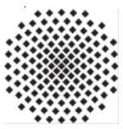


Universität Stuttgart - Institute for Modeling Hydraulic and Environmental Systems
Department of Hydromechanics and Modelling of Hydrosystems
Independent Junior Research Group: Stochastic Modelling of Hydrosystems
Prof. (jun.) Dr.-Ing. Wolfgang Nowak, M.Sc.

Auslandsorientierter Studiengang
Wasserwirtschaft
Master of Science Program
Water Resources Engineering and Management - WAREM

Master's Thesis

Optimizing early-warning monitoring systems for improved drinking water resource protection

Submitted by

Emmanuel Lelarge d'Ervau

Matriculation Number 2668291

Stuttgart, 8th of October 2013

Examiners: Jun.-Prof. Dr.-Ing. Wolfgang Nowak

Dr. rer. nat. Thomas Wöhling

Supervising Tutor: Dipl.-Ing. Felix Bode

I hereby certify that I have prepared this master's thesis independently, and that only those sources, aids and advisors that are duly noted herein have been used and / or consulted.

Date Signature

Acknowledgements

I want to thank Wolfgang Nowak and Felix Bode for supervising this work with insight and effectiveness, and all the members of the Independent Junior Research Group “Stochastic Modeling of Hydrosystems” for their scientific and technical support.

Abstract

Since the 2004 tsunami in the Indian Ocean, early-warning systems have become of public interest. Early-warning systems are monitoring systems which focus on the time that remains for action after a hazardous trigger event has been detected. By allowing in-time implementation of mitigation measures, they help protecting human lives as well as infrastructures and other goods of public interest.

There is a need for monitoring in drinking water production and supply systems because undetected contamination events can lead to severe consequences in terms of human health. Existing methods are useful, but show their limitations: protection zones are preventive measures, they do not aim at warning us when a contamination actually heads towards a well; on-line monitoring does not detect pollutions before the contaminants enter the system. With the implementation in well catchments of early-warning systems that feature high probabilities to detect contamination events as well as an early detection of these contaminations, more efficient mitigation measures could be taken.

This master's thesis sets up a method to design optimal early-warning systems for well catchments, while having knowledge about the potential contamination spots, with respect to three conflicting objectives: (1) maximizing the probability of detection of contaminations, (2) maximizing the early-warning time, i.e. the remaining time between detection and arrival at the production well, and (3) minimizing the cost of sampling the system. Advanced search algorithms (metaheuristics) are used to carry out this multi-objective optimization.

Contents

List of Figures

1 Introduction

1.1 Motivation

Since the 2004 tsunami in the Indian Ocean, early-warning systems have become of public interest. Early-warning systems are monitoring systems which focus on the time that remains for action after a hazardous trigger event has been detected. By allowing in-time implementation of mitigation measures, they help protecting not only human lives, but also infrastructures and other goods of public interest, and are therefore widely used in risk reduction strategies. The Famine Early Warning System Network (1985) (designed to deal with food insecurity) and the Ballistic Missile Early Warning System (1959) (first operational ballistic missile detection radar) are typical examples of early-warning systems.

Also in drinking water production and supply systems there is a need for monitoring, since undetected contamination events can lead to severe consequences in terms of human health. At the level of well catchments, protection zones are usually established, such as the 50-day line intended to neutralize biological pollutants (Mull, 1981; DVGW, 2006). In addition, on-line monitoring, where it exists, takes the form of on-line sensors within the urban water system. A review concerning existing sensor technology and optimal sensor placement in the water distribution network was conducted by Storey et al. (2011). These methods are useful, but show their limitations: protection zones are preventive measures, they do not aim at warning us when a contamination actually heads towards a well; on-line monitoring does not detect pollutions before the contaminants enter the system. With the implementation in well catchments of early-warning systems that feature high probabilities to detect contamination events as well as an early detection of these contaminations, more efficient mitigation measures could be taken (e.g. such as described in the DVWG monitoring guideline, 2009).

A groundwater monitoring system is composed of a number of monitoring wells designed to collect information about underground water contamination. Possible criteria on how to design such systems include maximizing detection probability, or estimating the characteristics of a plume, e.g. estimating a plume map or contaminant mass. Due to the character of the problem setting, the design of monitoring systems (i.e. selecting the location of the monitoring wells) requires the use of global optimization tools such as metaheuristics. In addition to the detection probability and the warning time (time before a detected contaminant arrives at the well), the design of an early-warning system in a well catchment requires to take into account the cost of the system. These three concerns provide the basis for a design method based on multi-objective optimization. As shown in Section 1.2, little research has been carried on, within the groundwater monitoring field, on the topic of early-warning. The only appearance of the early-warning aspect in the groundwater area addresses the monitoring of landfills, where there is just one possible contamination source to monitor (Yenigül et al., 2006). Well catchments strongly differ from landfills in that there are many (and not just one) potentially hazardous sites or contamination sources.

1.2 State of the art for metaheuristics in groundwater monitoring

This section deals with the application of metaheuristics in the area of optimizing groundwater monitoring. Metaheuristics are guided random search techniques which provide, in a reasonable time, valuable though non-optimal solutions to hard and complex problems in science and engineering. An insight into methodological details of metaheuristics, explanation why they are essential to the design of monitoring systems, and a description of the algorithms mentioned in this section are provided in Chapter 2.

1.2.1 Historical landmarks

Metaheuristics are not a recent topic: the first genetic algorithm was proposed by Holland (1975), later followed by the discoveries of other methods such as simulated annealing

(Kirkpatrick et al., 1983) and tabu search (Glover, 1989). However, new variants and applications are continuously being found, taking advantage of the ever-increasing computational power of computers.

Due to the complexity of groundwater problems, the use of metaheuristics to address them has been widely explored. The first use of a metaheuristic to address a groundwater management optimization problem was reported by Dougherty and Marryott (1991). They used a simulated annealing algorithm to optimize remediation strategies. Later on, McKinney and Lin (1994) were the first to make use of a genetic algorithm for a groundwater supply optimization problem.

Cieniawski et al. (1995) implemented the first application of a multi-objective algorithm to a groundwater monitoring problem. Since the early 2000s, efforts have focused on developing multi-objective algorithms that efficiently produce in a single run a wide range of optimal solutions (solutions that cannot be improved with regard to one of the objectives without decreasing the performance concerning any other objective). An important milestone towards this target was the NSGA II algorithm (Deb et al., 2002b), which reduced the computational costs with regard to its predecessors while enhancing diversity (i.e. the ability of the algorithm to screen the entire search space).

1.2.2 State-of-the-art

Reed et al. (2013) conducted a study providing a clear state-of-the-field assessment for multi-objective optimization within water resources applications. Several algorithms showed good performances: GDE3 - a differential evolution algorithm (Kukkonen and Deb, 2006), epsilon-NSGA II - a classical genetic algorithm (Kollat and Reed, 2006), AMALGAM - another auto-adaptive algorithm (Vrugt and Robinson, 2007) and OMOPSO - a particle swarm optimization algorithm (Sierra and Coello, 2005). Yet the algorithm which performed the best, with consistent results on all the applications, was an auto-adaptive evolutionary algorithm called BORG. BORG works with an entire collection of possible optimization operators within the area of genetic algorithms, and features a highly versatile and efficient adaptation strategy to select those operators that contribute most to the optimization success. This result led the authors to call for a focus on developing new auto-adaptive algorithms.

Apart from OMOPSO, all the algorithms included in the above-mentioned study were genetic algorithms. Genetics algorithms (GAs) combine mutation and crossover operators to make a population of solutions evolve through mating and replacement schemes towards good solutions. Suspecting that GAs may not be sufficiently adapted to problems featuring complex interdependencies or correlations between decision variables (such as network design problems), some researchers have explored other ways. For example, the epsilon-hBOA (Kollat et al., 2008) algorithm builds upon a probabilistic model. Another example is the use of an ant colony optimization paradigm by Li and Chan Hilton (2007). In both cases, these algorithms applied to a network design problem gave better results than genetic algorithms. Dhar and Datta (2009) developed a methodology to formulate a network design problem as a linear optimization problem, thus ensuring that the optimal solution is found with a single run of a branch-and-bound algorithm. This method, however, is restricted to the redundancy reduction of a network designed to evaluate the global characteristics of a plume.

Also in the groundwater area, though not about monitoring, Jha and Datta (2012) recently used an adaptive simulated annealing algorithm to identify the characteristics of a contaminant source, while Yang et al. (2013) adapted a tabu search method to the multiobjective optimization of a remediation system.

Groundwater systems applications of metaheuristics basically fall into two parts: groundwater remediation and groundwater monitoring. Indeed, most of the studies addressing groundwater monitoring deal with getting information about a contaminant plume. Thus, in spite of the relatively abundant literature on the subject of monitoring, little has been done on the topic of early warning. Two studies were found that approach this topic. Yenigül et al. (2006) considered the design of a monitoring system downstream of a landfill subjected to leakage. Their approach presented three objectives: (1) maximization of the probability of detection of contaminant plumes, (2) minimization of the contaminated area before detection, and (3) minimization of the cost of the system. The second objective is equivalent to detecting the plume as early as possible. However, no optimization algorithm was used and only nine potential designs were considered (rows of a varying number of monitoring wells at a larger or smaller distance of the landfill). Papapetridis and Paleologos (2011) proposed a similar approach, with an emphasis on the importance of the number of simulations and tracking particles.

1.3 Goal, approach and contribution

Goal

The goal of this work is to set up a method to design optimal early-warning systems for well catchments, while having knowledge about the potential contamination spots, with respect to three conflicting objectives: (1) maximizing the probability of detection of contaminations, (2) maximizing the early-warning time (i.e. the remaining time between detection and arrival at the production well), and (3) minimizing the cost of sampling the system.

Approach

A numerical flow and transport model for a synthetic, exemplary drinking water catchment is available. A huge range of optimization algorithms can be found in the literature. The idea is to define and implement an appropriate suite of objective functions which use information provided by the model to evaluate a given design (i.e. number and location of the monitoring wells), and to couple these functions with an optimization algorithm. An extensive study of search optimization techniques is carried out in order to find appropriate state-of-the-art algorithms. The most promising candidate is then selected and applied to our problem. Attempts to enhance it so that it better fits our problem are carried out. Application to simple test cases brings to light distinctive features of effective early-warning system designs.

Novelty and contribution

To the best of my knowledge and as shown in Section 1.2.2, early-warning systems in well catchment areas have not been investigated yet. With the implementation of such systems that feature high probabilities to detect contamination events as well as an early detection of these contaminations, more efficient mitigation measures could be taken. Thus, an increased safety in drinking water supply systems will be achieved.

1.4 Structure of the thesis

This thesis falls into three parts:

- theoretical background (chapters 2)

- preliminary work (chapters 3)

- personal contribution (chapters 4 to 8)

Chapter 2 provides an overview of search optimization techniques, and introduces the tools which will be used to evaluate and compare algorithms.

Chapter 3 presents the flow and transport model developed and implemented in a related project.

Chapter 4 gives details about the formulation of the problem (choice of the objective functions and choice of the way the design problem is encoded - binary or continuous). Chapter 5 presents the chosen algorithm and the changes that were brought to it. In Chapter 6, results are shown and discussed. Chapter 7 summarizes the results and draws conclusions. Chapter 8 provides a short outlook.

2 Metaheuristics

The goal of this Chapter is to provide the basis for the selection of the metaheuristics that are appropriate for the design of an early-warning monitoring network. Section 2.1 to Section 2.3 provide an overview of the different metaheuristics available in the literature. Then the key compounds for multi-objective optimization are reviewed (Section 2.4). The actual choice of algorithms will be done in Chapter 5, based on the information presented in this Chapter, on the results of a diagnostic assessment of state-of-the-art multiobjective evolutionary algorithms (Reed et al., 2013), and on the formulation of the problem, presented in Chapter 4.

2.1 Definition and necessity for a metaheuristic

As stated before, metaheuristics are guided random search techniques which provide, in a reasonable time, valuable though non-optimal solutions to hard and complex problems in science and engineering. As guided, they differ from the totally random Monte Carlo method. As random, they oppose calculus-based techniques such as the Newton method, which makes use of gradient information in order to find the right direction to the optimal solution. They also oppose enumerative techniques which scan the whole domain of potential solutions. For problems that cannot be solved within satisfying search time by calculus-based or enumerative techniques, the use of a metaheuristic to obtain an approximate solution is justified.

In designing a metaheuristic, two competing criteria need to be taken into consideration: diversification and intensification. Diversification aims at visiting new regions of the search space to ensure that the search space is evenly explored. In intensification, regions

where good solutions have been found are further explored in the hope of finding better solutions (Talbi, 2009).

Like many other combinatorial optimization problems, the one featured here - known as facility location problem - belongs to the NP-hard class of problems (Wikipedia, 2013). This implies that it cannot be solved in polynomial-time by an algorithm that would guarantee to find the best solution. Moreover, the size of the search space makes the use of enumerative techniques unsuitable. Therefore, the use of a metaheuristic appears as being both justified and necessary.

Metaheuristics can be sub-divided into those that are single-solution based and those that are population-based. Both groups are discussed in the following (Section 2.2 and Section 2.3).

2.2 Single-solution-based metaheuristics

2.2.1 Simulated annealing

Simulated annealing (Kirkpatrick et al., 1983) is inspired by controlled cooling processes in metallurgy, which aim at minimizing the number of dislocations in the material through a slow cooling. A small rate of defects corresponds to a low thermodynamic free energy. If the cooling is too fast, the diffusion process fails to destroy the dislocations and the system gets stuck in a metastable state (local optimum).

The objective function of the metaheuristic is associated with the thermodynamic free energy of the system, while the decision variables of our problem correspond to the positions of the molecules in the material. The fundamental idea is to allow moves resulting in solutions of worse quality than the current solution (i.e. an increase in the free energy) in order to escape from local minima.

At each iteration, a new solution is randomly generated within the neighborhood of the current one. If the evaluation function gives a better result (i.e. lower free energy) for the new solution, then it is automatically selected. Otherwise, the new solution is selected with a probability that depends on the current value of the “temperature” parameter and

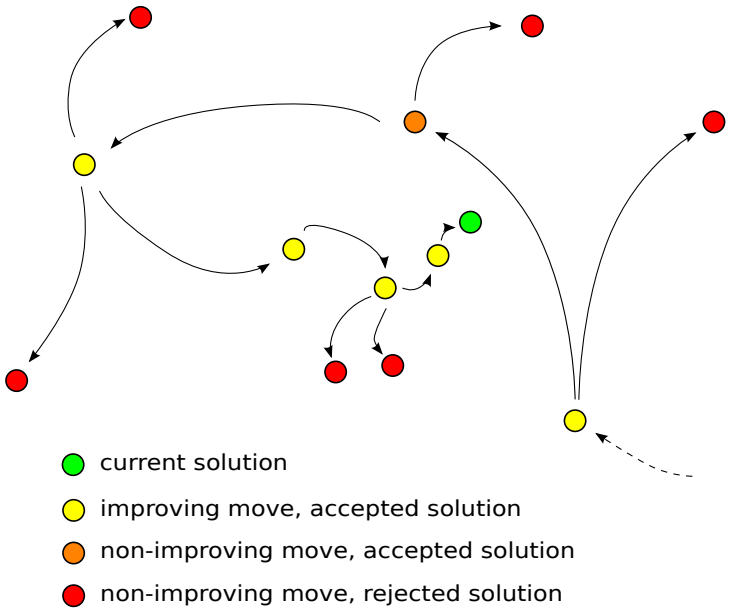


Figure 2.1: Simulated annealing

on the degradation of the evaluation function. The greater this degradation (difference between the evaluations of the old and new solutions), the smaller the probability for the new solution to be selected. As the algorithm processes, the probability that such uphill moves are accepted decreases, and at the end of the search non-improving moves are not accepted anymore.

2.2.2 Tabu search

Tabu search (Glover, 1989) proposes another approach to escape from local minima. The particular feature of tabu search is the use of memory to store information related to the search process.

At every step, the algorithm deterministically explores the neighborhood of the current solution (e.g., in the case of a binary representation of the solutions, all the solutions that differ in one binary digit from the current one are tested). Then the best solution found is selected, whether it is better than the current one or not (the latter would be the case where the current solution was a local optimum). Three memory structures are used to enhance the search. The tabu list (or short-term memory) stores a constant number of recently visited solutions to exclude them from the search, thus avoiding cycles. The

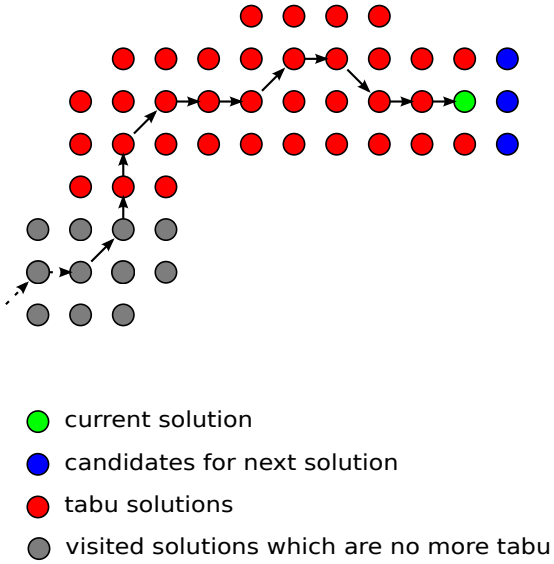


Figure 2.2: Tabu search

medium memory stores the best solutions found during the search, while the long-term memory stores information about the already visited areas. Both medium and long-term memory promote intensification and diversification of the search (Talbi, 2009).

Yang et al. (2013) invented a tabu search algorithm specially designed to address multi-objective design of groundwater remediation systems, called the Niched Pareto Tabu Search algorithm. This algorithm features components designed to deal with the multi-objective aspect of the problem, such as selection and archiving mechanisms similar to the ones described in Section 2.4.

