Description	Lower Bound (x_i^{min})	Upper Bound (x_i^{max})
6.	GW_DELAY	Groundwater delay time (days)	0.001	500
7.	$ALPHA\_BF$	Baseflow alpha factor	0.001	1
8.	$GWQMN$	Threshold groundwater depth for return flow (mm)	0.001	500
9.	LAT_TTIME	Lateral flow travel time (days)	0.001	180
10.	$ESCO$	Soil evaporation compensation factor	0.01	1
11.	$CN2^a$	Runoff Curve Number multiplicative factor	0.75	1.25
12.	$DepthT^b$	Soil profile total depth range factor	0	1
13.	BD^b	Moist bulk Density factor	0	1
14.	AWC^b	Available water capacity range factor	0	1
15.	$Ksat^b$	Saturated hydraulic conductivity range factor	0	1

[a] $CN2$ is a multiplicative factor used to simultaneously adjust all spatially

variable base runoff curve numbers ($CN2$) up to a maximum of 98.0

^{b} $DepthT$, BD^{b}, AWC ^{b} and $Ksat^{b}$ are factors linearly scaling the soil type specific physical properties ($SOIL\_depth$, $SOIL\_bd$, $SOIL\_Kdat$ and $SOIL\_AWC$) between their minimum (factor=0) and maximum (factor=1) values as reported in Tolson and Shoemaker (2007).

Formulation-2: Simultaneous Flow, Sediment and Phosphorous Calibration

For this formulation single basin Townbrook and multi basin Cannonville watershed SWAT2005 models were calibrated respectively, (simultaneously) for flow, sediment and total phosphorous against real daily flow and water quality loading data. For this formulation to take into account the difference in magnitudes of flow, sediment and total phosphorous measurements the objective function defined in equation 5.1 needs to be modified or normalized. In this study the calibration is done using the following optimization formulation:-

$$\begin{aligned}\min_{x} E_x \quad = \quad & \frac{(\sum_{t=1}^{T}(Qmeas_t - Qsim_t(x))^2)^{1/2}}{std(Qmeas_t)} \\ & + \frac{(\sum_{t=1}^{T}(SSmeas_t - SSsim_t(x))^2)^{1/2}}{std(SSmeas_t)} \\ & + \frac{(\sum_{t=1}^{T}(MPmeas_t - SSsim_t(x))^2)^{1/2}}{std(MPmeas_t)} \\ & + \frac{(\sum_{t=1}^{T}(OPmeas_t - PPsim_t(x))^2)^{1/2}}{std(OPmeas_t)} \end{aligned} \tag{5.3}$$

subject to

$$x_i^{min} \leq x_i \leq x_i^{max}, \qquad i = 1, ..., D \tag{5.4}$$

where E_x is the normalized sum of squared error for daily flows, sediments and total phosphorous. x is a vector of D model parameters that are each subject to bound constraints. $Qmeas_t$ and $Qsim_t(x)$ are the measured and simulated flows

(model output) on day t. $SSmeas_t$ and $SSsim_t(x)$ are the measured and simulated suspended sediments (model output) on day t. $MPmeas_t$ and $MPsim_t(x)$ are the measured and simulated mineral phosphorous (model output) on day t. $OPmeas_t$ and $OPsim_t(x)$ are the measured and simulated organic phosphorous (model output) on day t. T is the total number of days in the calibration period. D is the number of parameters (32 for simultaneous flow, sediment and phosphorous calibration i.e. for Scenarios 2-1 and 2-2). For Townbrook (Scenarios 2-1) in this formulation, T is 2192 days (October 1998-September 2004), and for Cannonville (Scenarios 2-2), T is 2192 days with a different calibration period (January 1994- December 1999).

Table 5.1 lists the parameters used to calibrate the model to flow (Formulation 1) with the respective description and bounds. Additional phosphorous and sediment parameters and bounds used only in Formulation-2 are listed in Table 5.2.

Table 5.2: SWAT2005 Additional Sediment and Phosphorous related parameters in Formulation 2 but not in Formulation-1

Parameter No. (i)	Parameter Name	Brief Description	Lower Bound (x_i^{min})	Upper Bound (x_i^{max})
1-15.	Table 5.1	Flow parameters as in Formulation 1		
16.	*ADJ_PKR*	Peak rate adjustment factor for sediment routing in *tributary channels*	0.5	1.5

Table 5.2: (continued)

Parameter No. (i)	Parameter Name	Brief Description	Lower Bound (x_i^{min})	Upper Bound (x_i^{max})
17.	PRF	Peak rate adjustment factor for sediment routing in *main channel*	0.5	1.5
18.	$SPCON$	Channel sediment routing parameter (linear)	0.0001	0.001
19.	$SPEXP$	Channel sediment routing parameter(exponential)	1	2
20.	LAT_SED	Sediment Concentration in lateral and groundwater flow (mg/l)	0.1	22.8
21.	$SLSUBBSN\_f^a$	Average slope length (m)	0.5	1.5
22.	$SLSOIL\_f^a$	Slope length for lateral subsurface flow (m)	0.5	1.5
23.	CH_EROD	Channel erodibility factor	0	0.6
24.	$CLAY\_f^b$	Soil layer clay content range factor	0	1
25.	$ROCK\_f^b$	Soil layer rock content range factor	0	1

Table 5.2: (continued)

Parameter No. (i)	Parameter Name	Brief Description	Lower Bound (x_i^{min})	Upper Bound (x_i^{max})
26.	$MUSLE\_adj^c$	Cannonville model specific calibration factor controlling snow cover influence on sediment yield	0	1
27.	$PPERCO$	Phosphorous percolation coefficient (10 m^3/mg)	10	17.5
28.	$PHOSKD$	Phosphorous soil partitioning coefficient (m^3/mg)	100	200
29.	CMN	Rate factor for humus mineralization of active organic Phosphorous	0.0001	0.003
30.	P_UPDIS	Phosphorous uptake distribution factor	0.1	100
31.	$ERORGP$	Phosphorous enrichment ratio for loading with sediment	1	5
32.	PSP	Phosphorous availability index	0.01	0.75

[a] $SLSUBBSN\_f$, $SLSOIL\_f$ and $CN2\_f$ are multiplicative factors used to

simultaneously adjust all spatially variable base values of the $SLSUBBSN$, $SLSOIL$ and $CN2$ parameters, respectively.

[b] $CLAY\_f$ and $ROCK\_f$ are factors linearly scaling the soil type specific physical properties (CLAY and ROCK) between their minimum (factor = 0) and maximum (factor = 1) values as reported in Tolson and Shoemaker (2007). The ranges for CLAY and ROCK were derived from soil survey data.

[c] The $MUSLEadj$ parameter was added to SWAT for the Cannonsville application (Tolson and Shoemaker, 2007) and controls the snow cover influence on Hydrologic Response Unit sediment yield.

5.3 Algorithm Description

Radii based Optimization using Dynamically Dimensioned Search, RODDS (Chapter 2, Singh, 2011) algorithm is a parallel stochastic heuristic global search algorithm that tries to minimize the computational expense of an optimization problem by effectively utilizing the multi-core machines. The algorithm is inspired by but different from serial algorithm DDS (Tolson and Shoemaker, 2007). The RODDS algorithm searches (as in serial DDS) globally at the start of the search and becomes a local search as the number of iterations approaches the maximum allowable number of function evaluations. This transition is achieved by dynamically and probabilistically limiting the dimensional space of the neighborhood as the search progresses.

