Optimal sampling in a noisy genetic algorithm for risk-based remediation design

Gayathri Gopalakrishnan, Barbara S. Minsker and David E. Goldberg

ABSTRACT

A groundwater management model has been developed that predicts human health risks and uses a noisy genetic algorithm to identify promising risk-based corrective action (RBCA) designs. Noisy genetic algorithms are simple genetic algorithms that operate in noisy environments. The noisy genetic algorithm uses a type of noisy fitness function (objective function) called the sampling fitness function, which utilises Monte-Carlo-type sampling to find robust designs. Unlike Monte Carlo simulation modelling, however, the noisy genetic algorithm is highly efficient and can identify robust designs with only a few samples per design. For hydroinformatic problems with complex fitness functions, however, it is important that the sampling be as efficient as possible. In this paper, methods for identifying efficient sampling strategies are investigated and their performance evaluated using a case study of a RBCA design problem. Guidelines for setting the parameter values used in these methods are also developed. Applying these guidelines to the case study resulted in highly efficient sampling strategies that found RBCA designs with 98% reliability using as few as 4 samples per design. Moreover, these designs were identified with fewer simulation runs than would likely be required to identify designs using trial-and-error Monte Carlo simulation. These findings show considerable promise for applying these methods to complex hydroinformatic problems where substantial uncertainty exists but extensive sampling cannot feasibly be done.

Key words | design, genetic algorithms, groundwater, optimization, remediation, uncertainty

Gayathri Gopalakrishnan*
Barbara S. Minsker (corresponding author)
Civil and Environmental Engineering,
University of Illinois,
3230 NCEL, MC-250,
205 N Mathews Avenue,
Urbana,
IL 61801,
USA
*Now at Geomatrix Consultants Inc.,
Minneapolis,
MN,
USA

David E. Goldberg
General Engineering,
University of Illinois,
117 Transportation,
MC-238,
104 S Mathews Avenue,
Urbana,
IL 61801,
USA

INTRODUCTION

Simple genetic algorithms (GAs) have been used in numerous engineering design applications within water resources (e.g. Wang 1991; Ritzel *et al*. 1994; Wang & Zheng 1997; Smalley *et al*. 2000; Aksoy & Culver 2000) to identify optimal solutions. One of the principal reasons for using a GA as compared to more traditional optimisation methods is that the decision space is searched from an entire population of possible designs. This allows the GA to solve discrete, non-convex, discontinuous problems without differentiation (Goldberg 1989). However, applying GAs or any other optimisation method to real-world problems that have numerous sources of uncertainty ('noise') in the system can be challenging. Here 'noise' is defined as any factor that hinders accurate evaluation of the fitness (objective function) of a given trial design.

These factors can include the use of approximate fitness functions, the use of noisy data, knowledge uncertainty, sampling and human error. Some examples of problems where 'noise' exists include computer models where computational speed is an issue, resulting in the use of a fast but less accurate fitness function compared to a slow but accurate and less 'noisy' one; problems in the groundwater remediation field where there is uncertainty ('noise') in the data used to evaluate the problem due to incomplete knowledge of parameters such as aquifer hydraulic conductivity; and any problem where there is a potential for human or other errors while collecting field data.

A GA that operates in a noisy environment is referred to as a 'noisy GA'. Noisy GAs were used for the first time

for image registration (Grefenstette & Fitzpatrick 1985; Fitzpatrick & Grefenstette 1988). They can also be used in problems where the optimal design must be effective for a range of possible parameter values or models. The noisy GA uses sampling from the noisy fitness function to evaluate fitness of candidate solutions. Smalley *et al.* (2000) demonstrated the efficiency of a noisy GA in identifying reliable risk-based corrective action (RBCA) designs with only a few samples. However, because computing time is usually a constraint when solving real-world problems, determining an optimal sample strategy is essential in order to reduce the amount of computational effort involved. The purpose of this paper is to present several methods for determining the optimal sampling strategy using existing theory from the genetic algorithm literature. The methods are demonstrated within the framework of a RBCA design case study. The effects of using different sampling strategies on the GA's performance and the reliability of the designs produced by the algorithm are analysed. Guidelines for selecting parameter values for a noisy GA are also presented, extending the method of Reed *et al.* (2000) to the noisy GA.