2.3 Population-based metaheuristics

Population-based metaheuristics differ from single-solution based metaheuristics in that, at each iteration, not a single new solution is generated but an entire set (called a population). This new population is then integrated into the current one using selection procedures. The search process goes on until a given condition is met (e.g. convergence criterion, or a given number of calls to the evaluation function).

The generation and selection procedures vary from one algorithm to another, as reviewed in the following.

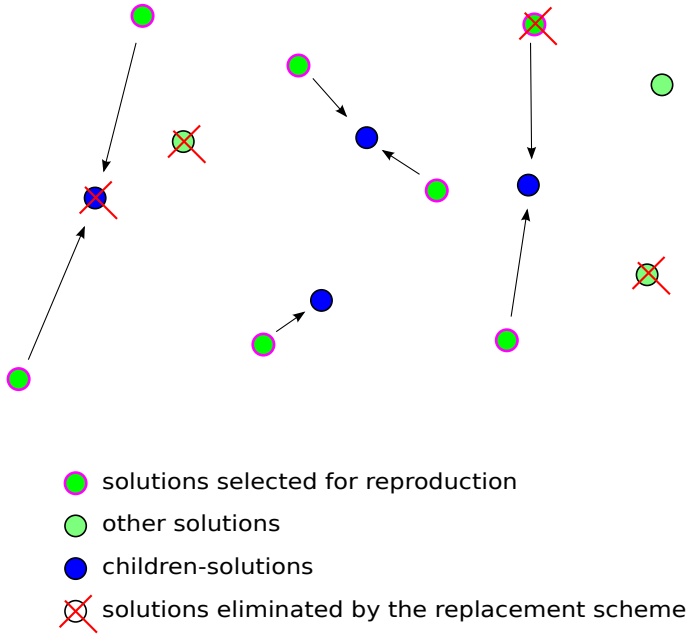


Figure 2.3: Genetic algorithm

2.3.1 Genetic algorithms

Genetic algorithms belong to the larger group of evolution algorithms. Evolution algorithms are the most studied population-based algorithms (Talbi, 2009). They use mechanisms inspired by biology and the theory of evolution such as reproduction, mutation and selection. At each iteration, parent solutions are stochastically selected from the current population, with good solutions having a better probability of being chosen. Variation operators are then applied to the parent solutions to generate an offspring. Finally, part of the population formed by the offspring and their parents is eliminated via a replacement scheme. The process goes on until some suitable set of termination criteria is reached.

Genetic algorithms, which are the most frequently used evolution algorithms (Nicklow et al., 2009), usually combine a crossover operator (two parent-solutions are mixed to form one or several children-solutions) with a mutation operator that randomly modifies one or several variables to enhance diversity.

2.3.2 Probabilistic model building algorithms

Probabilistic model building algorithms are evolution algorithms that gather information about the interdependencies between decision variables. At each step, a probabilistic model is built, based upon a selection of promising solutions. This model represents the decision variables and their conditional dependencies (e.g. probability of having a monitoring well at location j given that there is a monitoring well at location i). The probability distribution is then sampled to generate new solutions. This type of algorithm is particularly efficient when the decision variables are closely connected.

The hierarchical Bayesian optimization algorithm, developed by Pelikan (2005), is representative of this class of algorithms. The conditional dependencies between the decision variables are modeled by building Bayesian networks of the decision space, i.e. through the use of probabilistic directed acyclic graphs (Directed acyclic graphs are directed graphs with no directed cycles: no sequence of directed edges starting at a given vertex will eventually loop back to this vertex). This algorithm was applied to a long-term groundwater monitoring design problem and showed better performances than state-of-the-art genetic algorithms (Kollat et al., 2008).

2.3.3 Ant colony optimization

The idea behind ant colony optimization (ACO) algorithms is to take inspiration from the behavior of ants, which use very simple communication mechanisms to perform complex tasks (such as finding the shortest path to a piece of carrot). While moving, ants lay a chemical trail - with substances called pheromones - on the ground. This volatile substance plays the role of a guide for the ants: when facing several paths, the probability for an ant to choose any of them will be greater if the amount of pheromone laid on this path is large.

ACO is traditionally applied to discrete optimization problems. Li and Chan Hilton (2007) developed a search algorithm based on the ant colony optimization paradigm to solve long-term groundwater monitoring problems. The algorithm aims at determining, for an existing network, which wells can be removed so that the overall data loss due to this removal is minimized. Results for a test case with 30 monitoring wells showed that

the algorithm was able to find the global optimum for the cases with 21 to 27 remaining wells with only 1500 function evaluations (to be compared with the 14 307 150 function evaluations needed to solve the 21-well case with enumeration).

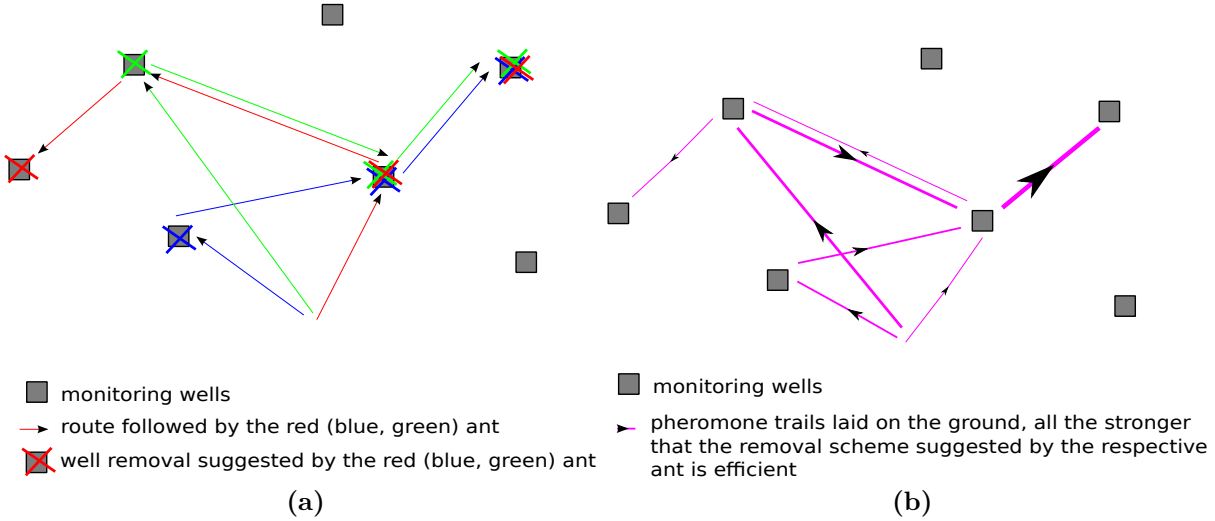


Figure 2.4: Ant colony optimization applied to groundwater monitoring for an efficient removal of superfluous wells

2.3.4 Particle swarm optimization

As well as the previous one, this metaheuristic is inspired from swarm intelligence. Here, a swarm consists of a population of particles flying around in the search space. The particles represent candidate solutions to the problem. At every iteration, each particle adjusts its velocity for moving in the search space as a function of two variables: the best position it visited so far, and the overall best position visited by the swarm. Thus, it will cycle around a weighted average of these two positions until it converges towards the global optimum.

Particle swarm optimization is traditionally applied to continuous optimization problems (Talbi, 2009). No application of this algorithm in the groundwater monitoring area was found in the literature.

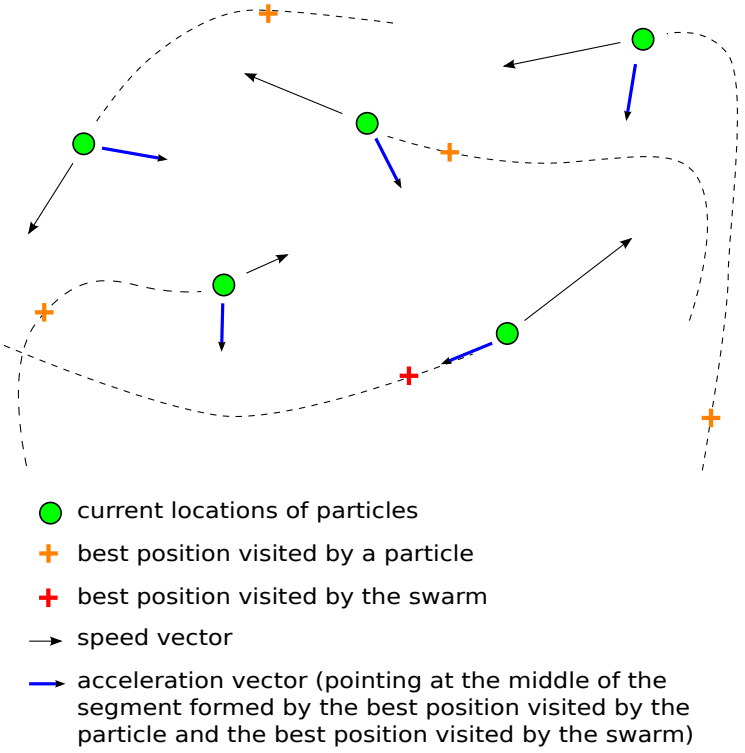


Figure 2.5: Particle swarm optimization

2.4 Components for multi-objective optimization

A simple approach to solve multiple objectives problems is to aggregate the objectives into a single objective by assigning relative weights to them, then to optimize for this aggregated function. The use of this method requires that:

- each of the objectives is clearly defined and measurable

- the objectives are commensurable

- the performance levels for different objectives can compensate for each other

- weighting and addition of the objectives are public and ethically justifiable

- the weighting is fixed a priori

In the case of an early-warning monitoring network, the performance levels associated with the detection probability, the average warning-time and the cost of the network

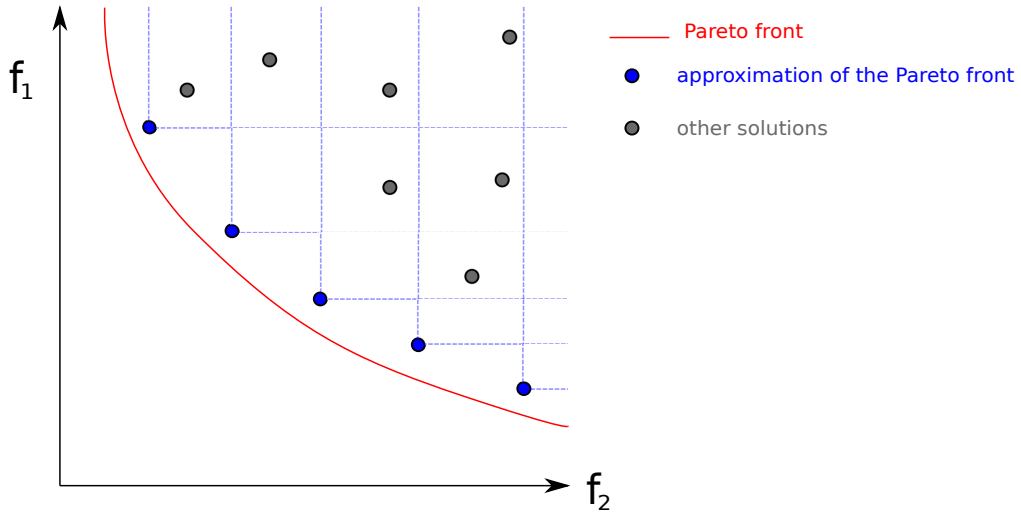


Figure 2.6: Pareto front

cannot compensate for each other. Therefore, the use of multi-objective optimization is justified.

2.4.1 Pareto domination

Multi-objective optimization aims at identifying the trade-offs between several design objectives of a problem. These trade-offs correspond to the solutions that are better than any possible other solution in at least one objective. They are called non-dominated or Pareto-optimal solutions (Pareto, 1896), and form the Pareto front. Thus, not a single solution is searched but the whole set of non-dominated solutions. The aim of a multi-objective metaheuristic is too obtain a good approximation of the Pareto front in cases where it cannot be determined analytically (Figure 2.6). In this context, good means close to the Pareto front, widespread and consistent (see Section 2.5). This approximation can then be exploited by decision makers to identify an appropriate compromise solution (Reed and Minsker, 2004).

An approach exists that aggregates the objectives into a single objective, this time with no fixed relative weights. Assuming the objectives are conflicting (e.g. cost vs. performance), different parts of the Pareto front can be generated by changing the weights. This approach has two drawbacks: it takes more time, and some portions of the Pareto front cannot be found (Nicklow et al., 2009).

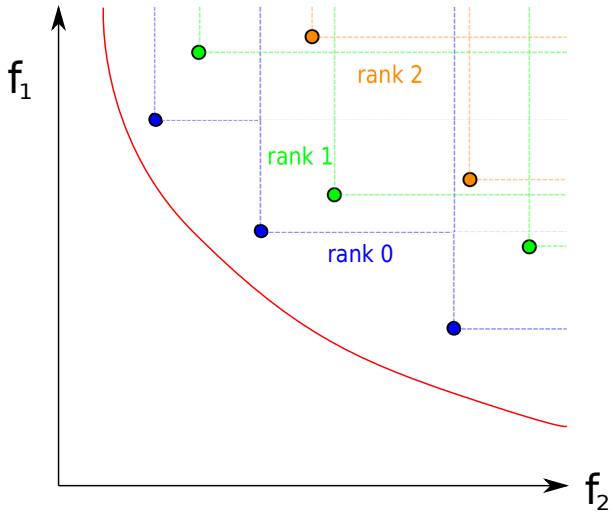


Figure 2.7: Non-dominated sorting

2.4.2 Non-dominated sorting and preservation of diversity

In order to solve multi-objective problems, metaheuristics need to be adapted. Most of the research in this field relates to genetic algorithms (Reed et al., 2013). To adapt genetic algorithms to multiobjective optimization, two key features are needed. The first one is a way to select good parent-solutions from the current population via a comparison operator based on non-dominated sorting (i.e. ranking of solutions that do not dominate each other). Here is how it works: each solution is assigned a non-domination rank which tells how many "layers of solutions", among the current population, separate this solution from the Pareto front. Solutions which are non-dominated are attributed rank zero. Among the remaining solutions, those which are dominated only by solutions of rank zero are attributed rank one, and so on. Figure 2.7 illustrates non-dominated sorting.

The second key feature is a way to preserve diversity among solutions of a non-dominated front in order to obtain a set of solutions that homogeneously covers the Pareto front.

In these regards, the release of the Non-dominated Sorting Genetic Algorithm II (NSGAII), developed by Deb et al. (2002b), represented a major progress. This algorithm introduced a faster way to rank the solutions of a population as well as a diversity maintenance strategy that did not require any parameterization. This strategy takes the form of a quantity called "crowding distance" which estimates how isolated a solution is on the Pareto front. This quantity is used during the replacement scheme (when part of

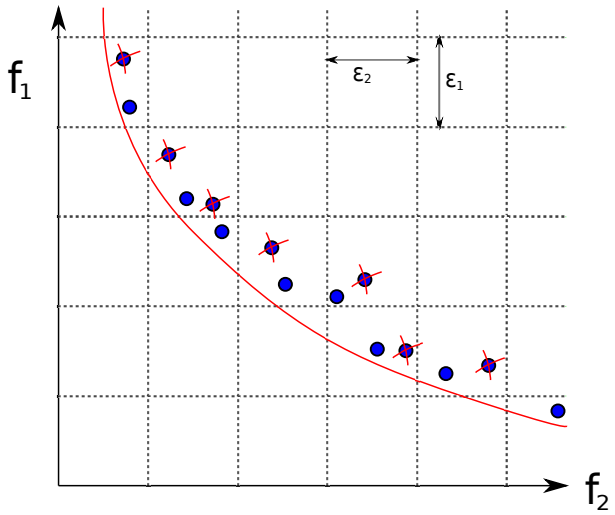


Figure 2.8: Epsilon-dominance: only one solution is kept within each box of dimension epsilon

the population is eliminated after a new generation has been generated): solutions are first sorted according to their non-domination rank, then the crowding distance is used to sort solutions with identical rank (the more isolated a solution, the higher it is ranked). Poorly-ranked solutions are then eliminated in order to reduce the population.

2.4.3 Elitism and epsilon-dominance archiving

During the algorithm run, the set of solutions that approximate the Pareto front (non-dominated solutions found so far) can be stored in an archive. This strategy is called elitism and prevents deterioration (loss of good solutions). The concept of epsilon-dominance, introduced by Laumanns et al. (2002), allows the user to specify the precision with which he wants to quantify each objective in a multi-objective problem (see Figure 2.8). The size of the associated archive is dynamic to maintain an approximate representation of the Pareto front with a resolution of epsilon.

One of the first algorithms to make use of this concept was the epsilon-MOEA algorithm (Deb et al., 2003). This algorithm was shown to achieve a better distribution of Pareto-optimal solutions than the NSGAII algorithm (see Section 2.4.2) while maintaining competitive computational times.

2.5 Diagnostic assessment tools for evaluating algorithms

According to Hadka and Reed (2012), the Multi Objective Evolutionary Algorithms (MOEA) literature does not show consensus on how multiobjective search performance should be evaluated and compared. After computing a broad range of performance metrics, Hadka and Reed picked out three quality metrics: (1) generational distance, (2) hypervolume, and (3) epsilon+ indicator. These metrics form a suitable combination to evaluate what they consider the three main functional objectives of MOEAs: proximity, diversity and consistency. They will be used in this study to evaluate and compare algorithms. Hadka (2012) developed a quite general software for multiobjective optimization, which will be used in this thesis for the calculation of the performance metrics.

Two of the abovementioned metrics require to measure distances from solutions to the Pareto front. Indeed the Pareto front is not known. Instead, a reference set is used, which is obtained by aggregating all the solutions obtained from all runs, and discarding the dominated ones.