The candidate points in RODDS are generated by perturbing the current best solution point in randomly selected dimensions. At a particular iteration the

choice of these evaluation points (for all processors) depends on all previously evaluated points and the respective function values i.e. RODDS tries to stay away from all previously generated high-cost points. The RODDS algorithm input parameters involve initial and final radii's, neighborhood perturbation factor, maximum allowed function evaluations, the problem size and the number of processors to be used for the run. The algorithm starts with a set of randomly generated uniformly spaced points. Each processor then evaluates the function and returns the objective function value to the main processor. Then a set of possible candidate points are generated. One of the main features of RODDS algorithm is that points evaluated by the expensive simulation are not allowed to by within the radius length of a point where an expensive simulation has been done. The radius length of this hypersphere decreases with the RODDS iteration number. The radius of this hypersphere depends on iteration number, maximum allowed function evaluations, current best cost value and the function values at all previously generated points (Chapter 2, Singh, 2011). If the candidate point happens to lie within the hypersphere, the whole generation procedure is repeated till this radii criterion is satisfied. Once the candidate solution points are generated, the main processor again assigns the candidate solution points to individual processors and the whole process (expensive objective function evaluation) is repeated until maximum allowed evaluations is exhausted. For more details reader is referred to (Chapter 2, Singh, 2011).

RODDS is designed to require little or no tuning of algorithms. The maximum number of simulations the user is willing to do to obtain a solution is "m". This number is known to the user and is hence not a tuned algorithm parameter. The variance for generating new points is r and we recommend using the default values of $r = 0.2$ (as suggested in Tolson and Shoemaker (2007)). RODDS

has two parameters describing the initial and final radius size. We recommend using the default values of $R_{initial} = 0.3$ and $R_{final} = 0.05$

5.4 Outline of Algorithm comparisons

The goal of this study is to implement a parallel RODDS algorithm to identify good solution points for computationally expensive calibration problems when wall clock time is limited. Thus the experimental runs for the study were designed to test the algorithm with a fixed number of total function evaluations (across all processors) under varying number of processors. So in the experiments as the number of processors increase, the effective wall clock time decreases. The results presented here compare the performance of RODDS with the serial version of DDS and our implementation of parallel DDS (called DDS-PC). The main difference between the RODDS and the DDS-PC is that RODDS uses hyperspheres to stay away from local minima for better exploration of search space. In order to compare the effectiveness of this method of candidate point generation, the plots include a version of DDS-PC as well as serial DDS. Tolson and Shoemaker (2007) in their DDS paper compared the performance of DDS algorithm with SCE and showed it to outperform SCE when number of simulations are limited. Hence this study focuses on comparison of new Parallel algorithm with the serial one.

Results for the study are presented in three sections and the last section then discusses the quality of results. Section 5.5 compares the performance of RODDS algorithm with the serial version of DDS and the DDS-PC, for watershed calibration test problems. Section 5.6 discusses the performance compari-

son with increase in the number of processors used for that respective run (i.e. decrease in wall clock time). Section 5.7 discusses and compares the results in terms of parallel computing metrics i.e. speedup and efficiency. Section 5.8 summarizes and discusses the quality of results obtained for the tested formulations.

5.5 Algorithm Performance Comparison

The first set of plots summarize the results of optimization run for all the Scenarios 1-1, 1-2, 2-1 and 2-2. The figures 5.1 and 5.2 compare the performance of RODDS algorithm with the serial DDS (Tolson and Shoemaker, 2007), DDS-PC and RODDS 1-processor for the Townbrook problem for formulation 1 and formulation 2 respectively. Optimization runs and comparisons were done with 4, 8, 16 and 32 processors (thus 4, 8, 16 and 32 times computational time savings) respectively. Plots show the convergence plot for the objective function values with varying number of processors used for the respective runs, with the maximum allowed function evaluations being limited to 400 for the 15 dimensional formulation-1 and 1600 for the 32 dimensional formulation-2. For each figure, the function value (y -axis) is plotted against the specific i^{th} function evaluation (x- axis). This function value is the averaged over 30 trial runs for the best solution found on or before the i^{th} function evaluation, respectively. For RODDS algorithm the one iteration equals 'w' function evaluations, where 'w' is the number of processors used for the run. Similarly figures 5.3 and 5.4 compare the performance of the tested algorithms on Cannonsville function.

Figure 5.1(a) and 5.3(a) show that RODDS with 4 processors performs pretty

well in comparison to serial DDS. In general RODDS performed better than the serial DDS with processors up to 16. In case of 32 processors the quality of results for same number (as in serial) function evaluations is not as good as the serial RODDS solution. Similarly sections 5.1(b), 5.1(c) and 5.1(d) compare the algorithm performance with 8, 16 and 32 processors respectively. The difference in the average best solutions obtained is much more clear in case with processors 4 and 8 (i.e. the results were better with 1/4th or 1/8th the serial computational time).

5.6 Processor Performance Comparison

Figures 5.5 and 5.6 summarizes the performance of RODDS with respect to the number of processors used for the run. Each figure plots the results for the different test functions with respect to the respective number of processors used for that particular run. The maximum allowed function evaluations per processor were chosen such that each run does same amount of total function evaluations in total. These figures show how the performance of RODDS is affected by increase in number of processors. Results (figures 5.5 and figures 5.6) here show that the idea of hyperspheres help RODDS to maintain good efficiency in terms of quality of results as the number of processors used in a particular run is increased. These plots show that as the number of processor increases, RODDS maintains good efficiency in terms of objective function value.