THE NOISY GENETIC ALGORITHM

The simple genetic algorithm

Genetic algorithms (GAs) search for the optimal solution to a problem using techniques that are analogous to Darwinian 'natural selection'. The decision variables defined for the optimisation model are coded as a string of binary digits (alleles) or real numbers. These strings, each representing a decision variable, are linked to form the 'chromosome'. Each chromosome represents a single trial design. The management model in this paper uses binary coding to represent the solutions. Several chromosomes or trial designs are grouped together to form a 'population', which in turn forms a 'generation'.

In order to determine the optimal solution, the GA first randomly generates an initial population of trial designs. The fitness of each member of this initial population is determined using a user-defined objective

function and constraints for the problem of interest. Once the fitness of the entire population of designs is determined, the GA uses three classic Darwinian operators to evolve the population to the optimal or near-optimal solution—selection, crossover and mutation. The process of selection occurs first. In this procedure randomly selected strings are compared and the 'fittest' of them are allowed to enter the mating pool. A number of selection techniques exist in the literature but binary tournament selection is used in this paper because Goldberg & Deb (1991) showed that tournament selection is the most efficient and least prone to premature convergence of all of the selection mechanisms. The next operator is crossover. In this process, members of the mating pool are coupled together to mate with a specific probability P_c. Mating involves selecting one or more 'crossover' points where the strings exchange bits (genes) with each other. Uniform crossover, where each bit of the offspring is copied from the corresponding bit in one of the two parents with equal probability, is used in this paper as the crossover technique. The final operator used by the genetic algorithm is mutation, where randomly selected bits within the new population are changed with a given probability of mutation P_m. The process of forming new designs and evaluating existing ones continues until the GA converges or a maximum number of generations is reached.

The noisy genetic algorithm

As mentioned earlier, a noisy genetic algorithm (NGA) is simply a genetic algorithm that operates in a 'noisy' environment. A noisy environment is commonly encountered while solving real-world problems, where knowledge about the domain is scarce and uncertainty is present. This type of 'noise' prevents accurate evaluation of the fitness of individual members of the population. As a result of this inaccurate evaluation of the individual's fitness, the user-defined objective function (also known as the fitness function) that operates in a noisy environment is termed a 'noisy' fitness function (see Miller (1997) for details).

A type of noisy fitness function called a 'sampling fitness function' is often used in noisy GAs (Miller &

Goldberg 1996). This function uses sampling in order to evaluate the amount of noise from fitness evaluations in noisy environments. Sampling is performed by taking the mean of multiple noisy fitness evaluations for a given trial design in accordance with the Central Limit Theorem. By this theorem, if the fitness function follows any distribution with mean μ and variance σ^2, the sampling distribution of the mean approaches a normal distribution with mean μ and variance σ^2/n as the sample size n increases (for reference, see Ross 1985). Using this theorem, Miller & Goldberg (1996) showed that a sampling fitness function with a sample size of n can be described as follows:

$$f_{i,n}^* = \frac{1}{n} \sum_{j=1}^{n} f_{i,j}^* \qquad (1)$$

where $f_{i,j}^*$ is the jth noisy fitness evaluation of individual i and $f_{i,n}^*$ represents an approximation to the actual (unknown) noisy fitness value.

Unlike Monte Carlo simulation modelling, which requires extensive sampling, the noisy GA with the sampling fitness function performs best with few samples. To understand why this is true, consider how a GA works. Goldberg (1989) states that the optimal solutions in a GA are obtained by combining highly fit building blocks in the populations of strings. Because the population contains many strings representing each building block, multiple samples of a particular building block's fitness will be found in the population, even if only one sample is drawn from the noisy fitness function for each string. As the population evolves, any strings that fail to perform well under a variety of sampled conditions will be outperformed by more robust designs and eliminated from the population using the evolutionary operators. Hence the noisy GA gives robust designs even with little sampling.