2.5.1 Hypervolume

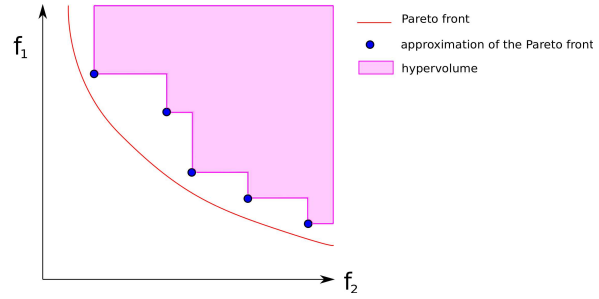
Hypervolume measures the volume of the objective space covered/dominated by the approximation set. It thus represents a combination of proximity and diversity.

2.5.2 Generational distance

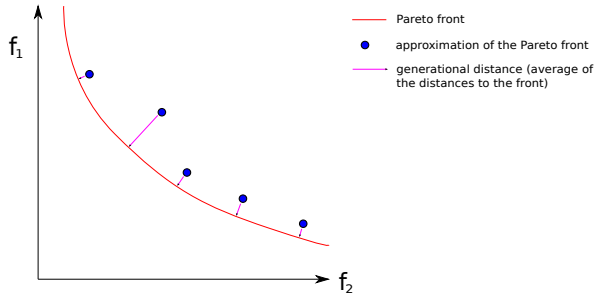
Generational distance is the average distance from objective vectors in the approximation set to the nearest objective vector in the reference set, thus representing proximity.

2.5.3 Epsilon+ indicator

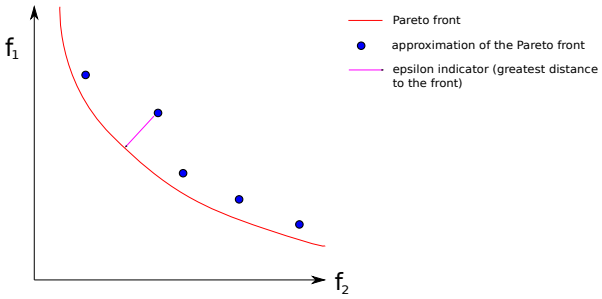
The epsilon+ indicator is the smallest distance the approximation set must be translated so that every objective vector in the reference set is covered. This identifies situations in which the approximation set contains one or more outlying objective vectors with poor



(a) Hypervolume



(b) Generational distance



(c) Epsilon indicator

Figure 2.9: Performance metrics

proximity. If an approximation set shows good proximity for the most part except for a few of its objective vectors, then this approximation set is not consistent.

3 System definition, numerical implementation and test cases

The design of an early-warning system in a water catchment requires data concerning the transport of possible contaminants in this area. In this study, a flow and transport model is applied to an exemplary drinking water catchment. To deal with the uncertainties concerning the hydraulic conductivity, a Monte-Carlo approach is used to generate a large number of conductivity fields, based on the concept of Bayesian geostatistics (Kitanidis, 1986). For a given pattern of possible contaminant sources and a given conductivity field, one run of the model results in the creation of a matrix which stores, for all potential locations of monitoring wells and all possible contaminant sources, the information whether detection is possible and, if appropriate, the associated warning time. This matrix is then used as an input by the objective functions of the optimization problem to evaluate potential network designs (see Chapter 4).

The flow and transport model is presented in Section 3.1. A description of the test cases designed for the evaluation of the multi-objective optimization algorithms is provided in Section 3.2.

3.1 Flow and transport model

The flow and transport model described in this section was carried out in MATLAB in a related project.

3.1.1 2D groundwater flow model

We consider a steady-state flow in a depth-averaged confined aquifer with constant thickness m. The aquifer is considered isotropic and heterogeneous. The groundwater flow equation is:

$$\frac{\partial}{\partial x}\left(K\frac{\partial h}{\partial x}\right)+\frac{\partial}{\partial y}\left(K\frac{\partial h}{\partial y}\right)=0$$

where:

- x and y are the Cartesian coordinates in a 2D Euclidian space $[L]$

- K is the hydraulic conductivity $[L/T]$

- h is the hydraulic head $[L]$

Dirichlet and Neumann boundary conditions are applied, as provided in Section 3.2.

The equation is solved using a Standard Galerkin Finite Element Method (Zienkiewicz and Taylor, 1977) in an implementation as described in Nowak et al. (2008).

3.1.2 Contaminant transport model

The advection-diffusion equation assuming no volumetric source/sink is:

$$\frac{\partial c}{\partial t}+v_x\frac{\partial c}{\partial x}+v_y\frac{\partial c}{\partial y}-\frac{\partial}{\partial x}\left(D_{xx}\frac{\partial c}{\partial x}\right)-\frac{\partial}{\partial y}\left(D_{yy}\frac{\partial c}{\partial y}\right)=r$$

where:

- c is the concentration of the contaminant at time t at location (x, y) $[M/L^3]$

- v_x and v_y are the average groundwater flow velocity components in the x- and y-direction $[L/T]$

- D_{xx} and D_{yy} are the components of the hydrodynamic dispersion tensor $[L^2/T]$

- r is the contaminant sources $[M/L^3/T]$

Dirichlet boundary conditions are applied, as provided in Section 3.2.

The equation is solved through a Particle Tracking Random Walk method (Kinzelbach, 1988; Salamon et al., 2006; LaBolle et al., 1996). The concentration reconstruction is based on Kernels.

3.2 Test cases

3.2.1 Parameters for a synthetic, exemplary drinking water catchment

We consider the design of an early-warning network in an illustrative study area with the dimensions $500 \times 500 \times 3m$ (see Figure 3.1). The boundary conditions for the groundwater flow are the following: constant-head boundaries at the left and at the right of the domain, and no-flow boundary at the top and bottom. A pumping well is located at the coordinates (400, 250) and features a pumping rate of 50e-3 m^3/s. The potential contaminant sources feature instantaneous leaks. The limit for a contaminant to be detected at a well is set to 4 particles $/m^2$ (the number of particles released per source is between ten and fifty million).

Hydraulic conductivity fields are generated with a Monte-Carlo approach based on second-order geostatistics (Delhomme, 1979) with a Matern covariance model (Matérn et al., 1960). The Matern model is a function of the variable κ, with the model being identical to the exponential covariance when $\kappa = 0.5$ and Gaussian when κ approaches infinity. Following the Bayesian approach to geostatistics (Kitanidis, 1986), the covariance function is itself subject to uncertainties. Here, κ is drawn out from a uniform distribution between 1 and 36. The parameters for the flow and transport model are presented Table 3.2. The variance is calculated with a Beta distribution function.

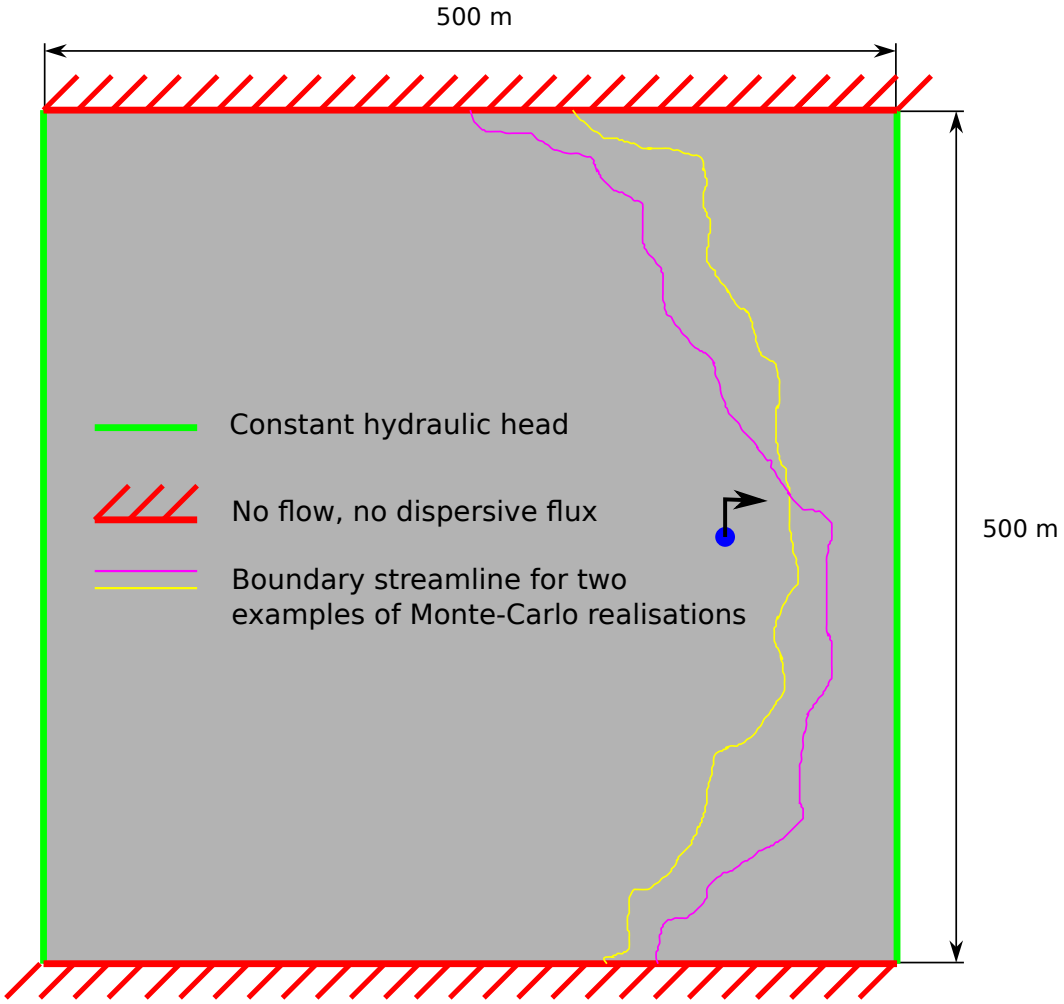


Figure 3.1: Illustrative study area

	Fixed value	Mean value	Variance
Hydraulic conductivity (m/s; m^2/s^2)	-	1e-4	2
Variance (m^2/s^2; m^4/s^4)	-	2	5
Correlation length (m; m^2)	-	15	6.5
Hydraulic head gradient (-)	0.05	-	-
Longitudinal dispersivity (m)	0.5	-	-
Transversal dispersivity (m)	0.125	-	-
Diffusion coefficient (m^2/s)	1e-9	-	-
Effective porosity (-)	0.35	-	-

Table 3.1: Parameters for the flow and transport model

	TC1	TC5	TC30
Number of particles generated for each source	5e4	5e4	1e4
Number of Monte-Carlo realisations	1000	200	30

Table 3.2: Parameters for the three test cases

3.2.2 Illustrative examples of contaminant sources patterns

Three test cases are designed for the evaluation of the optimization algorithms with one, five and thirty possible contaminant sources. These test cases are henceforth referred to as TC1, TC5 and TC30. From TC1 to TC5 and from TC5 to TC30, not only the number of contaminant sources increases but also their spreading over the catchment area. The aim of these test cases is to evaluate the performances of the algorithms for varied configurations of contaminant sources, and to observe how the complexity of the pattern formed by the possible contaminations impacts these performances.

The number of Monte-Carlo realizations decreases with an increasing number of contaminant sources due to computational reasons with regard to the optimization problem (the memory storage needed as well as the computational time are proportional both to the the number of possible contaminant sources and to the number of Monte-Carlo realizations). The number of particles generated for each source is lower for TC30 due to computational reason with regard to the flow and transport model (the running time is proportional to the total number of particles). The values for these parameters are displayed in Table 3.2.

3.2.2.1 TC1: one contaminant source

One contaminant source is located near the point of coordinates (100,250), with an uncertainty of $25m$ concerning its location (for every Monte-Carlo realisation, a different source location is generated within this radius). For this one source, 5×10^4 particles are generated within a radius of $5m$. 1000 Monte-Carlo realisations are carried out.

Figure 3.2 shows the location of the source. Figure 3.3 shows the contaminant plume obtained with one Monte-Carlo simulation. It can be observed that from the contamination source to the pumping well, the width of the plume increases first, then decreases. This increase of the width is due to dispersion, while the subsequent decrease reveals a sucking out of part of the particles by fast streams.

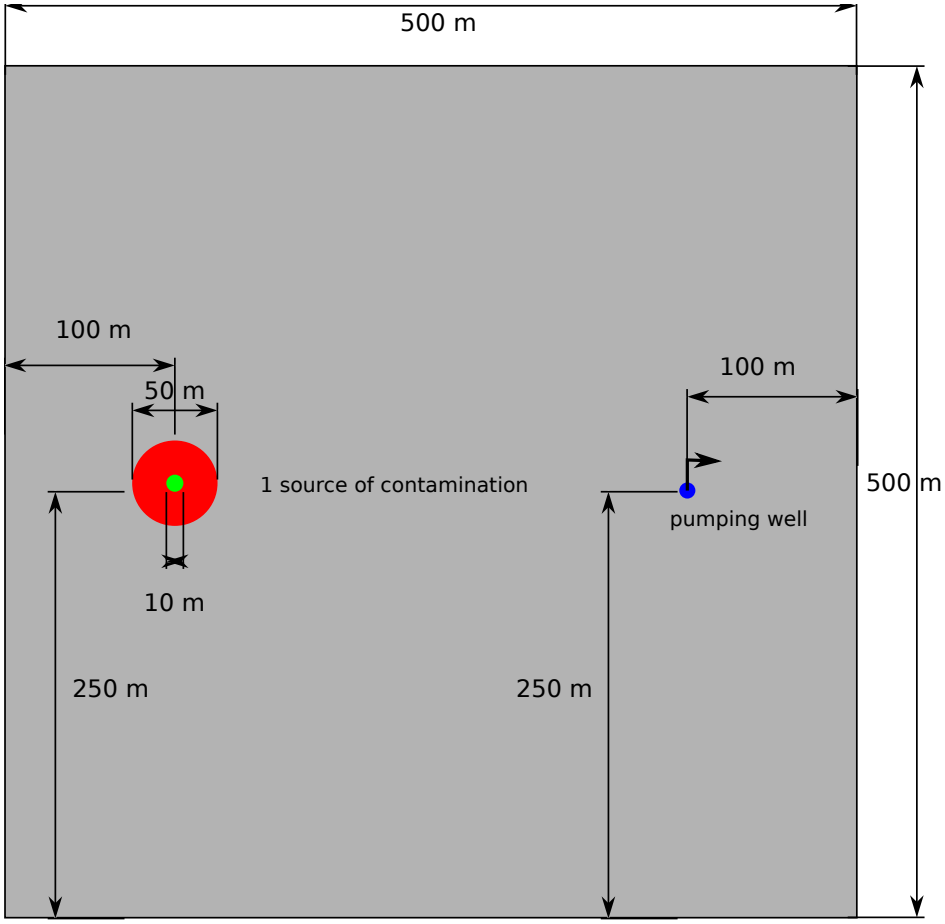


Figure 3.2: TC1: location of the contaminant source. In red, the incertitude area regarding the location of the source. In green, the extent of the source.

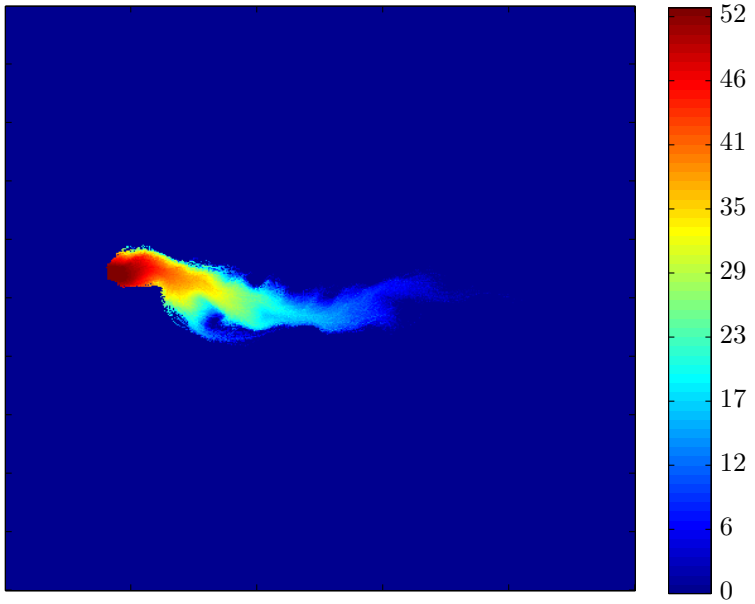


Figure 3.3: TC1: location and shape of the plume for one single Monte-Carlo simulation. The color corresponds to the warning time, in days.

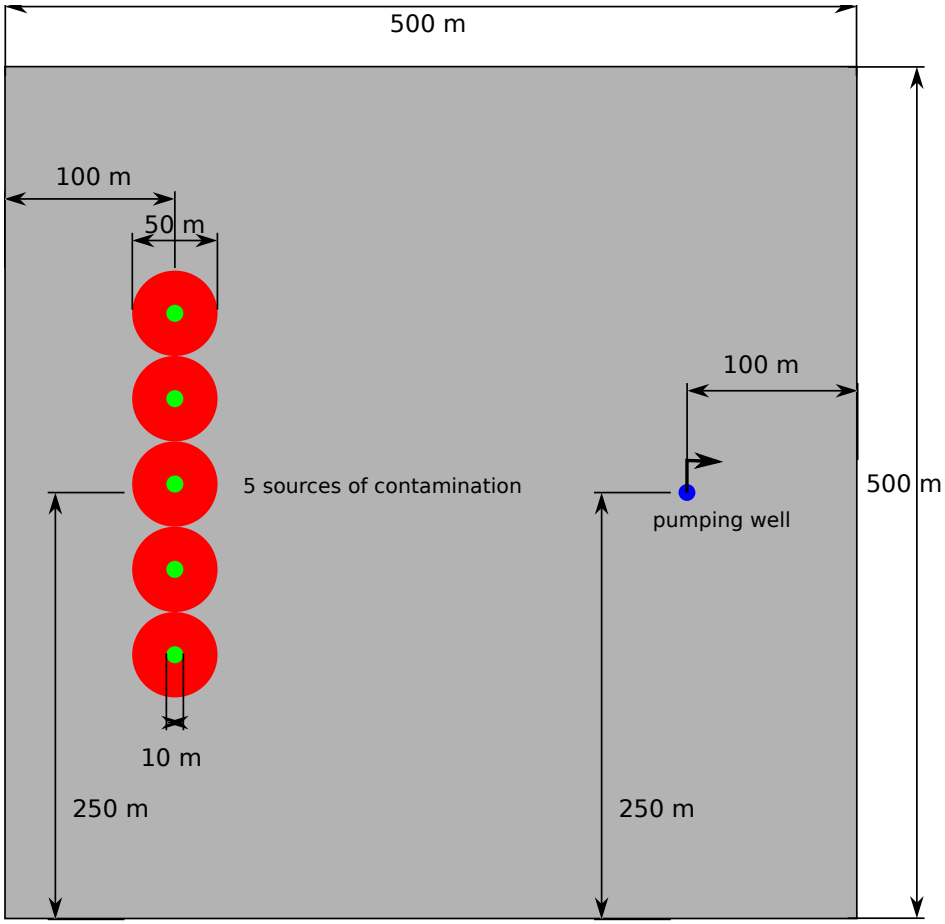


Figure 3.4: TC5: location of the contaminant sources. In red, the incertitude area regarding the location of the source. In green, the extent of the source.