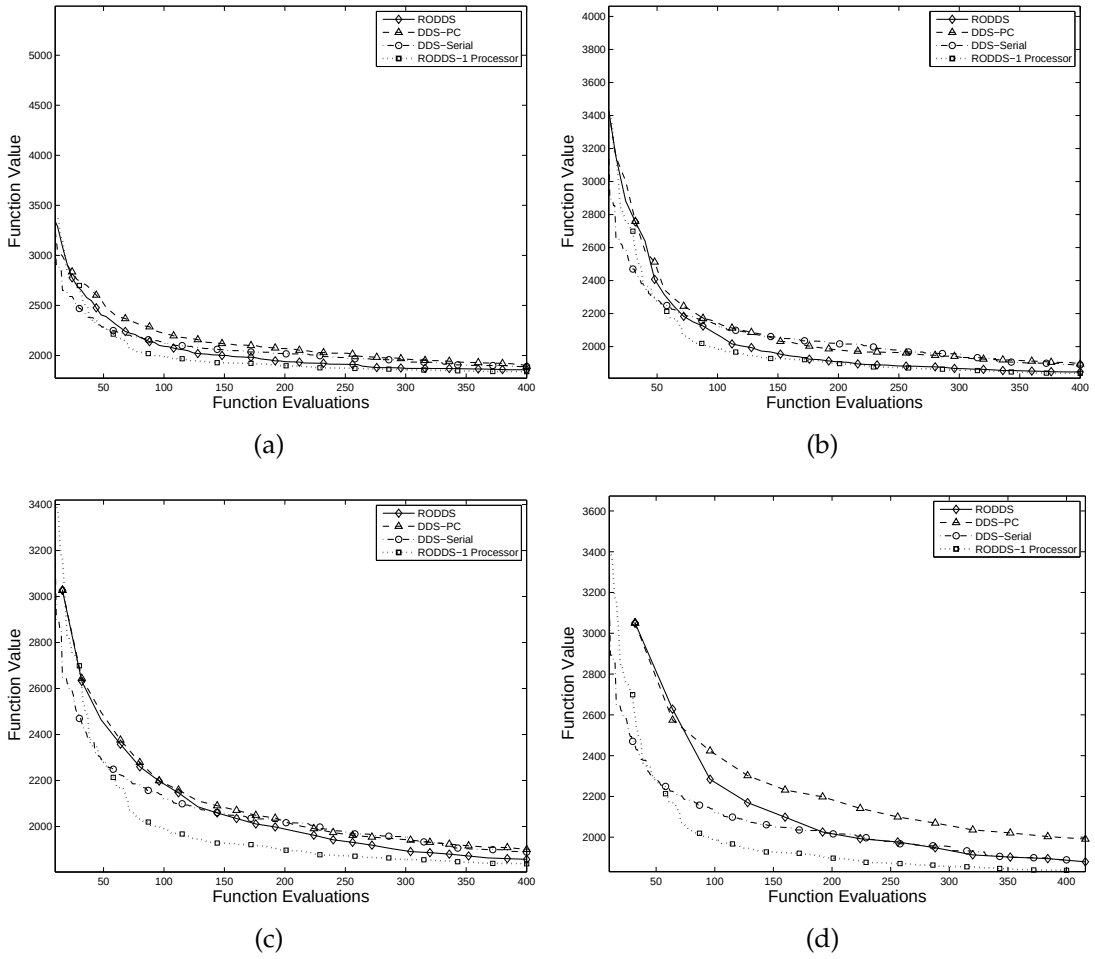


Figure 5.1: Results for Formulation-1 on Townbrook model: (a) with 4 Processors (Wall clock time 16 mins); (b) with 8 Processors (Wall clock time 9 mins); (c) with 16 Processors (Wall clock time 7 mins); and, (d) with 32 Processors (Wall clock time 5 mins). In all cases Wall clock time for serial DDS and RODDS-1 processor is 60 mins

5.7 Metrics of Parallel Performance

The most commonly used parallel implementation metrics are speedup and efficiency. Speedup is a measure of time gained i.e. by how much a parallel algorithm is faster than the respective serial algorithm. Efficiency metric reflects the processor utilization i.e. how well the work is distributed among the pro-

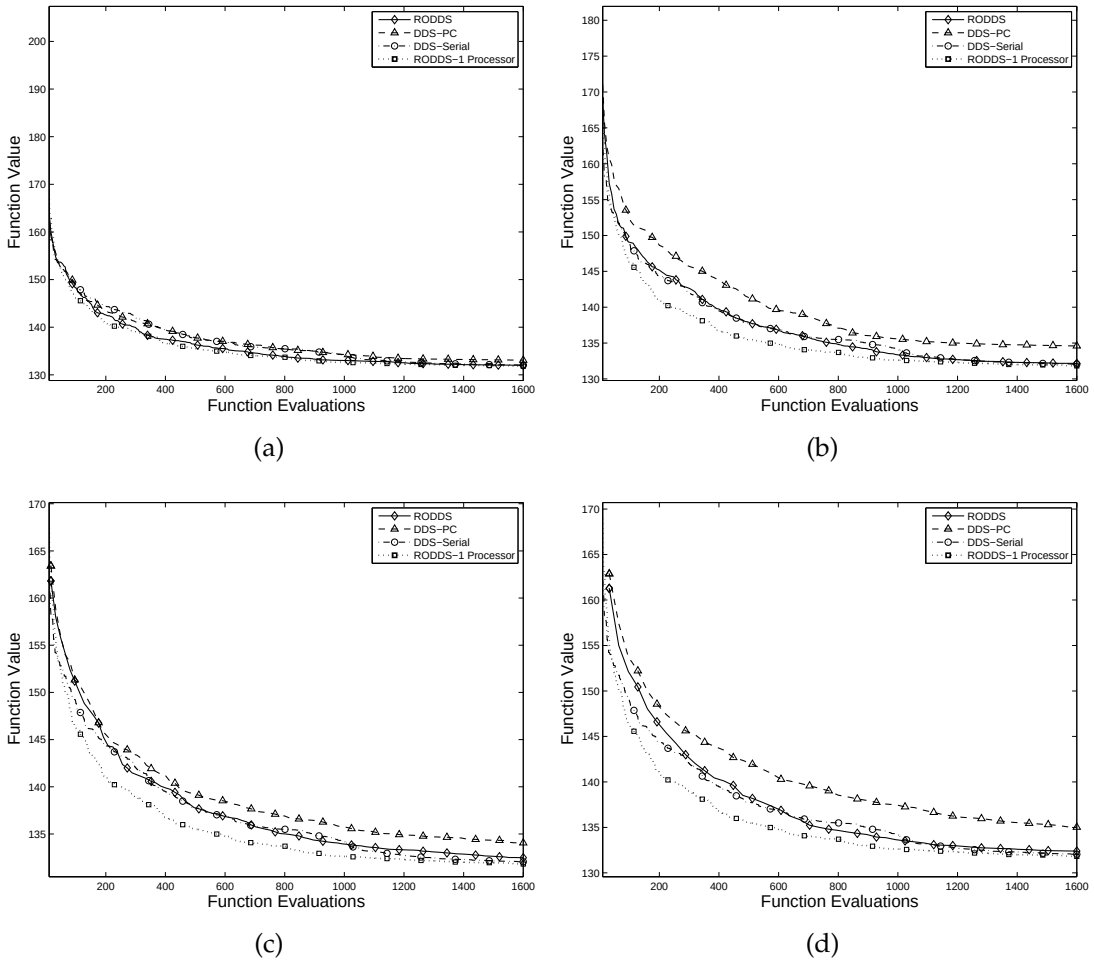


Figure 5.2: Results for Formulation-2 on Townbrook model: (a) with 4 Processors (Wall clock time 70 mins); (b) with 8 Processors (Wall clock time 36 mins); (c) with 16 Processors (Wall clock time 19 mins); and, (d) with 32 Processors (Wall clock time 10 mins). In all cases Wall clock time for serial DDS and RODDS-1 processor is 266 mins

cessors. We defined these criteria's based on the runs it took for an algorithm to reach within 1% of the final answer obtained by DDS-serial averaged over 10 trials for the Townbrook model (Scenarios 1-1 and 2-1), Cannonsville (Scenarios 1-2) and 5 trials for the Cannonsville (2-2).