However, using the noisy GA with the sampling fitness function could result in an entire range of possible fitness functions being generated by simply changing the sample size of the sampling fitness function. Thus, different sample sizes produce fitness functions that are Pareto-optimal in terms of speed and accuracy (Miller & Goldberg 1996). A major challenge in applying the noisy GA to real-world applications is in identifying the best sample size (n in Equation (2)) for the sampling fitness

function. When determining the optimal sample size for the sampling fitness function, the tradeoffs between increasing computational time and decreasing the noise level must be considered. Using a larger sample size gives the GA a more accurate fitness evaluation but results in additional computational time. The optimal sample size is where the performance penalty due to additional sampling is balanced by the faster convergence of the GA due to lower noise. The following sections describe how an optimal sampling strategy for the noisy GA can be determined for real-world applications such as the risk-based corrective design problem described below.

GROUNDWATER REMEDIATION DESIGN APPLICATION

The application used to demonstrate the performance of the NGA in this paper minimises the cost of a given remediation design while simultaneously meeting a target risk level. The case study used here was developed using data from the Borden site as detailed in Smalley *et al.* (2000). The aquifer configuration is shown in Figure 1. The dimensions of the study are approximately 60 m by 20 m and the aquifer was modelled using a coarse grid of 16×8 elements. A coarse grid was used because numerous runs needed to be done to test the sampling strategies for this study. The coarse grid was derived from a finer mesh of 128×64 elements that were used to generate multiple hydraulic conductivity realisations. The hydraulic conductivity generation technique is detailed in Smalley *et al.* (2000). Note that the techniques described in this paper are independent of the sampling methods used to generate realisations and can be applied to any other method of generating hydraulic conductivity realisations (e.g. Feyen *et al.* 2001). Multiple parameter sets were defined, with each set consisting of a single sample drawn randomly from the set of generated realisations and from each of nine variable exposure model parameter realisations (see Smalley *et al.* (2000) for a list of these parameters). The contaminant benzene, with an initial peak concentration of 133 mg/l, was assumed to be present on the site. 'Pump
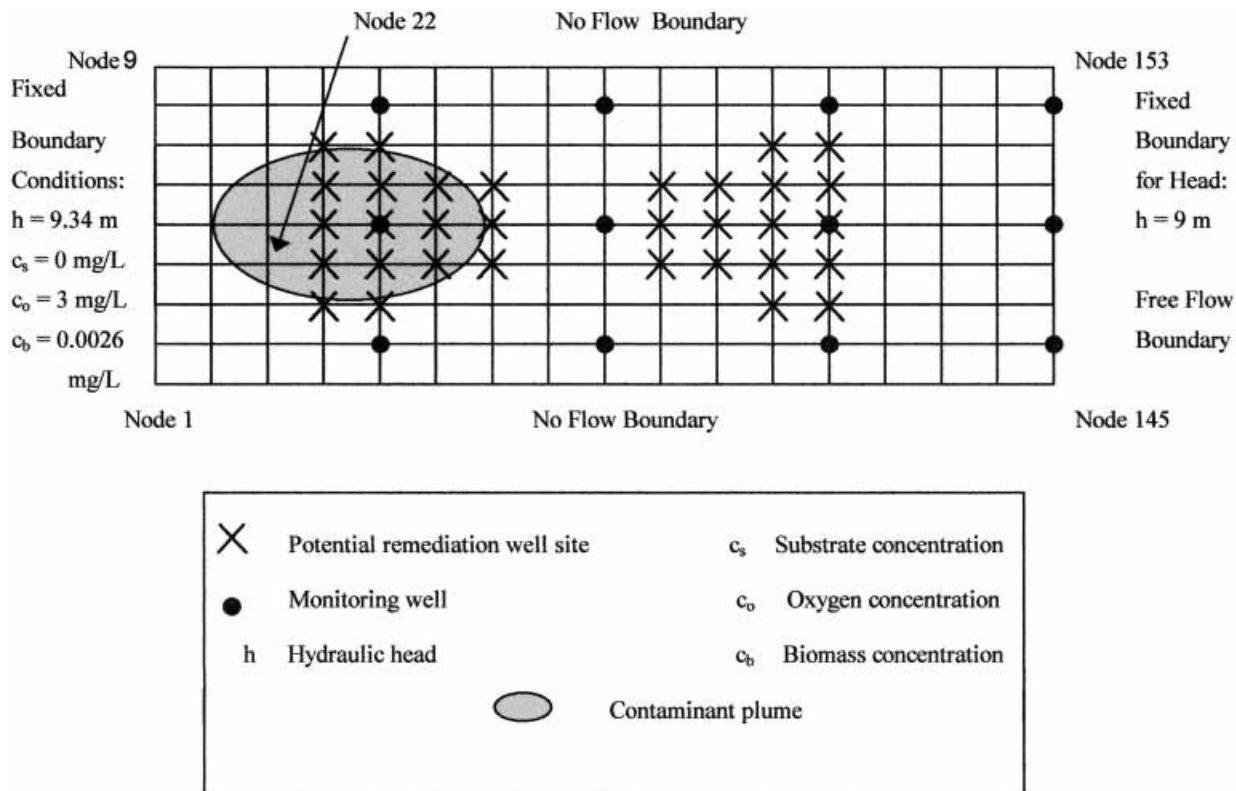
Figure 1 | Plan view of the case study aquifer.