3.2.2.2 Five contaminant sources in a row

Five contaminant sources are located in a row (see Figure 3.4), with uncertainties of $25m$ concerning their locations (for every Monte-Carlo realization, different source locations are generated within this radius). For each source, 5×10^4 particles are generated within a radius of $5m$. 200 Monte-Carlo realisations are carried out.

3.2.2.3 Thirty sources of contamination, randomly located

Thirty sources of contamination are randomly generated within the following domain: $25m < x < 350m$, $25m < y < 475m$ (see Figure 3.5). Again, for each source, there is

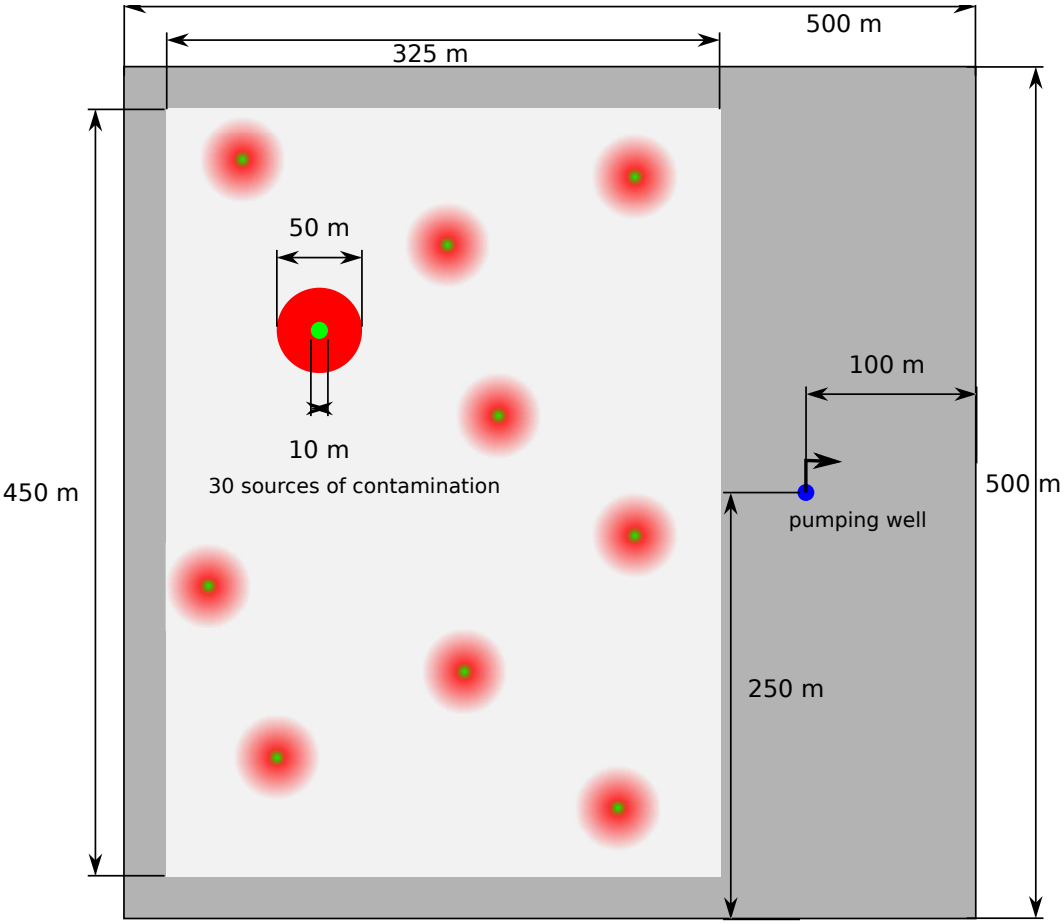


Figure 3.5: TC30: location of the contaminant sources. In red, the incertitude area regarding the location of the source. In green, the extent of the source.

a radial uncertainty of $25m$ concerning its location. For each source, 10^4 particles are generated within a radius of $5m$. Thirty Monte-Carlo realizations are carried out.

4 Formulation of the optimization problem: design of an early-warning monitoring network

The two key elements when formulating an optimization problem are the choice of the objective functions (Section 4.1) and the way candidate solutions are represented (binary or continuous representation, Section 4.2). A bad choice of the objective functions can lead to solutions that do not fully comply with the decision maker's actual concerns. The representation of solutions has a large impact on the way search operators operate, notably because the neighborhood of a solution depends on its representation (e.g. to change the value of a variable results in a change of the number of wells in the case of a binary representation, while in the case of a continuous representation it results in the shifting of a well). Moreover, as seen in Chapter 2, heuristics can be more adapted to one representation than to the other one.

4.1 Choice of the objective functions

In this study, the three performance objectives are rather straightforward. They include (1) minimizing the cost of sampling the system, (2) maximizing the detection probability of contaminations, and (3) maximizing the early-warning time, i.e., the time between the detection of a contamination and its arrival at the drinking water well.

The sampling cost of a monitoring system obviously grows with the number of monitoring wells present in the design. Therefore, the first objective function is the **number of monitoring wells**. In later studies, this may be extended to account for drilling costs

that depend on location, for the re-use of pre-existing monitoring wells, or for different sampling intervals and analysis costs.

The second objective function is the **detection probability** of contaminations. It is easily obtained by averaging the binary detection outcomes (detected or not) over all realizations and all contaminations. In later studies, this may be extended by a weighting that accounts for the possibly different seriousnesses of individual possible sources, both in failure frequency or predicted impact on the well.

The third objective concerns the **warning time**. For every hazard and realization, if the hazard is detected (by one or several monitoring wells of the scheme), the earliest time of detection is picked out. The associated warning time t is then transformed into a "utility warning time" t_u defined as follows (see Figure 4.1):

- if $t < t_1$, then $t_u = 0$

- if $t > t_2$, then $t_u = 1$

- else $t_u = \frac{t-t_1}{t_2-t_1}$

where t_1 and t_2 are respectively the limit below which a warning time is too short to be useful and the limit beyond which a gain of time is useless. All the utility warning times are then averaged over all realizations and all contaminations. This average is the third objective function. This choice of utility warning time is rather pragmatic, and may be altered easily to account for the specific preferences of catchment operators, or to specific properties of individual contaminant sources.

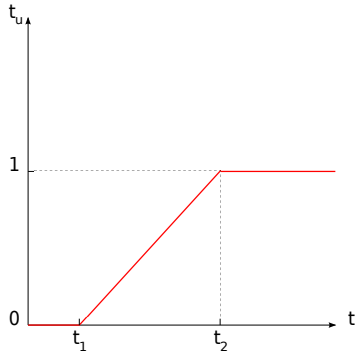


Figure 4.1: Utility warning time t_u

For the reasons mentioned in Section 2.4, it is more appropriate to solve the problem by generating the whole Pareto front via multiobjective optimisation rather than to aggregate the three objectives in a single one.

4.2 Choice of the representation

The problem can be represented as a continuous problem or as a binary problem, both presenting advantages and drawbacks as discussed in the following.

4.2.1 Binary representation

In the binary form, a solution is represented by an array of binary variables, where every variable corresponds to a possible location for a monitoring well. The variables take the value 1 if there is a well at the corresponding location, and 0 if there is none. This representation is the one which is usually chosen in the case of long-term groundwater monitoring design optimization, because most studies consider a limited number of possible locations (e.g. 25 in Kollat and Reed, 2006). It is indeed a must when the space where monitoring wells can be implemented is already restricted to a discrete set of locations. Moreover, the number of variables does not depend on the number of monitoring wells. This has benefits of simplicity, because the size of the solution vector (and hence the dimensionality of the problem) is fixed.

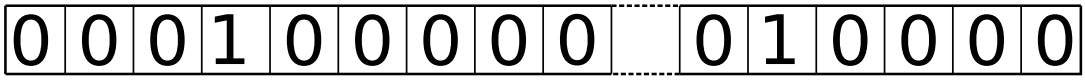


Figure 4.2: Binary representation of a solution

In the case of this study, however, no restrictive set of possible locations is predefined: monitoring wells can be located anywhere. In fact, the set of possible locations forms a fine mesh that covers the whole domain. This leads to an explosion of the number of variables from around 50 in traditional long-term groundwater monitoring design schemes to 250, 000 for a 500 times 500 mesh. The latter corresponds to a mesh width of one meter in our test cases, and would result in an unacceptable cost for a single call to the

evaluation function. For this reason, it is chosen to represent the problem as a continuous one, as explained next.

4.2.2 Continuous representation

In the continuous form, a solution is represented by an array of real variables, where the variables are coupled two by two, each couple of variables representing the coordinates of a monitoring well. An extra integer variable indicates how many monitoring wells the solution consists of.

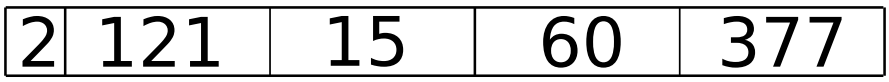


Figure 4.3: Continuous representation of a solution

One thing that must be considered when choosing the continuous representation is that it is generally not appropriate for combinatorial treatment. Therefore, the associated operators might be inadequate, i.e., inefficient. For example, crossover operators for genetic algorithms form children-solutions from parent-solutions in the following way: parent-values for the first variable are associated and mixed to form the first variable of the children-solutions, idem for the other variables. Thus, in our case, the coordinates of the first well of a children-solution will be computed from the coordinates of the first wells of the parents solutions, and so on. However, since the order of wells in a solution array are not correlated to their spatial location, this comes down to mixing the coordinates of randomly chosen parent-wells to compute the coordinates of the children-wells. This issue will deserve special attention in Chapter 5.

5 Choice and implementation of the algorithms

In this chapter, the most promising algorithm is picked out from the ones reviewed in Chapter 2 (Section 5.1). It is then modified, as a first step in order to adapt it to the problem (Section 5.2). Then, an attempt to measure and improve its performance is considered in Section 5.3 and Section 5.4. In Section 5.5, an alternative algorithm is considered.

5.1 Choice of the algorithm

BORG is a multi-objective evolutionary algorithm introduced by Hadka and Reed (2013) and based on the epsilon-MOEA algorithm introduced in Chapter 2. In addition, it features several interesting functionalities, among which are adaptive operator selection, adaptive population sizing and a measure of the convergence speed.

Adaptive operator selection is the most notable one. Instead of only one crossover and one mutation, many operators are available (see Section 5.2). At the beginning of the run, all of them have the same probability of being selected, leading to an application to equally distributed numbers of solution individuals within a population. After a given number of iterations, the probability for an operator to be selected becomes proportional to the number of solutions in the archive (i.e., non-dominated solutions) that have been found due to this operator. Thus, successful operators get a higher selection probability. This flexibility allows for a large assortment of problems to be addressed.

Adaptive population sizing maintains the population size proportionate to the size of the archive, i.e., to the size of the Pareto front. When the ratio $population\ size/archive\ size$

steps below a given level, new individuals are injected into the population. A large Pareto front is associated to a large diversity within the set of solutions we are looking for. By increasing the population when the Pareto front is large, the diversity within the population also increases, what favors search diversity.

The third characteristic is a **measure of the convergence speed** named epsilon-progress. This tool detects stagnation of the search. When, during a given time, no new solution can be found which represents a significant improvement, the algorithm triggers restart mechanisms, such as the generation of a new random population. As seen in Section 2.4.3, the archive where non-dominated solutions are stored has a resolution of epsilon, what means that each square (in a case with two objectives; if there are three objectives it is a cube) of side epsilon contains only one solution. A new solution represents a significant progress if it occupies a previously empty square.

Evaluated along nine other MOEA on a representative suite of water resources applications including long-term groundwater monitoring (Reed et al. (2013)), BORG showed the best performances. For this reason and because of its high flexibility, allowing us to easily modify or add operators, it is chosen to begin the study with this algorithm.

5.2 Adaptation of existing operators and addition of a mutation operator

Seven continuous operators are integrated in the BORG algorithm: five crossover operators and two mutation operators. These need to be adapted to the chosen continuous representation of the problem (see Section 4.2.2). Moreover, one additional operator has been added.

Presentation of the operators

The operators integrated within the BORG algorithm are the following:

- Simulated binary crossover (SBX; Agrawal et al., 1994). This operator is similar to the uniform crossover used in binary representation: each variable of a child-solution basically

has 50% chance of inheriting the value for this variable from its first parent, and 50% from its second parent.

- differential evolution (DE; Storn and Price, 1997): the differences between the variables of two parent solutions are calculated. A child solution is produced by adding these differences to the variables of a third parent solution.

- parent-centric crossover (PCE; Deb et al., 2002a): the offspring distribution is centered around the parent solutions.

- simplex crossover (SPX; Tsutsui et al., 1999): the offspring distribution occupies all the space between the parent solutions.

- unimodal normal distribution crossover (UNDX; Kita et al., 1999): the offspring distribution is centered around the mean of the parent solutions.

- uniform mutation (UM): each variable is mutated with a probability of $1/number\ of\ variables$. The probability distribution for the new value is uniform.

- polynomial mutation (PM; Agrawal et al., 1994): each variable is mutated with a probability of $1/number\ of\ variables$. The probability distribution for the new value is centered on the present value.

Adaptation of existing operators

It has been chosen (see Chapter 4) to represent solutions as an array of variables, where the first variable is discrete and represents the number of monitoring wells, while the others are continuous and represent the coordinates of the wells. Thus, different solutions have different numbers of variables. All crossover operators built-in in BORG are modified in order to affect only variables which are present in all the parent solutions: if one parent has five monitoring wells and the second parent only two, then the crossover operator is applied to the coordinates of the first two wells and leaves the last three wells of the first parent unaffected. The operators are also modified so that they do not affect the first variable (which corresponds to the number of monitoring wells): a crossover between a five-well parent and a two-well parent will produce two children solutions, featuring five and two wells.

A modification is brought to the SBX operator so that the variables corresponding to the x and y coordinates of a well are consistently inherited from the same parent. This modification is not needed for the other operators since they do not feature this characteristic of favoring a randomly chosen parent for the computation of each variable of the offspring.

Addition of a bit-flip mutation operator

So far, none of the available operators is designed to modify the number of wells. Therefore, a new mutation operator is created, which either adds a well at a random location or randomly removes one of the wells, with 50% chance of adding a well and 50% chance of removing one. This operator basically plays the same role as the bit-flip mutation operator used in binary representation, which flips the value of the variables with a given probability. It is henceforth called the simulated bit-flip mutation operator.

5.3 Evaluation of the operators

As seen in Section 4.2.2, most of the crossover operators implemented in the BORG framework are likely to be inadequate for the problem considered in this thesis. Hence, a series of runs is carried out with all the operators in order to determine which ones are the most useful. Table 5.1 sums up the parameters chosen for these runs.

Test case	Thirty contaminant sources
Maximum admitted number of monitoring wells	5
Number of calls to the evaluation function	2e5
Resolution for the approximation of the Pareto front (epsilon)	0.01 (corresponds to 1% detection probability or 2 days warning time)
Number of repeated optimization runs	5

Table 5.1: Parameter values for the series of runs designed for the evaluation of the operators

Figure 5.1 shows, along the course of one run of the algorithm, the origin (i.e. which operators produced them) of the last twenty solutions added to the archive. We see that the simulated binary crossover clearly dominates.

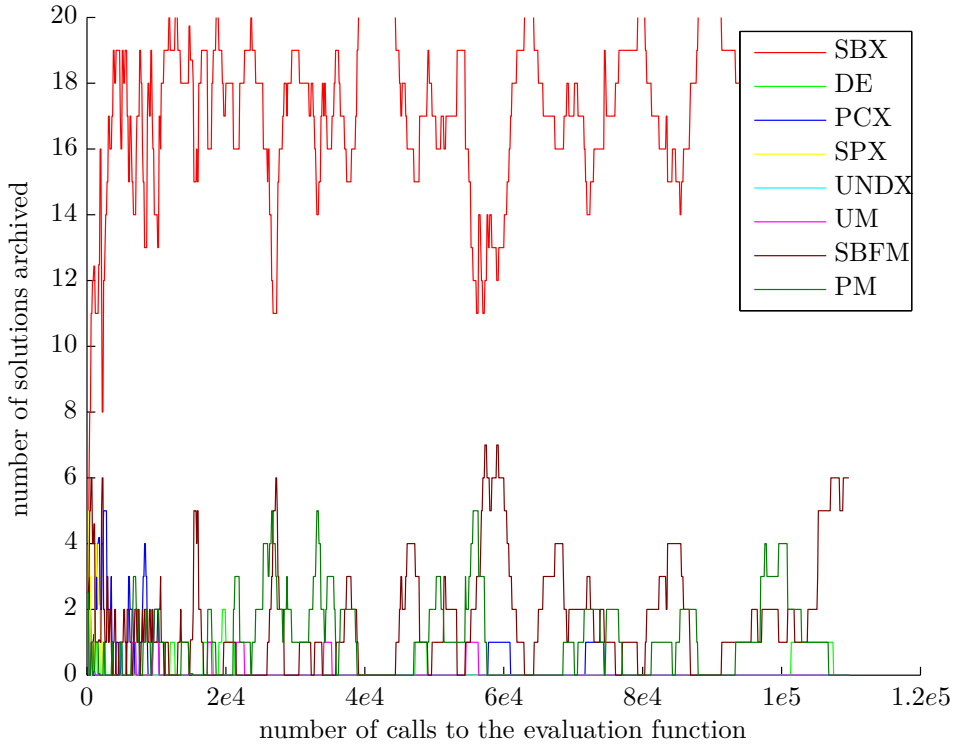


Figure 5.1: Along one run, origin of the last twenty solutions that have been archived. Abbreviation of operators defined in Section 5.2.