Tables 5.3 to 5.6 list the parallel metrics for the watershed functions for RODDS and DDS-PC, respectively. Figure 5.7 explains the relation between

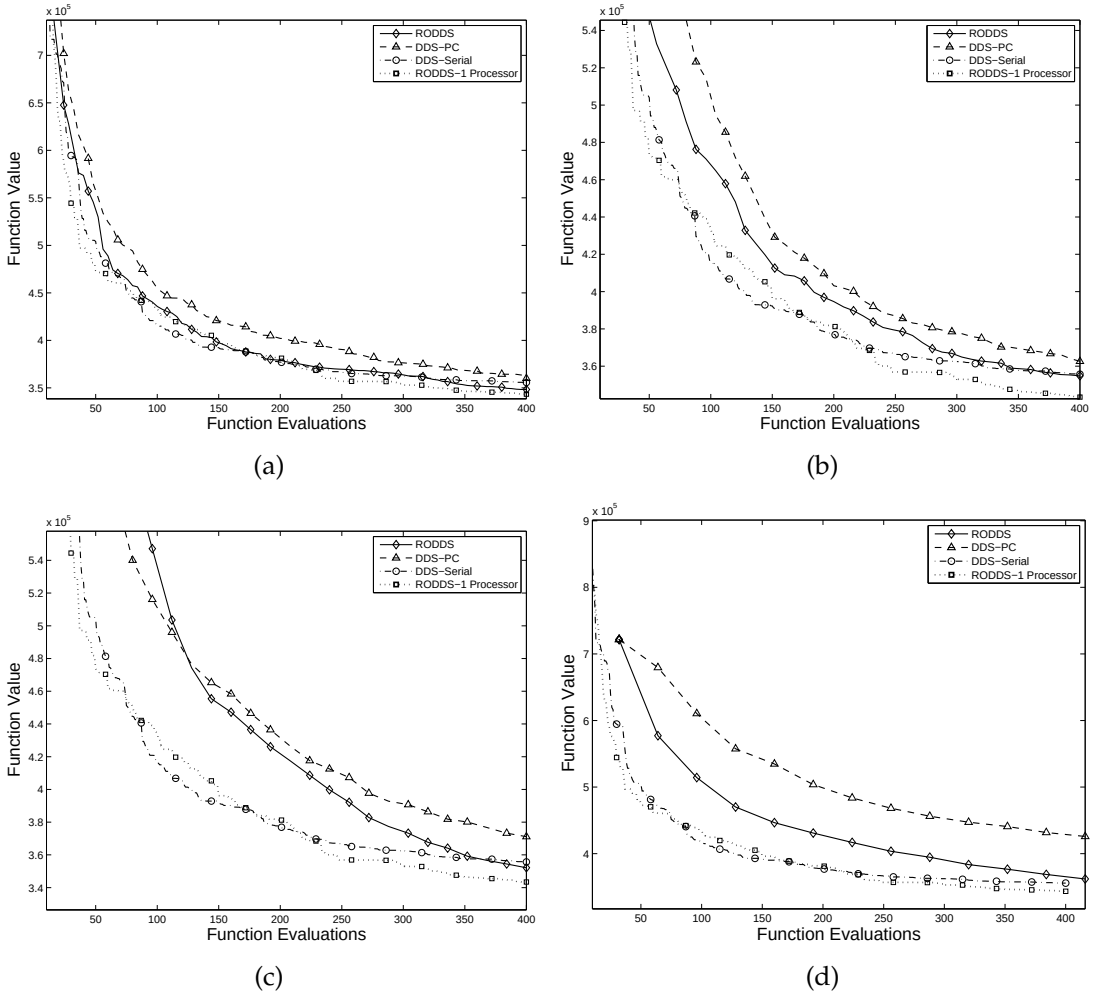


Figure 5.3: Results for Formulation-1 on Cannonsville model: (a) with 4 Processors (Wall clock time 204 mins); (b) with 8 Processors (Wall clock time 104 mins); (c) with 16 Processors (Wall clock time 54 mins); and, (d) with 32 Processors (Wall clock time 34 mins). In all cases Wall clock time for serial DDS and RODDS-1 processor is 800 mins

wall clock times and modified metrics and the following tables for respective functions list the modified metrics. These tables use the following definitions:

- np is the number of processors used for a particular run.
- D_f is the difference between the results of RODDS and DDS-serial at the end of optimization run averaged over 30 trials.

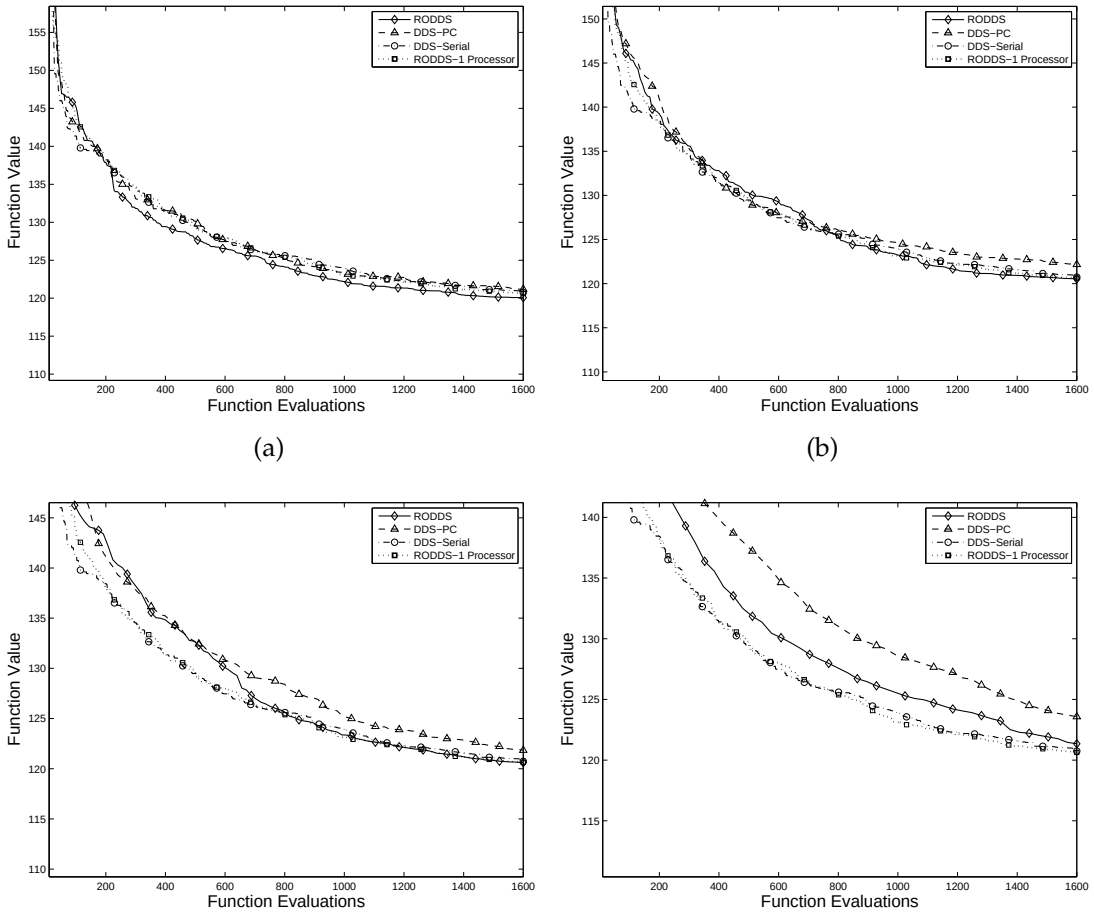
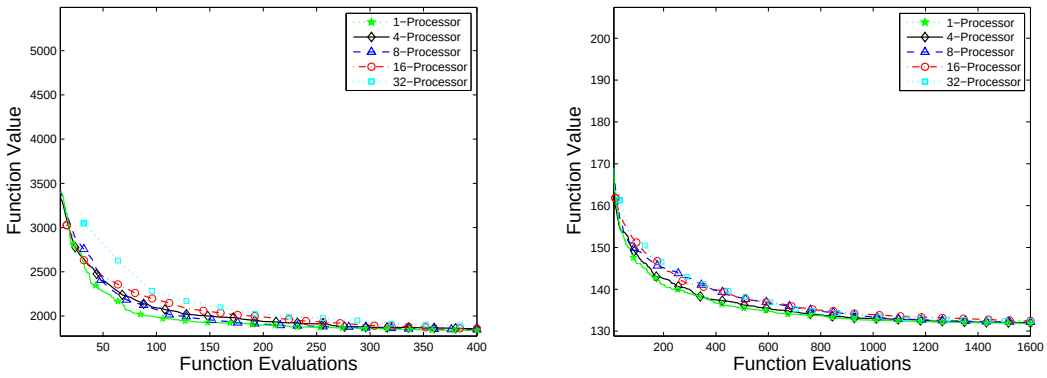