and treat' with extraction wells was the assumed remediation technology. Because the contaminants are the semi-volatile BTEX compounds, air stripping was the selected *ex situ* technology. At most two extraction wells were assumed to be installed at the locations shown in Figure 1. Pumping rates were allowed to vary between 0 and 200 m³/d and it was assumed that the extracted water was treated with the *ex situ* treatment system.

A risk-based remediation plan was developed for the case study using the risk management model developed in Smalley *et al.* (2000). The management model combines a genetic algorithm with a fate and transport simulation model and a risk assessment module to identify promising remediation designs.

The fate and transport model, RT3D, is used to predict contaminant concentrations that would be measured in the contaminant source area for each possible design

solution, which consists of well locations and pumping rates for extraction wells. An existing reaction package (the no-reaction or tracer transport package) in RT3D was used to model advection, dispersion and diffusion of the contaminant (see Clement *et al.* (1998) for details). In order to minimise the computational effort involved in this case study so that numerous test runs could be made, biodegradation was not considered.

The risk assessment module uses an analytical model that predicts contaminant concentrations at off-site exposure wells given concentrations in the source area and estimates human health risks associated with the predicted concentrations. For further information on this module and the rationale for using an analytical model, see Smalley *et al.* (2000).

The goal of this optimisation is to identify a least-cost design that meets a specified risk level. Each design is

represented by a binary string consisting of six decision variables, which are the locations of the extraction wells, their pumping rates, and the decision to install each well or not. The cost of each design is represented by the following objective function:

$$\text{Min} C_{\text{TOT}} = C_{\text{REM}} + C_{\text{MON}} + C_{\text{SYST}} \qquad (2)$$

where the total cost C_{TOT} consists of three components: C_{REM}, which is the capital and operating costs for the wells; C_{MON}, which is the cost of on-site monitoring; and C_{SYST}, which includes additional capital and operating costs for the *ex situ* treatment system (for details on the first two terms, see Smalley *et al.* (2000)). The cost of the *in situ* bioremediation system in Smalley *et al.* (2000) was replaced here by the cost of an on-site *ex situ* treatment system. The cost data for the treatment technology were obtained from RACER (1999), a parametric cost modelling system:

$$C_{\text{SYST}} = c^{cap} + [c^{op}](P/A,i,n) \qquad (3)$$

where

c_{cap}	= capital cost of *ex situ* technology
c^{op}	= operating cost of the technology
$(P/A,i,n)$	= financial factor for converting a series of n O&M payments to present worth, given an interest rate of i.

The genetic algorithm searches for solutions that best meet this objective subject to the constraints described below.

The first constraint ensures that the total individual lifetime human health risk, $Risk_{t,k}^{\text{TOTAL}}$, at a time t and exposure location k, is less than the target risk level, TR:

$$Risk_{t,k}^{\text{TOTAL}} = Risk_{t,k}^{w} + Risk_{t,k}^{shw} + Risk_{t,k}^{nc} \le TR \, \forall t, \forall k \qquad (4)$$

where $Risk_{t,k}^{w}$, $Risk_{t,k}^{shw}$ and $Risk_{t,k}^{nc}$ are the cancer risks due to ingestion of contaminated drinking water, inhalation of volatiles from contaminated water due to showering, and inhalation of volatiles from contaminated water due to other non-consumptive uses respectively (see Smalley *et al.* (2000) for details on how these terms are calculated).