Table 5.2 shows the percentage of non-dominated solutions found by each operator. The operators that provide most of the valuable (i.e. non-dominated) solutions are the simulated binary crossover (SBX), the simulated bit-flip mutation (SBFM) and the polynomial mutation (PM). In average, 42% of the solutions present in the archive at the end of a run were found by the simulated binary crossover operator, 26% were found by the bit-flip mutation operator and 13% by the polynomial mutation operator, while the remaining 19% were found by the other operators. It is noticeable that the two most useful operators - SBX and SBFM - are the ones that were adapted from operators used in combinatorial optimization with binary variables (namely the uniform crossover and the bit-flip mutation).

Three series of runs are then performed to compare the performances of three different sets of operators: the first set is composed of SBX and SBFM, the second of SBX, SBFM, PM and PCX, and the third one of all the operators. The parameters are the same as in

% of non-dominated solutions found during the nth quarter of the run	first	second	third	fourth
simulated binary crossover (SBX)	24	8	5	5
differential evolution (DE)	3	0	0	0
parent centric crossover (PCX)	7	0	0	0
simplex crossover (SPX)	3	0	0	0
unimodal normal distribution crossover (UNDX)	2	0	0	0
uniform mutation (UM)	3	0	0	0
simulated bit-flip mutation (SBFM)	14	5	4	3
polynomial mutation (PM)	6	3	2	2

Table 5.2: Respective performances of the operators along a run of the algorithm

Table 5.1, except the number of calls to the evaluation function which is now increased to 1e6. This way the search goes further, in case an operator would prove to be efficient only lately in the run. The results are presented in Table 5.3. As seen in Section 2.5, a large value for the hypervolume indicates good performance, while for the generational distance and the epsilon indicator a low value is seeked.

operators	hypervolume	generational distance	epsilon indicator
SBX and SBFM	1.31e-1	3.8e-4	6.7e-2
SBX, SBFM, PM and PCX	1.31e-1	4.0e-4	6.8e-2
all the operators	1.31e-1	3.3e-4	6.5e-2

Table 5.3: Performances of the BORG algorithm using different sets of operators

As can be seen, the performances are, for these results averaged over five runs, very similar. Runs with four operators give slightly worse results than runs with two operators. As for runs with all the operators, they perform as well as runs with SBX and SBFM for the hypervolume, 15% better for the generational distance and 3% better for the epsilon indicator. Concerning the generational distance, a value of 4e-4 corresponds to 0.04% detection probability or 0.08 day warning time. These results are much lower than the precision of the flow and transport model. Thus, it is concluded that runs with SBX and SBFM (the best two operators) perform as good as runs with more operators. The other operators are therefore dismissed for the sake of simplicity.

5.4 Integration of other algorithms into the BORG framework

In this section, the metaheuristics presented in Chapter 2 are considered in order to find out which of them could be added to the BORG framework with the idea of improving its performance.

Due to its deterministic exploration, Tabu Search is more adapted to a binary or even a discrete representation than to a continuous one. Moreover, the complexity inherent to the method (because of its multiple memory structures) is likely to make its implementation (and, later on, its parameterization) uneasy. Therefore, Tabu Search is dismissed.

Hierarchical Bayesian Optimization models dependencies between the decision variables. It is not applicable to our problem because it hardly fits with a continuous representation of solutions: assuming that a given well location is present in several solutions, the coordinates of this well are likely to be coded in different variables. For example, this well could be the first well of a solution, and at the same time the third well of another solution. Therefore, the decision variables would have very low correlation between each other, even if there is in fact a meaningful dependence between the individual locations. For this reason, Hierarchical Bayesian Optimization is also set aside.

Particle Swarm Optimization requires a whole population of solutions as input to produce only one solution. The solutions belonging to this population should feature the same number of wells. The main problem concerning the application of this algorithm to our problem stems once again from the chosen representation: the wells coded first in different solutions may be spread all over the domain. Therefore, it is unlikely that Particle Swarm Optimization applied to an initial population of good solutions would lead to better results than if the initial population is composed of random solutions. For this reason, it does not seem to be an interesting option.

Ant Colony Optimization and Simulated Annealing seem more promising. In the following subsections, we discuss their implementation as operators of the BORG algorithm. This integration allows to associate the characteristics of these algorithms with the multiobjective components of BORG.

5.4.1 Crossover operator based of an ant colony algorithm

The ant colony algorithm designed by Li and Chan Hilton (2007) optimizes the removal of wells in a monitoring scheme. We implement it as a crossover operator within the BORG framework in the following way : the wells of two parent solutions are combined together. Half of the wells are then removed with the help of the ant colony algorithm to form a child-solution. The two control parameters are the number of ants and the number of iterations. To increase these numbers improves the quality of the crossover but also increases the computational cost.

Many runs are performed with various combinations of parameters, with the number of ants varying between one and five and number of iterations between one and thirty. It appears that, when the probabilities for all operators to be chosen are proportional to the number of non-dominated solutions they produce, then the Ant Colony Crossover (ATX) is more used than SBX and SBFM. However, the computational cost of ATX needs to be taken into account by adjusting its selection probability so that it is inversely proportional to the number of calls to the evaluation function. Henceforth, the selection probability of ATX proves to be so low that the algorithm never reaches the point where the efficiency of ATX would outweigh its computational cost.

5.4.2 Mutation operator based on a simulated annealing algorithm

As has been seen in Chapter 2, simulated annealing is single-solution based. Since this heuristic takes one solution as an input and transforms it step-by-step into a better solution, it can be integrated into the BORG genetic algorithm as a mutation operator. The control parameters are the cooling schedule and the number of iterations. A slower cooling schedule linked with a higher number of iterations improves the quality of the resulting solution at the expense of speed.

In the same way as with the ant colony algorithm, many runs are performed with different numbers of iterations and different starting temperatures. Again, when taking into account the computational cost for the calculation of the selection probability, the efficiency of the new operator proved to be too low - even at the end of the run - for it to be useful.

The reason why this attempt to integrate more sophisticated algorithms within the BORG framework eventually failed may be that in our problem the dependencies between potential monitoring well locations are not complex enough for such algorithms to be efficient. The known information concerning the exemplary drinking water catchment is limited: only the boundary conditions are defined. Therefore, although each of the Monte-Carlo simulations of the hydraulic conductivity field may feature complex patterns, by averaging over a high number of realizations these complexities cancel out. Moreover, the boundary conditions are simple: constant head at the left and right side of the domain and no flow at the top and bottom. Coupled with the simplicity of the averaged conductivity field, this implies a simplicity of the groundwater velocity field. Consequently, it is assumed that the conclusions drawn out regarding the inefficiency of the ant colony algorithm and simulated annealing operators can be extended to all cases where the known information about the problem does not comprise explicit knowledge about complex patterns of the water circulation in the underground.

5.5 An alternative to BORG: the NSGA-II algorithm

In this section, an alternative to the BORG algorithm is considered. The aim is to find an algorithm available as a MATLAB code that would provide performances close to the performances of BORG. A MATLAB code would provide the advantage of having both the flow and transport model and the optimization program in the same language. This would be more user-friendly, for example in view of commercial applications. Moreover, the MATLAB code considered in the following can be freely used, modified and distributed, which is not the case of the BORG algorithm.

An implementation of the NSGA-II algorithm (introduced in Chapter 2) is available in MATLAB (Seshadri, 2009). This genetic algorithm features two operators: simulated binary crossover and polynomial mutation. The simulated binary crossover is modified in the same way as it has been done for BORG (see Section 5.2), and the polynomial mutation is replaced by the simulated bit-flip mutation introduced in Section 5.2. Results for this algorithm are presented in Section 6.3.

6 Results and discussion

This chapter presents results obtained with series of runs performed for the three test cases and two algorithms: the BORG algorithm (implemented in C) and the NSGA-II algorithm (implemented in Matlab). Section 6.1 to Section 6.2.2 present results obtained with BORG for the three test cases, while Section 6.3 presents results obtained with the NSGA-II algorithm.

6.1 Performance assessment of the BORG algorithm

To assess the performances of the BORG algorithm, six series of runs are carried out with the parameters described in Table 6.1.

In the first three series of runs, corresponding to TC1, TC5 and TC30 respectively, a maximum of five monitoring wells is admitted. Henceforth, these three series of runs will be respectively referred to as TC1-5, TC5-5 and TC30-5. The algorithm stops after one million calls to the evaluation function. Fifty optimization runs are carried out for each of the test cases described in Section 3.2.

In the last three series of runs, again corresponding to TC1, TC5 and TC30 respectively, a maximum of twenty monitoring wells is admitted. The algorithm stops after five million calls to the evaluation function. This higher number of calls to the evaluation function is chosen because it is expected that due to the higher number of admitted monitoring wells, a longer run is necessary to obtain good results. Thirty optimization runs are carried out for each of the test cases described in Section 3.2. This lower number of optimization runs is chosen due to an increase in computational cost, as depicted in Section 6.1.1. These series of runs will be used to compare the behavior of BORG when switching from up to five allowed monitoring wells to up to twenty.

(a) TC1-5

Test case	One contaminant source
Maximum admitted number of monitoring wells	5
Number of calls to the evaluation function	1e6
Resolution for the approximation of the Pareto front (epsilon)	0.01 (corresponds to 1% detection probability or 2 days warning time)
Number of repeated optimization runs	50

(b) TC5-5

Test case	Five contaminant sources
Maximum admitted number of monitoring wells	5
Number of calls to the evaluation function	1e6
Resolution for the approximation of the Pareto front (epsilon)	0.01
Number of repeated optimization runs	50

(c) TC30-5

Test case	Thirty contaminant sources
Maximum admitted number of monitoring wells	5
Number of calls to the evaluation function	1e6
Resolution for the approximation of the Pareto front (epsilon)	0.01
Number of repeated optimization runs	50

(d) TC1-20

Test case	One contaminant source
Maximum admitted number of monitoring wells	20
Number of calls to the evaluation function	5e6
Resolution for the approximation of the Pareto front (epsilon)	0.01
Number of repeated optimization runs	30

(e) TC5-20

Test case	Five contaminant sources
Maximum admitted number of monitoring wells	20
Number of calls to the evaluation function	5e6
Resolution for the approximation of the Pareto front (epsilon)	0.01
Number of repeated optimization runs	30

Table 6.1: Parameter values for the series of runs designed for a performance assessment of the BORG algorithm

(f) TC30-20

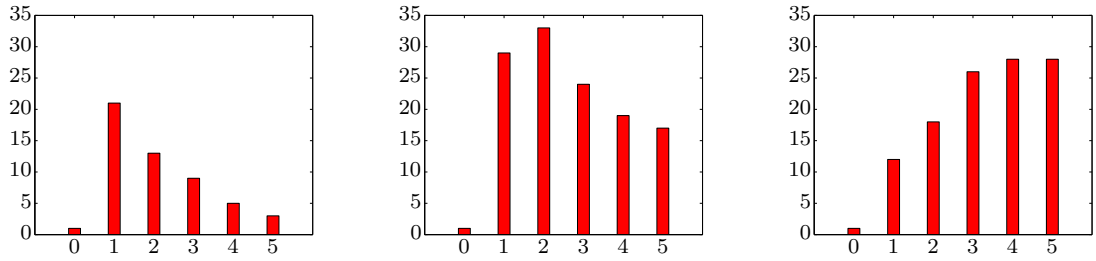
Test case	Thirty contaminant sources
Maximum admitted number of monitoring wells	20
Number of calls to the evaluation function	5e6
Resolution for the approximation of the Pareto front (epsilon)	0.01
Number of repeated optimization runs	30

Table 6.1: Parameter values for the series of runs designed for a performance assessment of the BORG algorithm

6.1.1 Distribution of the results

Figure 6.1 shows for one run of each series the distribution of the solutions produced according to their number of wells. We can see that, with an increasing number of contaminations (from the one-source test case to the five-source test case, and from the five-source test case to the thirty-source test case), the number of Pareto-optimal solutions produced increases, as well as the average number of wells per solution. The increase of the number of contamination sources also corresponds to an increasing dispersal of the contaminated locations, as shown in Section 3.2.2. In combination, this leads to the necessity to install more monitoring wells for comparable values of detection probability and utility warning time.

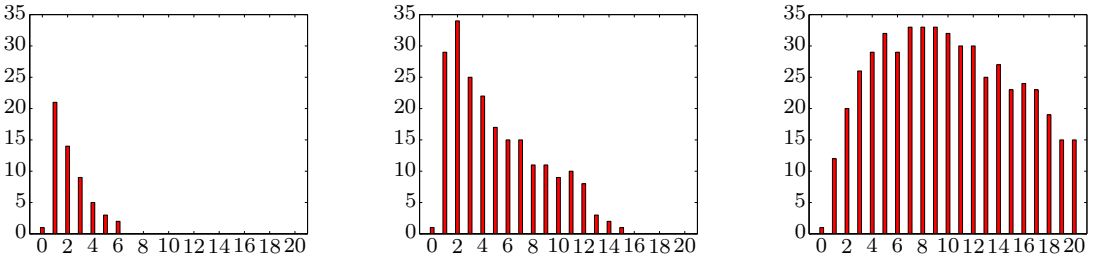
One single run lasts 5 minutes for TC1-5 on a standard computer, while it lasts 1.6 times longer for TC30-5. One single run of TC1-20 lasts 25 minutes, while it lasts 3 times longer for TC30-20. The fivefold increase in running time from TC1-5 to TC1-20 is due to the fivefold increase of the number of calls to the evaluation function. The 160% and 300% increases from TC1-5 to TC30-5 and from TC1-20 to TC30-20 respectively are due to an increase in the number of wells per solution, which is clearly more pronounced in the case where up to 20 monitoring wells are admitted than in the case where the number of monitoring wells per solution cannot exceed 5.



(a) One-source test case, up to 5 monitoring wells. 52 solutions produced

(b) Five-source test case, up to 5 monitoring wells. 123 solutions produced

(c) Thirty-source test case, up to 5 monitoring wells. 113 solutions produced



(d) One-source test case, up to 20 monitoring wells. 55 solutions produced

(e) Five-source test case, up to 20 monitoring wells. 213 solutions produced

(f) Thirty-source test case, up to 20 monitoring wells. 511 solutions produced

Figure 6.1: Distribution of the solutions produced by a run with respect to the the number of monitoring wells they feature (x-axis: number of monitoring wells; y-axis: number of solutions)

6.1.2 Performance metrics

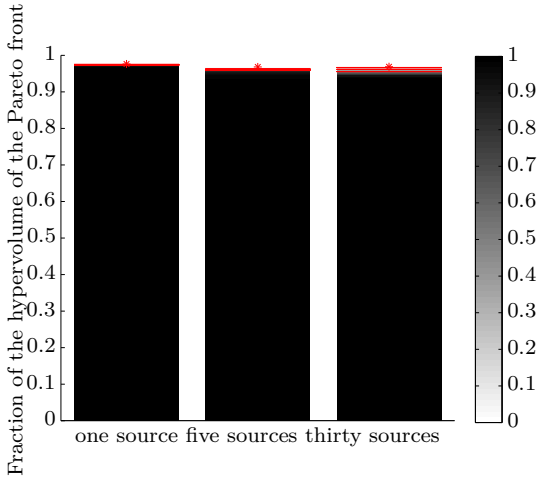
The performance metrics for the first three series of runs are presented in Figure 6.2. The metrics for the last three series of runs are presented in Figure 6.3. The Hypervolume metric should be maximized, while Generational distance and Epsilon distance should be minimized (see Section 2.5). Hypervolume is represented as a fraction of the hypervolume of the Pareto front (obtained by combining the results of all runs), while for the Generational distance and the Epsilon indicator the value 0.1 corresponds to 1% detection probability or 2 days warning time.

6.1.2.1 Runs where up to five monitoring wells are considered

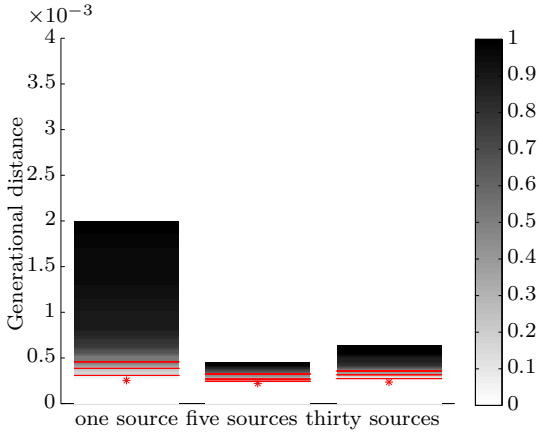
The results are described in the first paragraph of this Section, and discussed in the second one.