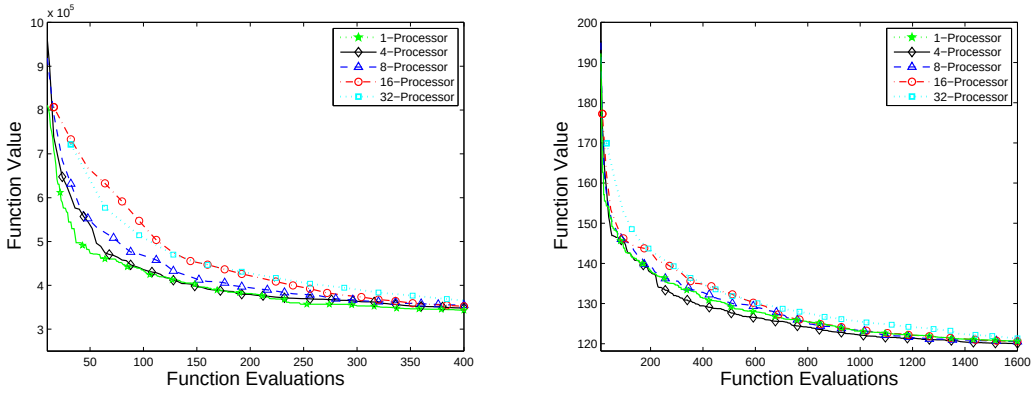
Figure 5.4: Results for Formulation-2 on Cannonsville model: (a) with 4 Processors (Wall clock time 13.5 hours); (b) with 8 Processors (Wall clock time 6.75 hours); (c) with 16 Processors (Wall clock time 3.4 hours); and, (d) with 32 Processors (Wall clock time 1.8 hours). In all cases Wall clock time for serial DDS and RODDS-1 processor is 54 hrs

- WT_f is the wall clock units to get to Serial answer.
- T_f is the total CPU units to get to Serial answer i.e. $nprocs$*WT_p.
- $T_f(c\%)$ is the wall clock units to reach within c% of the serial answer, for a respective algorithm.
- Speedup $Sp-1\%$ is the ratio of 'serial run to get to within c% of final serial answer' to 'total CPU units to get to serial answer' i.e. $\frac{T_s}{T_p}$, averaged over



(a) (b)

Figure 5.5: Processor performance comparison on Townbrook model: (a) Formulation-1; (b) Formulation-2;



(a) (b)

Figure 5.6: Processor performance comparison on Cannonsville model: (a) Formulation-1; (b) Formulation-2;

30 trials.

- Efficiency $Ef - 1\%$ is the ratio of $'Sp - 1\%'$ to the respective number of processors used.

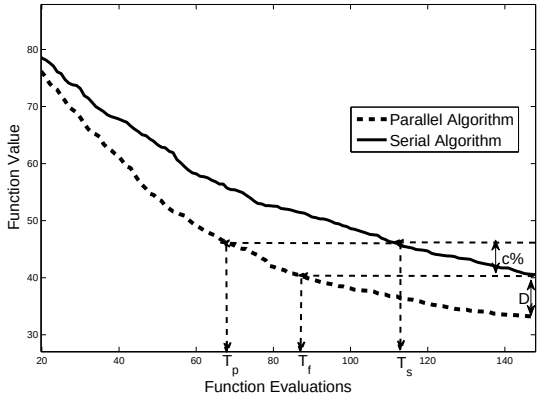


Figure 5.7: RODDS Metric Definitions

Table 5.3: RODDS Results for Formulation-1 on Townbrook. a) is for RODDS and b) is for DDS-PC

np	D_f	WT_f	T_f	$T_p(1\%)$	$Sp(1\%)$	$Ef(1\%)$
1	2.8	227	227			
4	1.7	66	264	65	5.2	1.3
8	2.3	29	232	25	13.6	1.7
16	1.6	20	320	17	19.9	1.2
32	0.5	12	384	10	33.9	1.06

a) RODDS

np	D_f	$T_s(1\%)$	$Sp(1\%)$	$Ef(1\%)$
1	0	339		
4	-0.1	99	3.4	0.85
8	-0.3	46	7.4	0.93
16	-0.6	25	13.56	0.85
32	-5.4			

b) DDS-PC

Table 5.4: RODDS Results for Formulation-2 on Townbrook. a) is for RODDS and b) is for DDS-PC

np	D_f	WT_f	T_f	$T_p(2\%)$	$Sp(2\%)$	$Ef(2\%)$
1	0.2	1355	1355			
4	0	382	1528	170	5.6	1.4
8	0.08	200	1600	101	9.4	1.18
16	-0.3	100	1600	54	17.6	1.1
32	-0.2	-	-	25	38	1.18

a) RODDS

np	D_f	$T_s(1\%)$	$Sp(2\%)$	$Ef(2\%)$
1	0	950		
4	-0.8	240	3.9	0.97
8	-1.9	184	5.16	0.64
16	-2	82	11.58	0.73
32	-2.3			

b) DDS-PC

Table 5.5: RODDS Results for Formulation-1 on Cannonsville. a) is for RODDS and b) is for DDS-PC

np	D_f	WT_f	T_f	T_p(2%)	Sp(2%)	Ef(2%)
1	3.4	294	294			
4	2.1	84	336	75	3.8	0.95
8	0.2	47	376	39	7.4	0.92
16	0.1	23	368	21	18.5	1.1
32	-1.7	-	-	13	22	0.69

a) RODDS

np	D_f	T_s(1%)	Sp(1%)	Ef(1%)
1	0	290		
4	-1.2	99	2.9	0.72
8	-1.2	49	5.9	0.73
16	-4.3	-	-	-
32	-2.4	-	-	-

b) DDS-PC

Table 5.6: RODDS Results for Formulation-2 on Cannonsville. a) is for RODDS and b) is for DDS-PC

np	D_f	WT_f	T_f	T_p(2%)	Sp(2%)	Ef(2%)
1	0.1	1557	1557			
4	0.5	342	1368	219	4.9	1.23
8	0.1	184	1472	126	8.5	1.06
16	0.1	96	1536	64	16.75	1.05
32	-0.5	-	-	42	25.52	0.8