The remaining constraints are as follows:

$$u_{min,i} \le |u_i| \le u_{max,i} \qquad \forall i \qquad (5)$$

$$h_{min,l} \le h_{i,l} \le h_{max,l} \qquad \forall i \quad \text{at each } l \qquad (6)$$

Equations (5) and (6) represent limits on pumping rates and hydraulic heads, where $u_{min,i}$ and $u_{max,i}$ represent the minimum and maximum pumping rates for a given remediation well i (m³/d); $h_{i,l}$, $h_{min,l}$ and $h_{max,l}$ are the computed hydraulic head for remediation well i (m), the minimum hydraulic head (m) and the maximum hydraulic head (m) allowed at remediation well location l, respectively.

The pumping constraints given in Equation (5) are enforced by limiting the number of bits allowed for the pumping rate decision variables (u_i) in the GA chromosomes to 6 bits. Penalties for violations of the risk and head constraints (Equations (4) and (6)) from Smalley *et al.* (2000) were added to the objective function given in Equation (2) to create the following fitness function:

$$\text{Fitness} = C_{\text{TOT}} + \omega_1 \times \text{Risk violation} +$$
$$\omega_2 \times \text{Head violation} \quad (7)$$

where C_{TOT} is given in Equation (2) and ω_1 and ω_2 are the penalty weights for the risk and head constraints, respectively. For this case study, the values of ω_1 and ω_2 have been set at 1000 and 1, respectively. Because the risk constraint is more important than the head constraint, any violations are more heavily penalised.

SAMPLE SIZE DETERMINATION: STEPS TO ENSURE HIGH QUALITY SOLUTIONS

Three steps are developed below for identifying an optimal sampling strategy for the noisy GA. First, the population size and other standard GA parameters are determined. Second, the noise variance and fitness variance are estimated. Finally, the optimal sample size is identified. Details on each of these steps are given below. The steps are tested using the remediation design application described earlier.

Step 1: determine the population size and other standard GA parameters

The most important step in designing a competent noisy GA is fixing the population size correctly. This is especially true for computationally intensive applications where the GA must find the optimal solution in a fixed amount of time. When the population size is too large, redundant individuals are processed, which reduces the number of generations the GA can process in a fixed computational time and hence reduces the solution quality. On the other hand, a small population size can cause the GA to converge prematurely to a sub-optimal solution.

To determine the population size that will result in optimal performance of the GA, the three-step method developed by Reed *et al.* (2000) for the simple GA can be used. However, the population sizing model (Equation (1) in Reed *et al.* (2000)) must be replaced by a model that considers noise. Harik *et al.* (1997) developed a population sizing model called the 'random walk' based on the gambler's ruin problem. Miller & Goldberg (1996) modified this model to account for the presence of noise in the system, resulting in the model shown in Equation (8):

$$N \geq -2^{K-1} \ln(\alpha) \left(\frac{\sqrt{\pi(\sigma_F^2 + \sigma_N^2/n)}}{d} \right) \qquad (8)$$

where N is the population size, K is the building block order, d is the minimum signal difference between competing individuals, α is the probability of failure, σ_F^2 is the variance of the true fitness function, σ_N^2 is the variance due to the noise and n is the sample size. The variance of the true fitness function σ_F^2 describes the variance in fitness that would be present across the population of candidate designs if there was no uncertainty in the system. The variance of the noise σ_N^2 describes the variance of the fitness of each design when sampling is done, i.e. when the design is exposed to a wide variety of conditions. The primary difference between Equation (8) and the model used by Reed *et al.* (2000) is the presence of the term σ_N^2/n relating the effect of noise on population size. This additional term increases the population size to ensure that a good solution is found despite the presence of noise.