Figure 6.2 presents the results when up to five monitoring wells are considered. Figure 6.2a shows that, as far as the hypervolume is concerned, the performances are consistently good for all three test cases. Indeed, it can be seen that every run leads to a higher value for the hypervolume than 95% of the value obtained for the reference set (obtained, as seen in Section 2.5, by aggregating the sets of solutions obtained for all runs). On Figure 6.2b we see that for all three test cases the best value that can be reached for the average distance to the Pareto front is about the same (around 3e-4). The performances are similar for TC5 and TC30, and slightly worse for TC1, which features a higher probability of getting a lower value. However, Figure 6.2c shows that the results for TC1 are the best with regards to the epsilon indicator, with a nearly constant value (close to 0.02). The values obtained for TC5 and TC30 are more sparse and the smaller values obtained for these test cases are higher than all the values obtained for TC1.

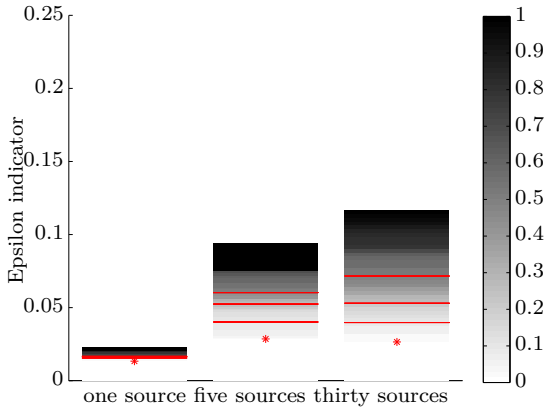
On Figure 6.1 it has been seen that the number of solutions produced is twice larger for TC5-5 and TC30-5 than for TC1-5. Moreover, we know that the algorithm maintains the population proportional to the size of the archive (which stores the non-dominated solutions). Therefore, along the run, the population of solutions must be about twice smaller for TC1-5 than it is for the other test cases and, since the number of calls to the evaluation function is the same for all test cases, the number of generations performed



(a) Hypervolume



(b) Generational distance



(c) Epsilon indicator

Figure 6.2: Performance metrics, up to 5 monitoring wells, BORG algorithm. The graded shades of gray correspond to the probability of reaching the corresponding value. The red star corresponds to the best value reached and red lines mark 50%, 75% and 90% of attainment probability.

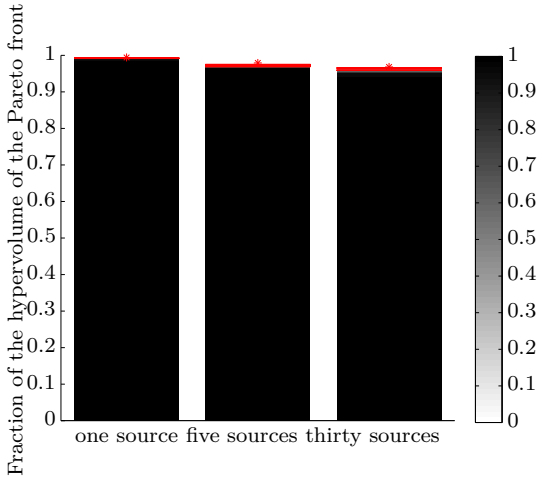
during a run is twice larger for TC1-5 than for TC5-5 and TC30-5. This higher number of generations coupled to a smaller (in average) number of monitoring wells per solution could lead to a more complete crossover within the population, hence the smaller spread within one test case observed on Figure 6.2c for TC1-5. As regards the poorer proximity observed for TC1-5 on Figure 6.2b, a smaller population and a smaller number of wells per solution imply a smaller number of potential locations, i.e. a sparser set of potential locations. This could result in a less accomplished intensification process than for the five and thirty-source test cases.

6.1.2.2 Runs where up to twenty monitoring wells are considered

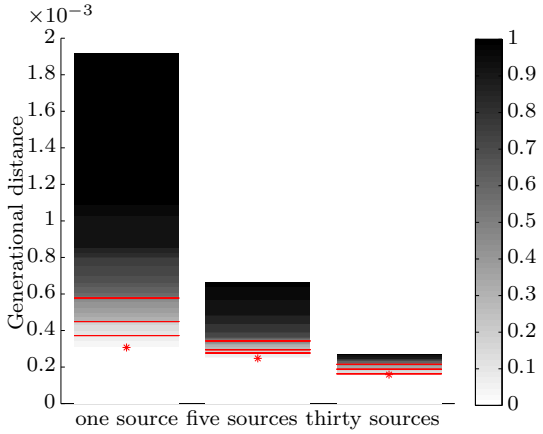
Figure 6.3 presents the results when up to twenty monitoring wells are possible. As can be seen on Figure 6.3a, the performances with respect to hypervolumes are, as well as in Section 6.1.2.1 where up to five monitoring wells were considered, very good for all three test cases. Figure 6.3b shows that the performance regarding the generational distance increases with an increasing number of sources of pollution (which corresponds, in our case, to an increasing dispersal of the sources of pollution). As explained in the previous paragraph, this could be caused by the subsequent increase in the size of the population generated by the algorithm. The algorithm maintains the population proportional to the number of solutions present in the archive and, as has been seen in Figure 6.1, for the series of runs with up to twenty monitoring wells the number of solutions produced notably increases from TC1 over TC5 to TC30. Concerning the epsilon indicator, Figure 6.3c shows that, similarly to what has been seen in Figure 6.2, runs for TC1 produce better and more consistent results than runs for TC5 and TC30. It can be noticed that runs with up to twenty monitoring wells produce better results for the epsilon indicator than runs with up to five monitoring wells. This is likely to be due to the higher number of calls to the evaluation function, which ensures a better mixing of the population in spite of the larger population size.

6.1.2.3 Acceptability of the results with regard to the nature of the problem

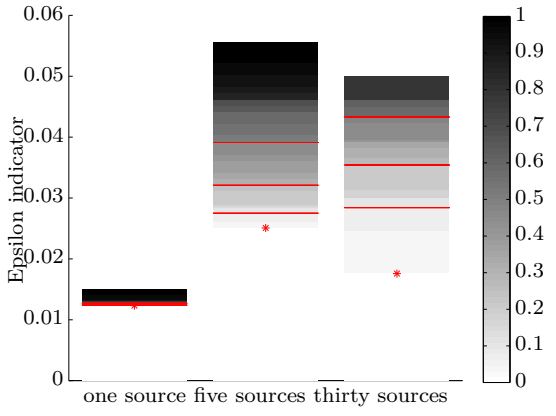
Beyond the relative performances between the different series of runs, the question arises whether these performances are acceptable regarding the nature of the problem. Looking



(a) Hypervolume



(b) Generational distance



(c) Epsilon indicator

Figure 6.3: Performance metrics, up to 20 monitoring wells, BORG algorithm. The graded shades of gray correspond to the probability of reaching the corresponding value. The red star corresponds to the best value reached and red lines mark 50%, 75% and 90% of attainment probability.

at Figure 6.3, we see that for all test cases, at least 25% of the runs feature a generational distance lower (i.e, better) than 5e-4 and an epsilon indicator lower (i.e., better) than 0.04.The acceptability of theses values is discussed further.

As stated in Table 6.1, a value of 0.01 corresponds to 1% detection probability or 2 days warning time. Therefore, considering the solutions produced by the 25% best runs, the performance gap between any produced solution and the closest point of the Pareto front corresponds in average to less than 0.05% of detection probability or 0.1 day of warning time. 0.02% of detection probability as well as 0.04 day (1 hour) are much lower than the precision of the model. Thus, asking for a better performance in the generational distance aspect is not necessary. This is different for the epsilon indicator. The distance from the worst solution to the Pareto front corresponds to less than 4% of detection probability or 8 days of warning time. One could argue that errors about 4% of detection probability and 8 days of warning time cannot be overlooked. The obtained results suggest that the consistency can be increased (i.e. the epsilon indicator decreased) by increasing the number of generations. Figure 6.2c shows that consistent results for the epsilon indicator are obtained for the one-source test case with 1e6 calls to the evaluation function. These runs with TC1-5 produce in average 50 solutions. To obtain similar results with TC5-5 and TC30-5, for which the runs produce more solutions which feature a higher number of wells, it seems necessary to have a higher number of calls to the evaluation function. On the other hand, to improve the metrics for the one-source test case, it is advised to increase the population. This can be achieved either by diminishing epsilon or by modifying the parameter which controls the ratio population/size of the archive.

6.1.2.4 Dynamics of the performance metrics

Figure 6.4 shows the evolution of the metrics along a run. The one, three and thirty-well test cases are considered with up to five monitoring wells. Each curve is obtained by averaging three runs. Hypervolume curves are normed to start with a value of 0 and end with a value of 1, while the curves for the generational distance and the epsilon indicator start with a value of 1 and end at 0. It can be observed that the hypervolume is the metric that converges the fastest. Under the realistic assumption that each single run of the algorithm converges towards an optimal set of solutions, a fast convergence is a positive thing. Concerning the generational distance and the epsilon indicator, they

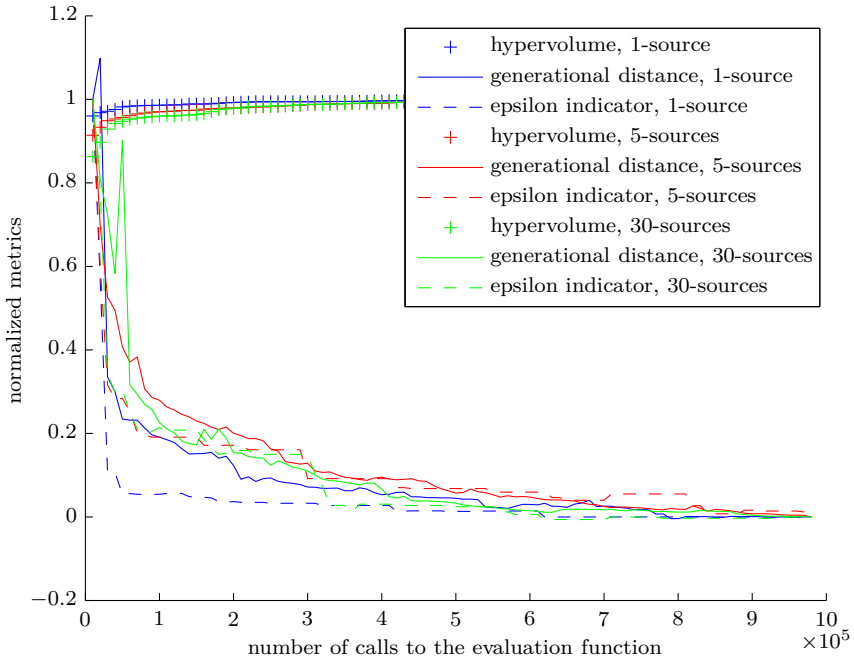


Figure 6.4: Dynamics of the performance metrics along a run. All metrics are normed such that hypervolumes start at 0 and end at 1, and generational distances and epsilon indicators start at 1 and end at 0.

converge at a comparable speed for all three test cases, apart from the epsilon indicator which converges much faster in the case of one source of contamination. The latter effect is consistent with what has been observed on Figure 6.2. This faster convergence is due to the fact that for TC1-5, one generation is completed after a twice lower number of calls to the evaluation function than for TC5-5 and TC30-5.

6.2 Results for the test cases

6.2.1 Results for the one-source test case (TC1)

Detection probability map

Figure 6.5 shows the detection probability map: the detection probability is displayed as a function of the location of one single monitoring well. The detection probability at a given location corresponds to the fraction of Monte-Carlo realizations for which the computed plume is detected at this location. The map is symmetrical and regular, what confirms that the complexities of the plumes obtained for single Monte-Carlo realizations (see Figure 3.3) average out with a high number of realizations. From the contamination source located at (100, 250), by increasing X_1, the detection probability increases until it reaches the value 76% at the location (190, 250). It then decreases until, close to the pumping well, it reaches a value close to zero. At the pumping well, located at (400, 250), the detection probability is 100%.

The line $X_2 = 250$ is considered (line that passes through the contamination source and the pumping well). Indeed, the lower X_1 (i.e., the further from the pumping well), the higher the average warning time. For $X_1 \in [190; 400[$, the lower X_1, the higher the detection probability. By lowering X_1, the performance is increased regarding both objectives: there is no trade-off. However, for $X_1 \in [100; 190]$, the lower X_1, the lower the detection probability. There is a trade-off between the two objectives: an increase of the warning time can only be achieved at the cost of a diminution of the detection probability. Two possibilities are thus available for the decision maker. The first one, in case only the detection probability is considered important, is to monitor at the location of the pumping

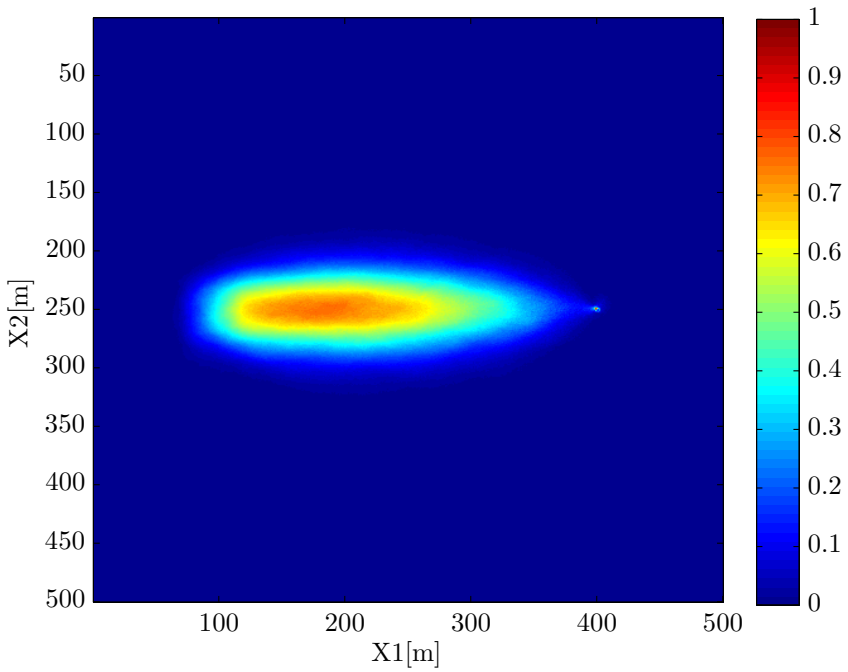


Figure 6.5: TC1: detection probability with one monitoring well

well. The second one is to locate the monitoring well at $X_2 = 250; X_1 \in [100; 190]$, according to the relative importance attributed to the two objectives.

The peak in the detection probability is not achieved at the location of the contamination source due to the incertitude regarding this location. As has been seen in Figure 3.3, from the contamination source to the pumping well, the width of the plume increases first, then decreases. This increase, along with the convergence of streamlines towards the pumping well, account for the increase of the detection probability observed for $X_1 \in [100; 190]$. The subsequent decrease of the plume width accounts for the observed decrease in the detection probability.

Pareto front

The Pareto front obtained with one single run with up to five monitoring wells is displayed Figure 6.6.

It can be seen that with two monitoring wells, it is possible to achieve an average warning

time of 50 days with a detection probability of 90%. With three monitoring wells, we can have the same warning time while enhancing the detection probability up to 98%. The graph shows that beyond three monitoring wells, the addition of a monitoring well does not lead to significant improvement. This could result from the fact that three monitoring wells suffice to offer performance close to the optimum, and will be further investigated in the next paragraph. Here the term optimum refers to the case where all occurrences of the contamination can be detected at the source point.

Gaps can be observed in the Pareto front. For example, with one monitoring well, a detection probability of 76% associated to an average warning time of 25 days can be achieved, but from this point it is not possible to trade warning time for a higher detection probability, unless if we monitor at the pumping well, what corresponds to a detection probability of 100% and a null warning time. This gap corresponds to the no-trade-off zone that has been observed on the detection probability map.

As can be seen on the graph, the warning-time is constrained between, on the one hand, this gap, and on the other hand, the abovementioned optimum, in a way such that the objectives are not very much competing.