a) RODDS

np	D_f	T_s(1%)	Sp(1%)	Ef(1%)
1	0	1072		
4	-0.4	256	4.18	1.04
8	-1.2	155	6.9	0.86
16	-0.9	80	13.4	0.84
32	-2.4			

b) DDS-PC

5.8 Discussion

This section is divided in two parts. The first section discusses the algorithm performance and second one briefly discusses the quality of results obtained. Numerical results from the optimization runs (figures 5.1-5.4) demonstrate that the RODDS algorithm is able to efficiently (computationally) use parallelism to get good results for the Townbrook and Cannonsville watershed calibration problems. Plots 5.1(a) and 5.3(a) show that RODDS outperforms DDS-serial as well DDS-PC, for 1/4th saving in time. Similar results are obtained for 1/8th and 1/16th time savings. For 1/32nd time savings it gets to answers very close to serial values. Cannonsville plots (5.5 and 5.6) also support the above-said statement. Tables 5.3 to 5.5 lists the modified speedups and efficiencies for the two calibration models. RODDS algorithm in general was able to reach efficiencies of greater than one also all of these efficiencies were much better as compared to DDS-PC (a parallel implementation for DDS), which show that the idea of hyperspheres helped RODDS to escape some local minima points and get to better solution point then DDS-PC.

The fact that RODDS is able to get better solution points than even serial DDS is remarkable since this results in a speed up that is greater than the number of processors and hence an efficiency greater than one. DDS-PC in general was able to locate few good points but also some bad ones. Shoemaker et. al. (2007) showed the multimodal shape of the SSE surface for the Townbrook SWAT model. The optimization method has to search over this surface with many local minima which promotes the need for an global optimization method which can avoid getting trapped in local minima. RODDS tries to use the idea of radii's (hyperspheres) to avoid or escape from local minima (if caught in one) in paral-

lel. The idea of using radii's for candidate point generation is to stay away from poor objective function value points as well as all previously calculated points. One important factor is that there is no restriction around the current best point. As the search progresses, the radius shrinks so that algorithm becomes a local optimization algorithm at the end of the run.

It is very noticeable that the efficiency E_f is greater than 1 for most applications of RODDS in Tables 5.3-5.5. This is very significant since implementation of parallel processing usually result in efficiency well below 1. Tables also show one interesting behavior to notice in RODDS results, for some runs efficiencies with 8 and 16 processors is better than 4 processors which is not expected since the parallel efficiency usually decreases with the number of processors. For example in Table 5.3 a) for row 3 (=8 processors) E_f is 1.7 and for row 2 (=4 processors) E_f is 1.3. Authors suggest this is because the "best solution" used in the next iteration is the best among the psimulations tested in one iterations. With serial DDS the next solution is only compared with the previous best solution. Thus it is more adaptive to avoid local minima as serial DDS moves only in one direction at a particular iteration.

In general for Scenario 2-2 i.e. simultaneous Flow, Sediment, Dissolved phosphorous and Organic phosphorous calibration for Cannonsville watershed (2192 observations), RODDS with 32 processors on average obtained correlation coefficients of 0.72, 0.65, 0.59 and 0.75 with a percentage bias of 4, 18, 23 and 38 respectively. However these values depend more the objective function formulations. This study focuses on implementation of RODDS on calibration so experimenting with the objective function formulations is beyond the scope of this paper.

5.9 Conclusion

For the two tested real watershed calibration problems, numerical results demonstrate that RODDS algorithm effectively used the parallel processing for automatic calibration. In general RODDS was able to locate solution points at least as good as the serial algorithm DDS but the wall clock time for RODDS was usually less than $1/P$ as long as the serial time (where P is the number of processors), which occurs when the efficiency E_f is greater than 1. The value of parallel algorithms in general (RODDS in this study) is greatest for computationally demanding models where there's limited time to get results. For example in this study where the objective function or model evaluations were limited, RODDS with 16 processors was able to locate as good solution point in 3.40 hours as were obtained by serial algorithm in 54 hours. RODDS algorithm like DDS is quite simple and thus can be easily coded in whatever programming language of choice. For the current study the algorithm has been implemented in MATLAB. RODDS is an attractive optimization tool for calibration of Watershed and other environmental simulation models because of increasing availability of inexpensive multi-core machines and the simple characteristics of RODDS algorithm.

BIBLIOGRAPHY

[1] Benaman, J. (2003) *Uncertainty and sensitivity analyses for watershed models: hydrology and sediment transport modeling on the Cannonsville Reservoir System*. PhD Thesis, School of Civil and Environmental, Cornell University, Ithaca, New York, USA.

[2] Benaman, J., Shoemaker, C. A. & Haith, D. A. (2005) *Modeling non-point source pollution using a distributed watershed model for the Cannonsville Reservoir Basin*, Delaware County, New York. ASCE J. Hydrol. Engng 10(5), 363374.

[3] Duan, Q. , Gupta, V. K. and Sorooshian, S. (1993) *Shuffled complex evolution approach for effective and efficient global minimization*. J. Optimization Theory and Applications 76:3 , pp. 501-521.

[4] Franchini, M., 1996., *Use of a genetic algorithm combined with a local search method for the automatic calibration of conceptual rainfall-runoff models* Hydrological Sciences Journal-Journal Des Sciences Hydrologiques, 41(1): 21-39.

[5] Franchini, M., Galeati, G. and Berra, S., 1998., *Global optimization techniques for the calibration of conceptual rainfall-runoff models*. Hydrological Sciences Journal-Journal Des Sciences Hydrologiques, 43(3): 443-458.

[6] Gupta, H.V., Bastidas, L.A., Sorooshian, S., Shuttleworth, W.J. and Yang, Z.L., 1999. *Parameter estimation of a land surface scheme using multicriteria methods*. Journal Of Geophysical Research-Atmospheres, 104(D16): 19491-19503.

[7] Neitsch, S.L., Arnold, J.G., Kiniry, J.R. and Williams, J.R. *Soil and Water Assessment Tool Theoretical Documentation Version 2005*, US Department of Agriculture - Agricultural Research Service, Temple, Texas.

[8] Neitsch, S.L., Arnold, J.G., Kiniry, J.R. and Williams, J.R. *Soil and Water Assessment Tool User's Manual Version 2005*, US Department of Agriculture - Agricultural Research Service, Temple, Texas.

[9] Shoemaker, Christine A., Regis Rommel G.,Fleming Ryan C.,*Watershed calibration using multistart local optimization and evolutionary optimization with radial basis function approximation*. Hydrological Sciences Journal, 2007.

[10] Singh Amandeep, PhD thesis *Efficient Optimization of Computationally Expensive Problems using a New Parallel Algorithm and Response Surface Based Methods* Cornell University, 2011.

[11] Singh, V.P. and Woolhiser, D.A., 2002. *Mathematical modeling of watershed hydrology*. Journal Of Hydrologic Engineering, 7(4): 270-292.

[12] Solomatine, D.P. *Genetic and other global optimization algorithms comparison and use in calibration problems*. Proc., 3rd Int. Conf. on Hydroinformatics, 1021-1027,1998.

[13] Solomatine D., Dibike Y.B. and Kukuric, N. 2000. *Automatic calibration of groundwater models using global optimization techniques*, In the Journal of Hydrological Sciences, Vol. 44, No. 6, pp. 879-893.