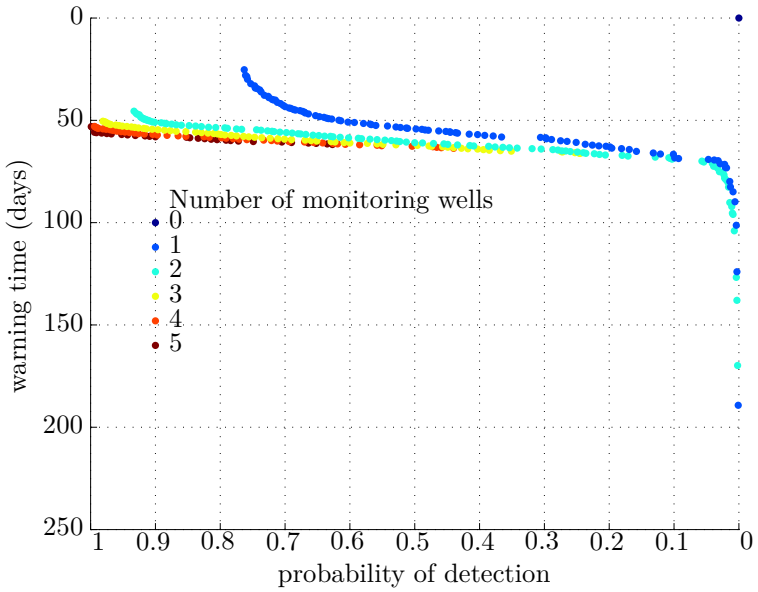
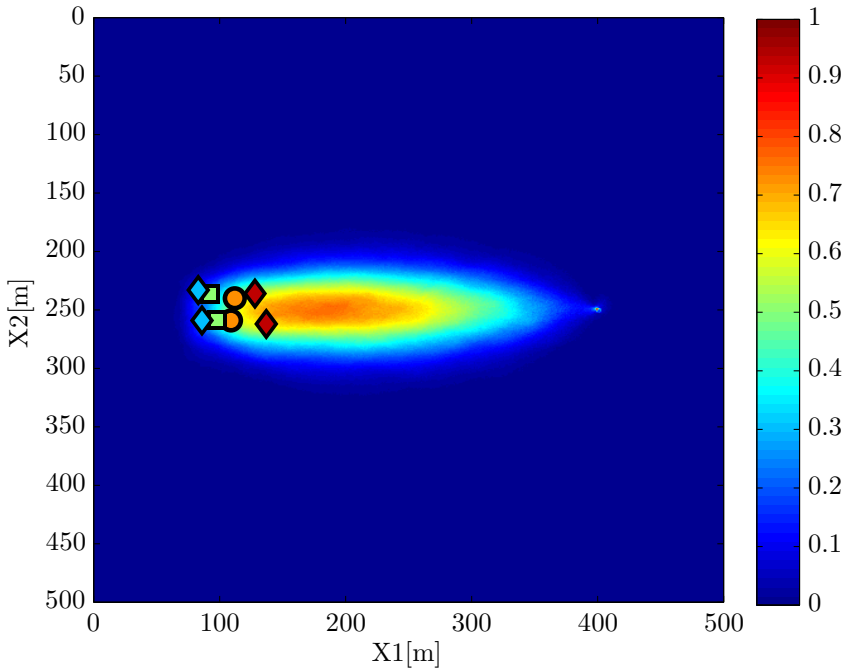
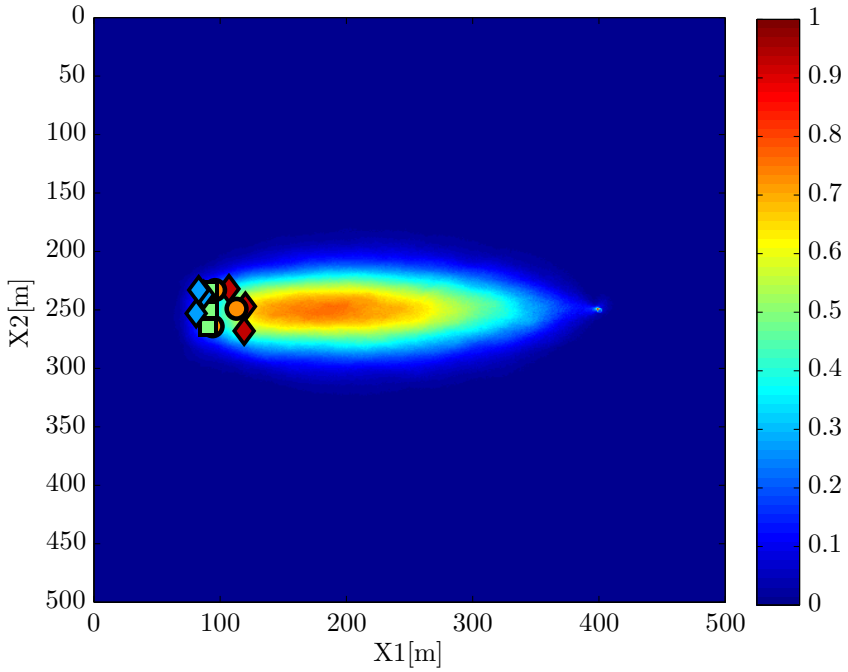


Figure 6.6: TC1: Pareto front with up to five monitoring wells



(a) with two monitoring wells



(b) with three monitoring wells

Figure 6.7: TC1: A few solutions from the Pareto front. The background color corresponds to the detection probability for a single monitoring well. The color of the symbols refers to the detection probability of the monitoring scheme.

Solutions from the Pareto front

Figure 6.7 shows, for different numbers of monitoring wells, a few of the solutions obtained with the optimization algorithm. The colors of the symbols correspond to the detection probability.

In accordance with the Pareto front, it can be seen on Figure 6.7a that a scheme with two monitoring wells allows a higher detection probability than a scheme with a single well (93% vs. 76%). This highest-detection-probability scheme with two wells is located about 50m to the left of the highest-detection-probability scheme with one well. The Pareto front indicates that the warning time associated with the two-well scheme is distinctly higher than the one associated with the single-well scheme. This is consistent with the fact that the closer the scheme is to the contaminant source, the higher the associated warning time.

On Figure 6.7b, it can be seen that, with equal detection probabilities, schemes with three monitoring wells are located slightly more to the left than schemes with two monitoring wells. This shows that a small extra warning time is obtained, at the cost of an additional monitoring well.

6.2.2 Results for the five-source test case (TC5)

Detection probability map

Figure 6.8 shows the detection probability map. The five contaminant sources are centered on (100, 150), (100, 200), (100, 250), (100, 300) and (100, 350). The pumping well is located at (400, 250). Due to the spreading of the contaminant sources, the detection probability for a single monitoring well is much lower than for TC1.

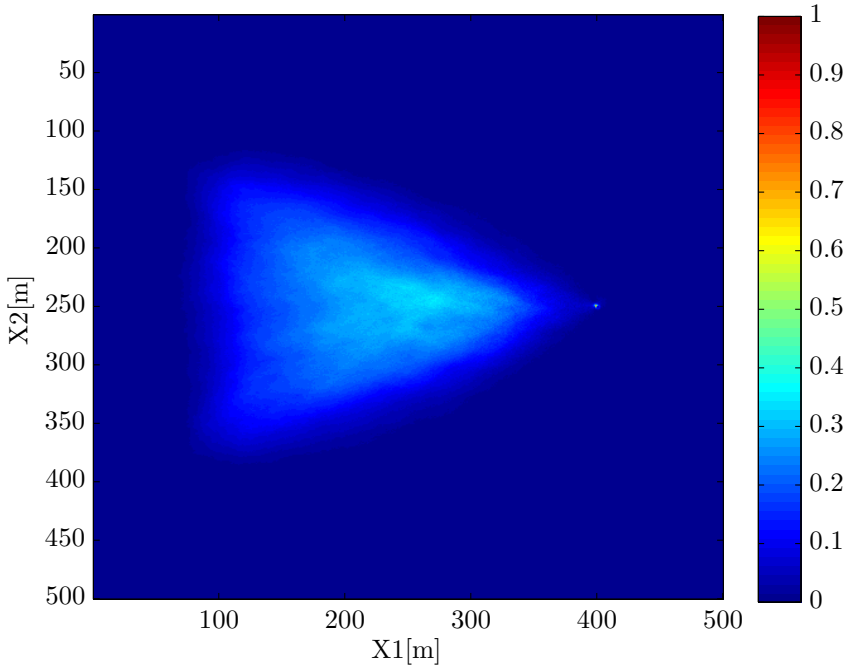


Figure 6.8: TC5: detection probability with one monitoring well

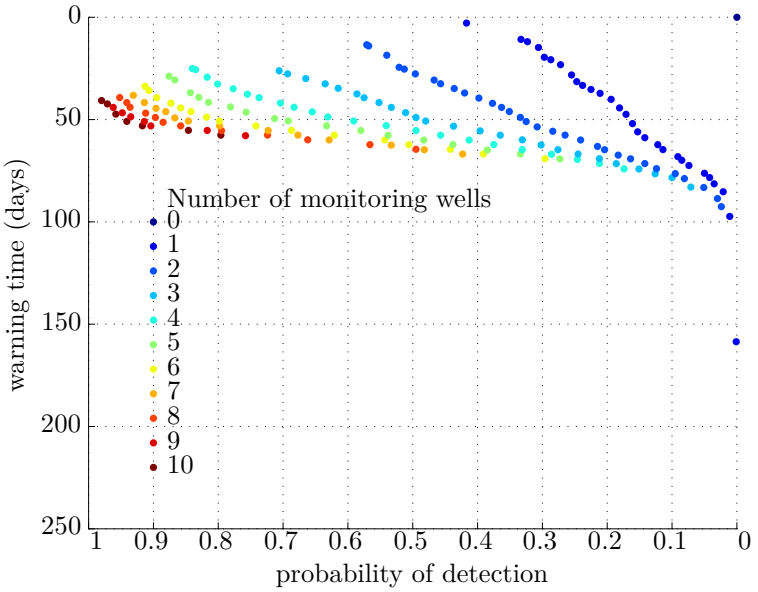


Figure 6.9: TC5: Pareto front with up to ten monitoring wells

Pareto front

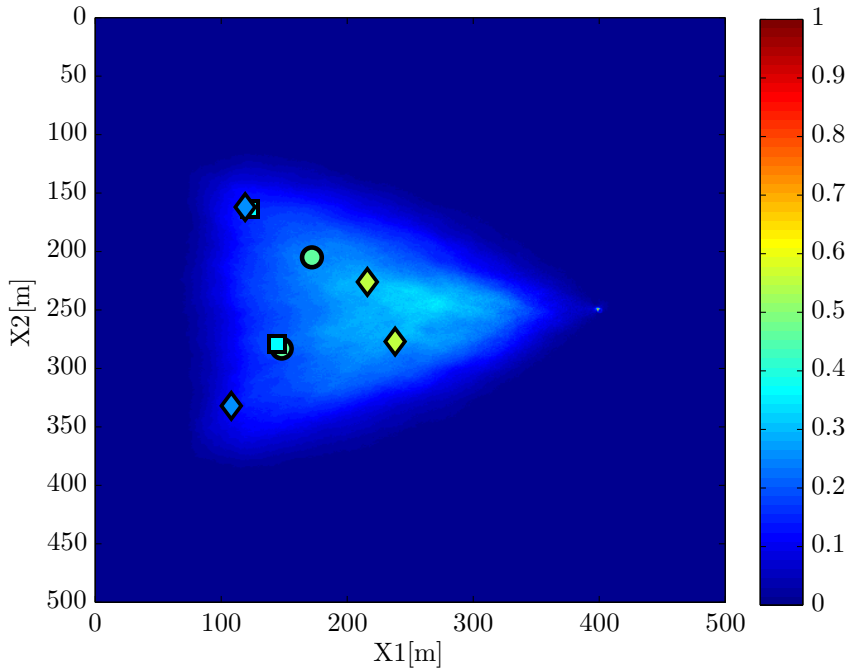
The Pareto front obtained with one single run with up to ten monitoring wells is displayed Figure 6.9.

With five monitoring wells, an average warning time of 50 days is achieved with a detection probability of 70%. With eight monitoring wells, we can have the same warning time while enhancing the detection probability up to 90%. This level of performance could be reached with two monitoring wells for TC1; here, a higher number of monitoring wells is necessary due to the spreading of the contaminant sources.

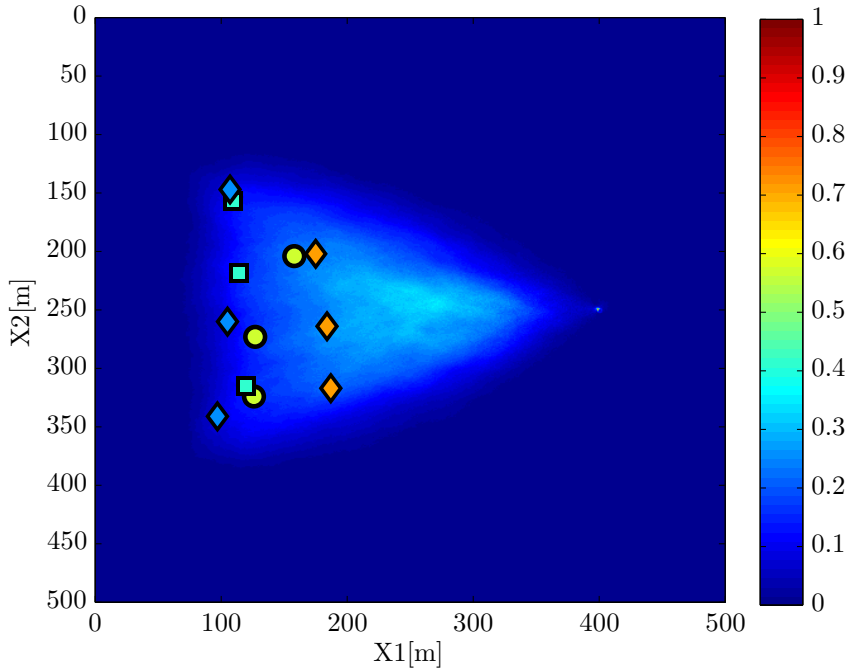
The sections of the Pareto front that correspond to the different numbers of monitoring wells are further from each other than for TC1. This indicates that the benefit obtained by the addition of a monitoring well is more significant. These sections also cover a wider range of warning times: this indicates that the trade-off between the probability of detection and the warning time is more significant than for TC1. Thus, the objectives are more competing here than for the one-source test case.

Solutions from the Pareto front

Figure 6.10 shows, for different numbers of monitoring wells, a few of the solutions obtained with the optimization algorithm. The colors of the symbols correspond to the detection probability. Once again, the trade-offs between the number of monitoring wells, the detection probability and the warning time can be observed. An increase of the number of monitoring wells allows an higher average warning time and a higher detection probability. A higher warning time can be obtained at the cost of a decrease of the detection probability. These observation are more distinct than for the one-source test case due to tougher trade-offs.

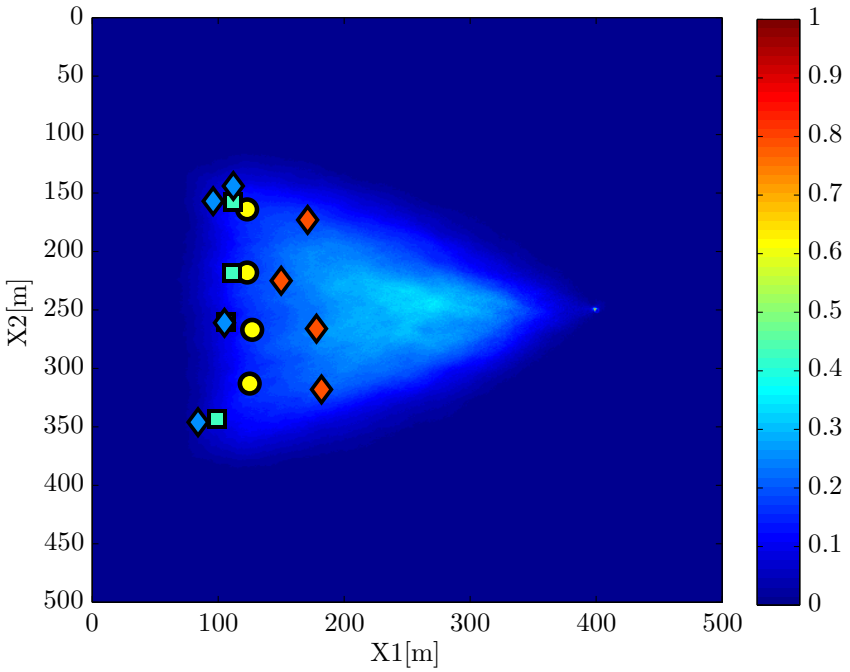


(a) with two monitoring wells

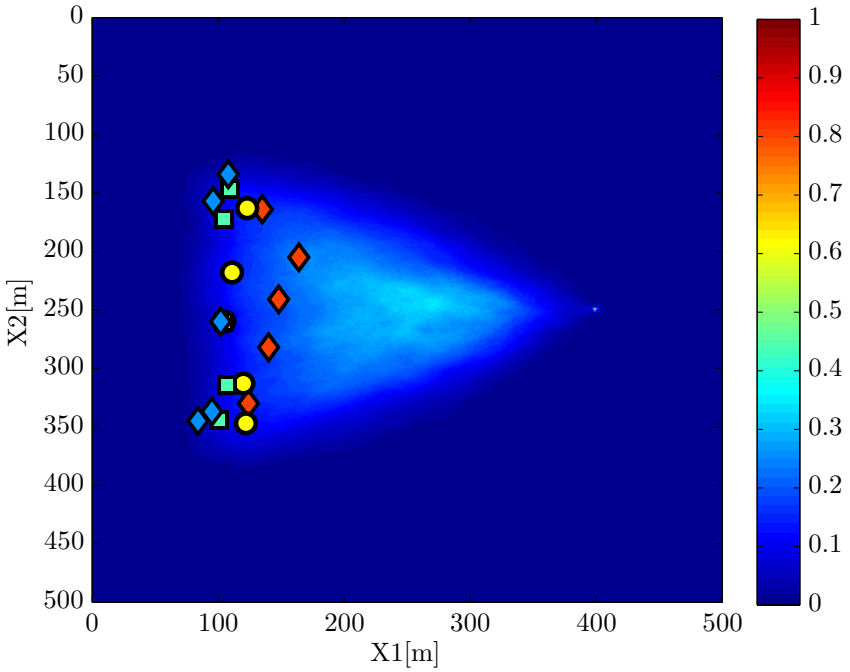


(b) with three monitoring wells

Figure 6.10: TC5: A few solutions from the Pareto front. The background color corresponds to the detection probability for a single monitoring well. The color of the symbols refers to the detection probability of the monitoring scheme.



(c) with four monitoring wells



(d) with five monitoring wells

Figure 6.10: TC5: A few solutions from the Pareto front. The background color corresponds to the detection probability for a single monitoring well. The color of the symbols refers to the detection probability of the monitoring scheme.

6.3 Performance assessment of the NSGA-II algorithm

To assess the performance of the NSGA-II algorithm, three series of runs are carried out with the parameters described in Table 6.2. These series correspond to the first three series of runs designed to evaluate the BORG algorithm. The parameters need to match so that the results can be compared. For the NSGA-II algorithm, the population and the number of generations are specified instead of the resolution of the Pareto front and the number of calls to the evaluation function for the BORG algorithm. For TC1-5, it has been seen that the BORG algorithm produces an average of 50 solutions. With the ratio population/size of the archive being set to four, this corresponds to a population of 200 individuals. Concerning the number of generations, the NSGA-II algorithm produces (and evaluates) at each generation approximately as many children-solution as the number of individuals in the population. Therefore, for a population for 200 individuals, the number of generations corresponding to one million calls to the evaluation function is 5000. For TC5-5 and TC30-5, it has been seen that the BORG algorithm produces close to 100 solutions. This corresponds to 400 individuals and 2500 generations.

The results concerning the performance metrics are presented on Figure 6.11. As can be seen, the hypervolume is above 95% for all test cases. Concerning generational distance, in all three cases, 50% of the runs produce a generational distance smaller than 6e-4. These results are similar to the ones obtained with the BORG algorithm (see Figure 6.2). As for the epsilon indicator, 50% of the runs for the one, five and thirty-source test cases respectively produce values smaller than 0.03, 0.09 and 0.08. These values are to compare to the values of 0.017, 0.06 and 0.07 obtained with the BORG algorithm. The values obtained with the NSGA-II algorithm are between 10 and 80% higher than the ones obtained with the BORG algorithm. Thus, over the three performance metrics, two are comparable for the BORG algorithm and the NSGA-II algorithm while the third one is up to one order of magnitude higher (i.e, worse) for the NSGA-II algorithm. It is expected that the performance for this metrics can be increased by increasing the number of calls to the evaluation function.

As mentioned above, while the BORG algorithm automatically adapts the size of the population to the size of the Pareto front, the NSGA-II requires it as an input parameter. Here, the value was already known, since TC1-5, TC5-5 and TC30-5 had been previously run with the BORG algorithm. To be able to use the NSGA-II algorithm instead of

(a) TC1-5

Test case	One contaminant source
Maximum admitted number of monitoring wells	5
Number of generations	5000
Population	200
Number of repeated optimization runs	50

(b) TC5-5

Test case	Five contaminant sources
Maximum admitted number of monitoring wells	5
Number of generations	2500
Population	400
Number of repeated optimization runs	50

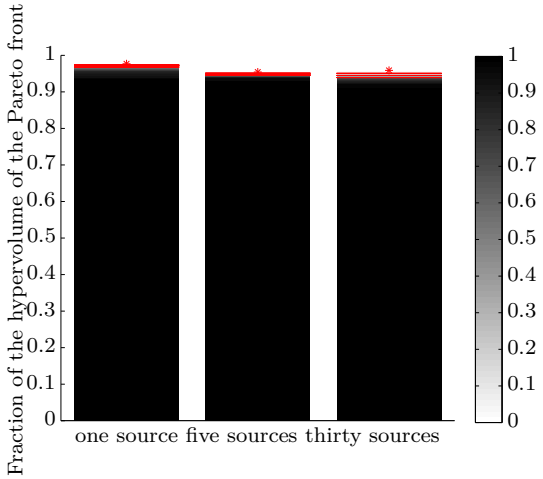
(c) TC30-5

Test case	Thirty contaminant sources
Maximum admitted number of monitoring wells	5
Number of generations	2500
Population	400
Number of repeated optimization runs	50

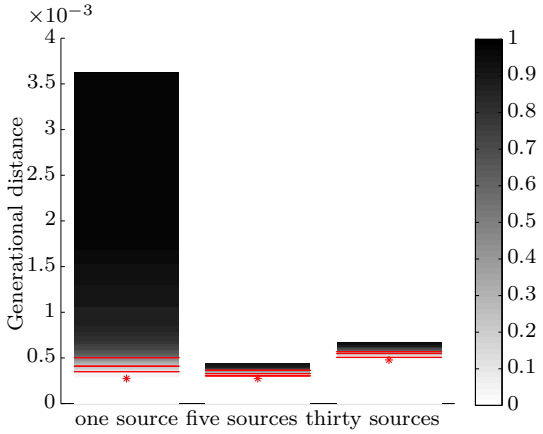
Table 6.2: Parameter values for the series of runs designed for a performance assessment of the NSGA-II algorithm

the BORG algorithm, there are two possibilities. The first one is to set up a method to estimate the size of the Pareto front before running the algorithm. This estimation would be based on key characteristics of the water catchment and the distribution of the contaminations sources. The second possibility is to modify the algorithm to make the size of the population dynamic, so that it adapts to the size of the Pareto front. This would require to implement an archive in the algorithm. This second possibility, not based on an estimation by nature prone to error, seems to be a priori more desirable than the first one.

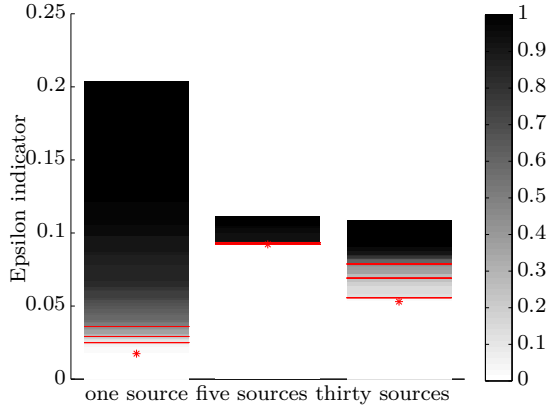
To conclude, the results obtained for TC1-5, TC5-5 and TC30-5 with the NSGA-II algorithm are promising, under two conditions. Firstly, that the performance concerning the epsilon indicator can be enhanced. Secondly, that a way is found for the size of the population to be adapted to the parameters of the problem.



(a) Hypervolume



(b) Generational distance



(c) Epsilon indicator

Figure 6.11: Performance metrics, up to 5 monitoring wells, NSGA-II algorithm. The graded shades of gray correspond to the probability of reaching the corresponding value. The red star corresponds to the best value reached and red lines mark 50%, 75% and 90% of attainment probability.

7 Summary and conclusion

This master's thesis considered the possibility to implement early-warning systems in well catchment areas. Existing methods to prevent contaminants from reaching the water distribution network comprise monitoring at the location of the well and preventive measures in the well catchment. Early warning monitoring systems in well catchments, featuring high probabilities to detect contamination events as well as an early detection of these contaminations, would allow more efficient mitigation measures to be taken.

The goal of this thesis was to set up a method to design optimal early-warning systems for well catchments, while having knowledge about the potential contamination spots, with respect to three conflicting objectives: (1) maximizing the probability of detection of contaminations, (2) maximizing the early-warning time, i.e. the remaining time between detection and arrival at the production well, and (3) minimizing the cost of sampling the system. A numerical flow and transport model was used to provide an exemplary well catchment. Advanced search algorithms (called metaheuristics) were used to carry out the multi-objective optimization. After a selection of the most promising algorithms, these were evaluated on three synthetic application scenarii (or test cases) with the help of performance indicators. The test cases featured different numbers of possible pollution sources. The performance indicators were used to evaluate the sets of solutions produced by runs of the algorithms with regard to three criteria: (1) proximity (to the optimum), (2) consistency (with regard to the optimum) and (3) diversity in the objective space.

The problem that was considered in this thesis differs from usual monitoring problems in that the potential locations for monitoring wells, from which optimal combinations shall be selected, form a fine meshing of the study area. Due to the great number of potential locations, the usual binary representation of the solutions was excluded. Instead, the monitoring wells of each solution are represented by continuous variables representing their x- and y-coordinates in the well catchment.

A large range of possible multi-objective metaheuristics was screened to find a suitable optimal algorithm for this problem. The algorithm that was found to be the most appropriate is a multiobjective genetic algorithm called BORG (Hadka and Reed, 2013), which allows to easily add, remove, modify or evaluate search operators. By starting with this sophisticated algorithm with many possible search operators and removing the unnecessary features, it was found that the algorithm shows good performance with only two simple search operators.

Series of optimization runs were carried out for the different test cases. Based on the results, the following statements can be made:

- The spatial distribution of the contamination sources determines the size and distribution in the objective space of the set of solutions produced. For the test case with possible sources of pollution spread all over the well catchment, the set of optimal solutions was found to be ten times larger than for the test case with one single possible source of pollution.

- Performance targets regarding the proximity and diversity of the solutions produced are easily achieved.

- Consistency of the produced set of solutions is the most difficult performance indicator to satisfy. It can be improved by increasing a parameter of the genetic algorithm: the number of generations.

Overall, it can be stated that the BORG algorithm gave satisfying results to the problem. Another algorithm, simpler, called NSGA-II, also gives relatively good results, but presents the drawback of having the size of the population fixed, while this parameter is self-adaptive in the BORG algorithm.

8 Outlook

While a highly satisfactory optimization algorithm could be found, the issue of some of the algorithm settings still poses open questions. A refined treatment of these settings could further improve the quality of the results, or the computational efficiency of the overall framework.

In the extensive testing of algorithm performances, it has been shown that the key performance metrics is the epsilon indicator, because it is the hardest to satisfy under the conditions and objectives featured in the early-warning problem settings. Further work should focus on estimating the number of generations necessary to reach an acceptable value for this metrics. It is expected that this number depends on the distribution of the size of the solutions (number of monitoring wells per solution).

Another issue is the size of the population. While the BORG algorithm automatically adapts the size of the population to the size of the Pareto front, the NSGA-II requires it as an input parameter. Within the larger context of the research project surrounding this thesis, the NSGA-II algorithm could play a larger role since it is available in MATLAB, like all other components used here, while BORG is available only in the C language.

To solve these issues, one possibility would be to set up a method to assess, according to key characteristics of the water catchment and the distribution of the possible contaminations sources, the expected size of the Pareto front and average size of the solutions. Thus, one could know how many solutions and how many generations are necessary, and better fine-tune the used algorithm to the problem at hand.

Bibliography

Agrawal, R. B., Deb, K., and Agrawal, R. B. (1994). Simulated binary crossover for continuous search space. *Complex Systems*, 9:1–34.

Ballistic Missile Early Warning System (1959). United States Air Force. `http://en.wikipedia.org/wiki/Ballistic_Missile_Early_Warning_System`, visited on 08/08/2013.

Cieniawski, S. E., Eheart, J. W., and Ranjithan, S. (1995). Using genetic algorithms to solve a multiobjective groundwater monitoring problem. *Water Resources Research*, 31(2):399–409.

Deb, K., Joshi, D., and Anand, A. (2002a). Real-coded evolutionary algorithms with parent-centric recombination. In *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, volume 1, pages 61–66. IEEE.

Deb, K., Mohan, M., and Mishra, S. (2003). Towards a quick computation of well-spread pareto-optimal solutions. In *Evolutionary multi-criterion optimization*, pages 222–236. Springer Berlin Heidelberg.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002b). A fast and elitist multiobjective genetic algorithm: NSGA-II. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197.

Delhomme, J. (1979). Spatial variability and uncertainty in groundwater flow parameters: A geostatistical approach. *Water Resources Research*, 15(2):269–280.

Dhar, A. and Datta, B. (2009). Logic-based design of groundwater monitoring network for redundancy reduction. *Journal of water resources planning and management*, 136(1):88–94.

Dougherty, D. E. and Marryott, R. A. (1991). Optimal groundwater management: 1. simulated annealing. *Water Resources Research*, 27(10):2493–2508.

DVGW (2006). Arbeitsblatt W 101: Richtlinien für Trinkwasserschutzgebiete; Teil 1: Schutzgebiete für Grundwasser. *Deutsche Vereinigung des Gas-und Wasserfaches e.V.*

DVGW (2009). Safety in drinking water supply - the new DVGW guidelines W 1001 and W 1002. *Deutsche Vereinigung des Gas-und Wasserfaches e.V.*

Facility location (2013). Wikipedia. `http://en.wikipedia.org/wiki/Facility_location`, visited on the 25/09/2013.

Famine Early Warning Systems Network (1985). US Agency for International Development. `http://www.fews.net/`, visited on 08/08/2013.

Glover, F. (1989). Tabu search-part I. *ORSA Journal on Computing*, 1(3):190–206.

Hadka, D. (2012). MOEA framework, version 1.17. `http://www.moeaframework.org/`, visited on the 08/08/2013.

Hadka, D. and Reed, P. (2012). Diagnostic assessment of search controls and failure modes in many-objective evolutionary optimization. *Evolutionary Computation*, 20(3):423–452.

Hadka, D. and Reed, P. (2013). BORG: An auto-adaptive many-objective evolutionary computing framework. *Evolutionary Computation*, 21(2):231–259.

Holland, J. H. (1975). *Adaption in natural and artificial systems.* The University of Michigan Press.

Jha, M. and Datta, B. (2012). Three-dimensional groundwater contamination source identification using adaptive simulated annealing. *Journal of Hydrologic Engineering*, 18(3):307–317.

Kinzelbach, W. (1988). The random walk method in pollutant transport simulation. In *Groundwater flow and quality modelling*, pages 227–245. Springer Netherlands.

Kirkpatrick, S., Jr., D. G., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

Kita, H., Ono, I., and Kobayashi, S. (1999). Multi-parental extension of the unimodal normal distribution crossover for real-coded genetic algorithms. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 2. IEEE.

Kitanidis, P. K. (1986). Parameter uncertainty in estimation of spatial functions: Bayesian analysis. *Water Resources Research*, 22(4):499–507.

Kollat, J. B., Reed, P., and Kasprzyk, J. (2008). A new epsilon-dominance hierarchical bayesian optimization algorithm for large multiobjective monitoring network design problems. *Advances in Water Resources*, 31(5):828–845.

Kollat, J. B. and Reed, P. M. (2006). Comparing state-of-the-art evolutionary multi-objective algorithms for long-term groundwater monitoring design. *Advances in Water Resources*, 29(6):792–807.

Kukkonen, S. and Deb, K. (2006). A fast and effective method for pruning of non-dominated solutions in many-objective problems. *Lecture Notes in Computer Science*, 4193:553–562.

LaBolle, E. M., Fogg, G. E., and Tompson, A. F. (1996). Random-walk simulation of transport in heterogeneous porous media: Local mass-conservation problem and implementation methods. *Water Resources Research*, 32(3):583–593.

Laumanns, M., Thiele, L., Deb, K., and Zitzler, E. (2002). Combining convergence and diversity in evolutionary multiobjective optimization. *Evolutionary Computation*, 10(3):263–282.

Li, Y. and Chan Hilton, A. B. (2007). Optimal groundwater monitoring design using an ant colony optimization paradigm. *Environmental Modelling & Software*, 22(1):110–116.

Matérn, B. et al. (1960). Spatial variation. stochastic models and their application to some problems in forest surveys and other sampling investigations. *Meddelanden fran statens Skogsforskningsinstitut*, 49(5).

McKinney, D. C. and Lin, M.-D. (1994). Genetic algorithm solution of groundwater management models. *Water Resources Research*, 30(6):1897–1906.

Mull, R. (1981). Ground-water protection zones. *GeoJournal*, 5(5):473–481.

Nicklow, J., Reed, P., Savic, D., Dessalegne, T., Harrell, L., Chan-Hilton, A., Karamouz, M., Minsker, B., Ostfeld, A., Singh, A., et al. (2009). State of the art for genetic algorithms and beyond in water resources planning and management. *Journal of Water Resources Planning and Management*, 136(4):412–432.

Nowak, W., Schwede, R. L., Cirpka, O. A., and Neuweiler, I. (2008). Probability density functions of hydraulic head and velocity in three-dimensional heterogeneous porous media. *Water Resources Research*, 44(8):W08452.

Papapetridis, K. and Paleologos, E. (2011). Contaminant detection probability in heterogeneous aquifers and corrected risk analysis for remedial response delay. *Water Resources Research*, 47(10):10518.

Pareto, V. (1896). *Cours d'economie politique*. F. Rouge.

Pelikan, M. (2005). *Hierarchical Bayesian optimization algorithm*. Springer Berlin Heidelberg.

Reed, P. M., Hadka, D., Herman, J. D., Kasprzyk, J. R., and Kollat, J. B. (2013). Evolutionary multiobjective optimization in water resources: The past, present, and future. *Advances in Water Resources*, 51:438–456.

Reed, P. M. and Minsker, B. S. (2004). Striking the balance: long-term groundwater monitoring design for conflicting objectives. *Journal of Water Resources Planning and Management*, 130(2):140–149.

Salamon, P., Fernàndez-Garcia, D., and Gómez-Hernández, J. J. (2006). A review and numerical assessment of the random walk particle tracking method. *Journal of Contaminant Hydrology*, 87(3):277–305.

Seshadri, A. (2009). NSGA-II: A multi-objective optimization algorithm. Matlab Central File Exchange. `http://www.mathworks.com/matlabcentral/fileexchange/10429-nsga-ii-a-multi-objective-optimization-algorithm`, visited on 03/09/2013.

Sierra, M. R. and Coello, C. A. C. (2005). Improving PSO-based multi-objective optimization using crowding, mutation and epsilon-dominance. In *Evolutionary Multi-Criterion Optimization*, pages 505–519. Springer Berlin Heidelberg.

Storey, M. V., van der Gaag, B., and Burns, B. P. (2011). Advances in on-line drinking water quality monitoring and early-warning systems. *Water Research*, 45(2):741–747.

Storn, R. and Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359.

Talbi, E.-G. (2009). *Metaheuristics: from design to implementation*, volume 74. John Wiley & Sons Hoboken.

Tsutsui, S., Yamamura, M., and Higuchi, T. (1999). Multi-parent recombination with simplex crossover in real coded genetic algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 1, pages 657–664.

Vrugt, J. A. and Robinson, B. A. (2007). Improved evolutionary optimization from genetically adaptive multimethod search. *Proceedings of the National Academy of Sciences*, 104(3):708–711.

Yang, Y., Wu, J., Sun, X., Wu, J., and Zheng, C. (2013). A niched pareto tabu search for multi-objective optimal design of groundwater remediation systems. *Journal of Hydrology*, 490:56–73.

Yenigül, N., Elfeki, A., Van den Akker, C., Dekking, F., et al. (2006). A decision analysis approach for optimal groundwater monitoring system design under uncertainty. *Hydrology and Earth System Sciences Discussions Discussions*, 3(1):27–68.

Zienkiewicz, O. C. and Taylor, R. L. (1977). *The finite element method*, volume 3. McGraw-Hill London.