[14] Tolson, B. A., and C. A. Shoemaker (2007), *Dynamically dimensioned search algorithm for computationally efficient watershed model calibration*, Water Resour. Res., 43, W01413, doi:10.1029/2005WR004723. Jan. 2007

[15] Tolson, B. A. (2005) *Automatic Calibration, Management and Uncertainty Analysis: Phosphorous transport in the Cannonsville Watershed*. PhD Thesis, School of Civil and Environmental, Cornell University, Ithaca, New York,USA.

[16] Tolson, B.A. and Shoemaker, C.A., 2005. *Cannonsville Reservoir Watershed SWAT2000 model development, calibration and validation*, Journal of Hydrology Volume 337, Issues 1-2, 15 April 2007, Pages 68-86.

[17] The Mathworks, Inc. (2009a) *Genetic Algorithm and Direct Search Toolbox for Use with MATLAB: Users Guide*, version 1. The Mathworks, Inc., Natick, Massachusetts, USA.

[18] The Mathworks, Inc. (2009a) *Optimization Toolbox for Use with MATLAB: User's Guide*, version 3. The Mathworks, Inc. Natick, Massachusetts, USA

[19] Wang, Q.J., 1997. *Using genetic algorithms to optimise model parameters*. Environmental Modeling & Software, 12(1): 27-34.

CHAPTER 6

CONCLUSION

This dissertation focuses on development and implementation of computationally efficient optimization algorithms for groundwater management and calibration of computationally expensive models. A new parallel algorithm, RODDS was developed and implemented on groundwater management and watershed calibration models. A new methodology SIT-RBF is developed in an attempt to minimize the computational expense of fixed cost problems (mixed integer problems) by implementing a sequential response surface based method.

RODDS is a new parallel optimization algorithm developed to find near optimal solution points for global optimization problems within fixed computational budget (with reduced total wall-clock time). RODDS algorithm implements hyperspheres to efficiently explore the search space in parallel. It was demonstrated through five test functions, a real groundwater contamination transport model and two real watersheds that for a given computational budget RODDS consistently identified near optimal solution points as obtained by serial algorithm for all functions. Numerical results show that RODDS algorithm was able to reach efficiencies for up to 32 processors greater than one as compared to Serial DDS for all tested functions. This is an excellent result since parallel efficiencies are typically much lower than 1 (e.g. 0.5) as the number of processors increase. Results also show the efficiency of hyperspheres used in RODDS candidate point generation by comparing the results with parallel implementation without the hyperspheres e.g. for the groundwater remediation problem with 16 processors RODDS achieved an efficiency of 1.1 as compared to

efficiency of 0.55 by the parallel implementation without the hyperspheres. The increasing availability of cheap multi-core machines coupled with simple characteristics of RODDS makes it an attractive optimization tool for Environmental simulations. In the future, we hope to better delineate the types of functions on which we expect RODDS to perform well. We also want to test RODDS performance with large number of processors (> 500) and on a parallelized simulation model (using parallel simulation for higher speedups). For example, RODDS with 20 processors using simulations each efficiently running on 25 processors can use 500 processors efficiently. Future studies will also be devoted to further analyzing the sensitivity of the performance relative to the parameter selection.

The results of algorithm comparisons (chapter 3) indicate that response surface based optimization methods can be effective tools in designing the management policy for computationally expensive groundwater models. The performance of four response surface based optimization methods was compared with heuristic and derivative-based methods for two EPA groundwater superfund sites i.e. Umatilla Chemical Depot,Oregon and Blaine Ammunition Depot, Nebraska. The response surface based methods were shown to be robust to different formulations of the objective function and different levels of computational complexity of the groundwater model. In multiple independent optimization trials, the stochastic RBF (Regis and Shoemaker, 2007) had a lower mean with small variance of the objective function values than heuristic and derivative-based optimization methods. For example in case of Umatilla problem Stochastic RBF had solution over an order of magnitude better than conventional methods. In the future, we intend to implement response surface based optimization methods to other application areas in water resources systems (water distribution networks etc.).

A new methodology SIT-RBF, integrating SIT integer optimization with Response surface based method was developed to minimize the computational expense for long term management policy of fixed cost problems in Water Resources. SIT algorithm generates a fixed number of candidate integer configurations from the neighborhood. The suggested SIT-RBF methodology tries to use sequentially the expensive function evaluation information (decision variables with respective function values) from different integer variables configuration to build RBF surfaces for the new configuration. This prior information improves the initial RBF fit for the new configuration thus the RBF approximations, hence reduces the amount of function evaluations to be done to find optimal value of continuous value variables corresponding to the new configuration. The study compared the suggested SIT-RBF methodology with GA based NSGA and a stand-alone mixed integer value optimizer, NOMAD. The results presented indicate that under limited computational budget, the integrated methodology was much more effective than the using these other two methods, which appear to be the most efficient among previous methods for this problem. For the groundwater remediation problem (Umatilla) the SIT-RBF methodology had solution over 150 times better than the conventionally used GA. This methodology can feasibly be used for much larger water resources simulation models than is possible with previously existing methods. In the future, we expect to do more numerical testing of the algorithm to better understand SIT-RBF performance. These numerical studies will also help us determine good default parameters for SIT-RBF. We also want to improve the SIT algorithm by generating dynamic number of candidate points from neighborhood to possibly explore whole neighborhood space. In this context it will also be critical to determine the benefits of using all previous simulation infor-

mation (decision variables with respective function values) versus using only some of the function evaluation information. We believe that using only some of the function value information might improve the performance of SIT-RBF methodology. We also intend to pursue versions of SIT-RBF that can take advantage of parallel computing environments for RBF minimizations (Parallel Stochastic RBF).

New Methods RODDS, SIT-RBF are shown to be very effective on water resources problem in two major sub areas Groundwater Hydrology and Watershed Hydrology. Although this study focused on groundwater management and watershed calibration models, the results are just as relevant to all environmental simulations of a computationally demanding model.

In further research, the suggested methodologies (RODDS and SIT-RBF) should also be extended for much larger scaled water resources problems and to the problems from other application areas using traditional optimization methods.

APPENDIX A

THE RODDS ALGORITHM

This appendix shows the general structure of RODDS algorithm and the way it generates candidate points for expensive objective function evaluation. RODDS is a new parallel optimization algorithm which tries to find near optimal solution points for global optimization problems by efficiently using nowadays commonly available multi-core machines. Figure A.1 shows the flowchart for the algorithm. The algorithm starts with main processor generating initial set of decision variables. This initial set is then passed on to respective processors for simulation run. After simulation run each individual processor passes on the respective objective function value to main processor. The main processor then generates the next set of candidate points for simulation A.1 and the whole process is repeated till maximum allowed function evaluations are exhausted.

A.1 Candidate Point Generation

RODDS algorithm uses hyperspheres in candidate point generation. The algorithm needs to generate more than one candidate points at any step for expensive function evaluation (by all processors). These points are systematically generated to efficiently explore the search space. The idea behind using hyperspheres is to keep the newly generated candidate points away from all previously evaluated expensive solution points. The radius of the hypersphere depends on the input parameters initial and final radius adjusted by a factor. This factor assigns higher weight to solution points with high objective function values. The radii for these hyperspheres decreases as the search progresses. Figure

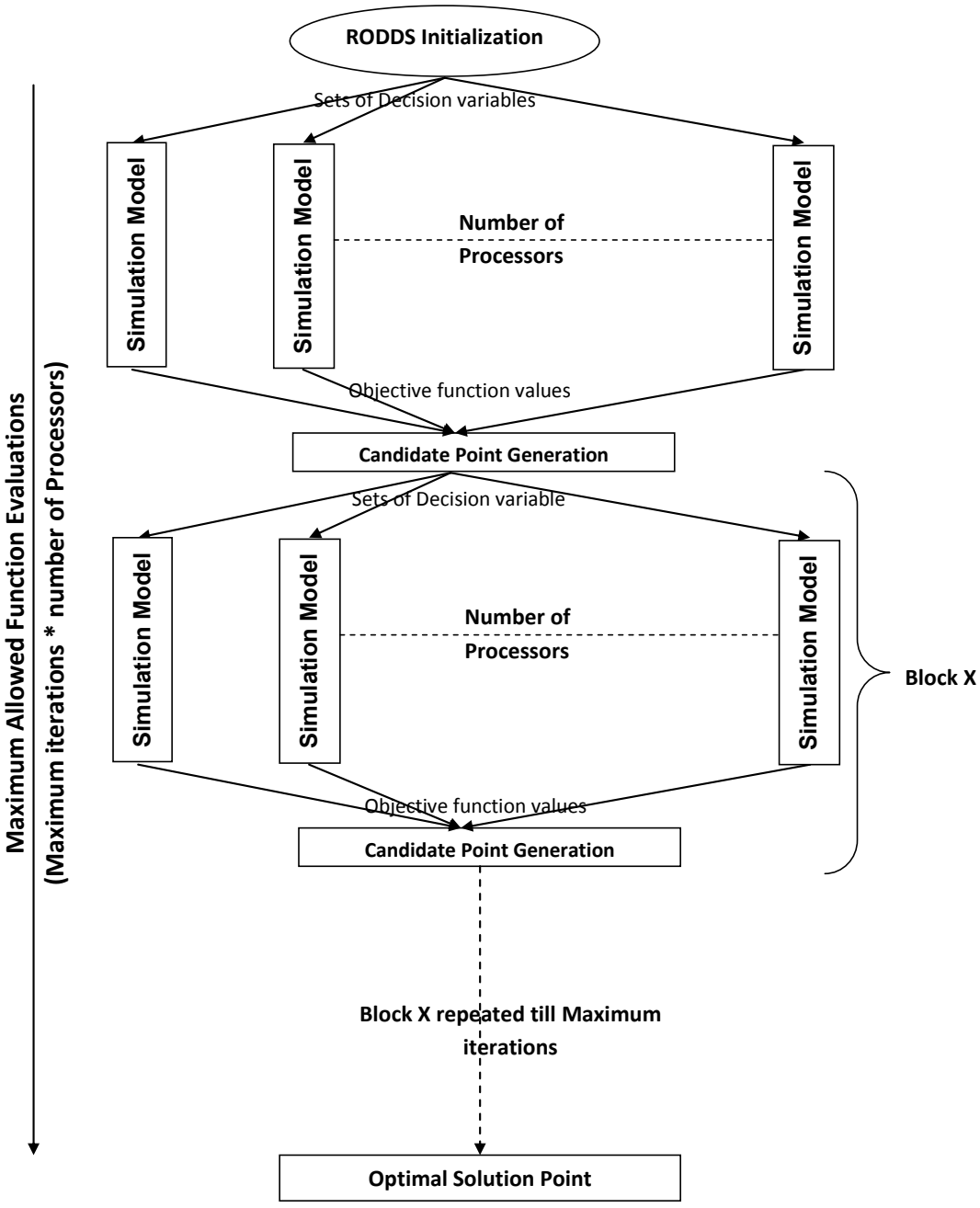


Figure A.1: RODDS Flowchart

A.2 shows the hypersphere idea for a two dimensional (two decision variable) model. Here circles represent hyperspheres at different stages of search process

i.e. dark ones represent the initial hyperspheres and lighter ones represent at later stage (hypersphere shrinks). At a particular iteration, candidate point generation process is repeated if the generated point falls inside the hypersphere.

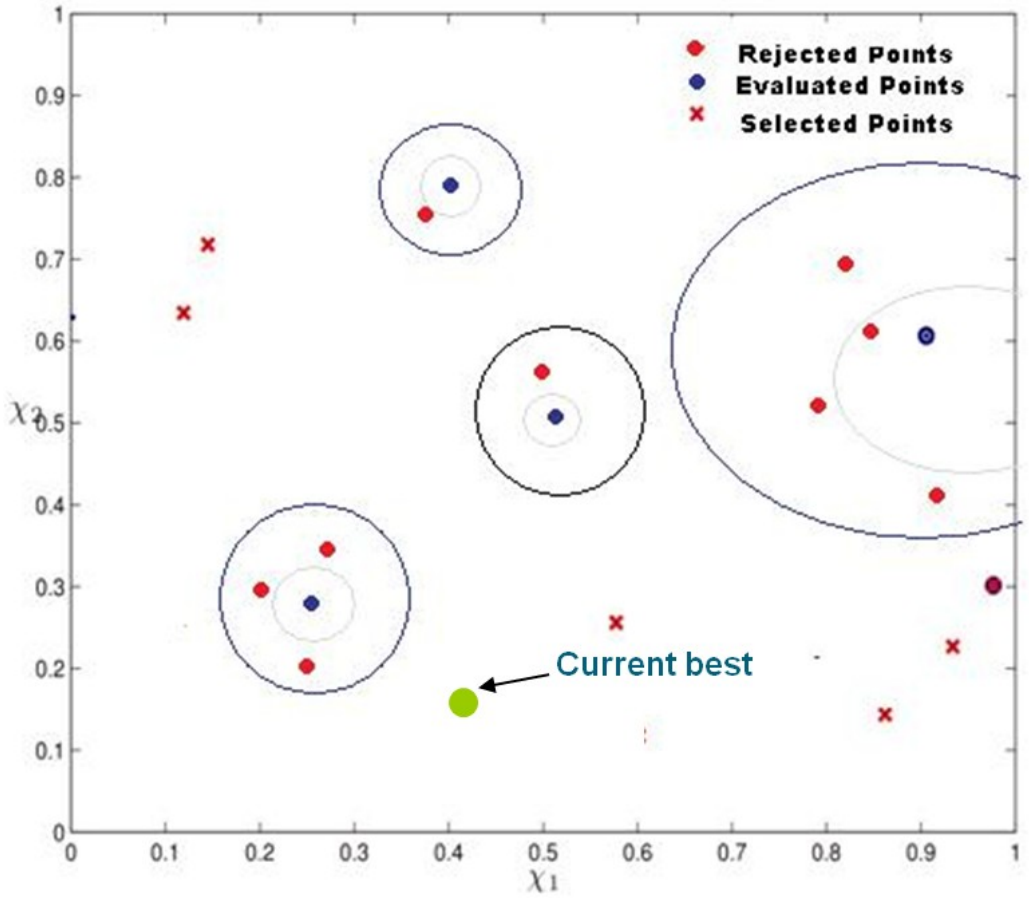


Figure A.2: RODDS Candidate point selection