Table 1 | Population sizing parameters

Parameter	Value
Standard deviation (σ)	1.83
Signal difference (d)	0.13
Probability of failure (α)	0.05
ln(a)	– 2.99
Pi(π)	3.14

Building block order	Population size
K	N
1	75
2	150
3	300
4	598
5	1,197

As mentioned in Miller (1997), the sum of the population fitness variance σ_F^2 and the noise variance σ_N^2/n in Equation (8) can be assumed to be equal to the initial noisy fitness variance σ^2 of the population. Reed *et al.* (2000) used a randomly generated trial population with 1000 members to determine the initial fitness variance and showed that this resulted in a conservative estimate of the population size. Similarly, a trial population with 1000 individuals and a sample size of 1 is used to determine the initial noisy fitness variance σ^2 of the noisy fitness function. As can be seen from Equation (8), using a larger sample size results in a smaller value of the variance and hence a lower value of the population size. Hence, using an initial sample size of unity in the trial population results in conservative estimates for the population size.

The parameters and the population sizes resulting from applying this method to the RBCA design application are listed in Table 1 below. As noted by Reed *et al.* (2000), the signal difference should be set to the smallest fitness

difference between competing individuals that the GA should be able to resolve. In this case the signal difference d was set to 0.13, which is the amount the penalty function would have to change in order to measure a 0.01% violation of the target risk and a 0.03% violation of the head constraint using the penalty function described in Equation (7).

One important consideration while determining the correct parameters for the population-sizing model is the correct formulation of the problem, especially setting the penalty weights for the constraints. As mentioned by Reed *et al.* (2000), excessive penalisation of the constraint violations results in infeasibly large population sizes. However, if constraint violations are not penalised enough, i.e. the penalty weights are set too low, then the constraints may not be satisfied. Hence, before either the population sizing model or the sampling strategy can be used, the penalty function needs to be formulated appropriately.

The other parameters in Equation (8), K (the building block order) and α (the probability of failure), are set according to the guidelines given by Reed *et al.* (2000). The building block order, which is the number of bits in each building block, is generally unknown in most real-world applications. However, the order can be assumed to be five or less because larger building blocks would be disrupted by the crossover operation (Reed *et al.* 2000). Finally, as recommended by Reed *et al.* (2000), the probability of crossover (P_c) for tournament selection was set to 0.5 and the probability of mutation (P_m) was set to be the inverse of the population size.

For the RBCA application, when the building block order K is equal to 1 or 2, population sizes of 75 and 150, respectively, must be used. The trial runs using these population sizes resulted in almost identical solutions being found for both cases. As suggested by Reed *et al.* (2000), this indicates that a population size of 75 will suffice for this problem and no additional runs for higher values of K are required. This approach will still be valid when sampling is needed because increasing the number of samples from 1 results in a lower value of the term σ_N^2/n in Equation (8) and hence a lower value for the population size. Thus, the method described above results in conservative estimates for the population size.

Step 2: estimate the noise variance and fitness variance

Once the population size is set, several other parameters for the noisy GA must be determined. To identify the optimal sample size, the noise variance σ_N^2 and the true fitness variance σ_F^2 must be estimated first. Once σ_N^2 is estimated, the value of the true fitness variance σ_F^2 can then be obtained by subtracting the variance due to the noise from the variance of the noisy fitness function determined earlier. For cases where the noise component is not dependent on the fitness of the individual, the noise variance can be obtained by determining the variance of y samples of a randomly selected individual in the trial population. When the noise component is dependent on the fitness of the individual, the noise variance can be set to the average mean noise variance of the trial population. This can be obtained by selecting x individuals, using y samples to obtain the noise variance of each individual and then taking the mean of the x noise variances that were obtained (Miller 1997). For this application, the noise was highly dependent on the fitness of the individual because the noise was present in the risk estimate, which appears in the penalty term of the fitness function.

When the fitness function evaluation is computationally intensive, it is desirable to minimise the amount of sampling required to estimate σ_N^2. To examine the effects of different amounts of sampling on the noise variance estimate, one generation of an initial trial run for a sample size of 5 and a randomly generated initial population of 75 was performed. Members with the maximum, minimum and average noise variance in this generation were found, representing the most conservative estimate, the least conservative estimate and an average value for the noise variance. For each of these three members, repetitive trial runs with sample sizes between 5 and 1000 were performed to obtain the noise variance estimates per sample size (σ_N^2/n) shown in Figure 2. This figure shows that the noise variance stabilised at a sample size of 300. However, the value for the noise variance when stabilisation occurred was less than the maximum value found with a sample size of 5. Hence, the most conservative estimate of the noise variance was that found with the smallest sample size.
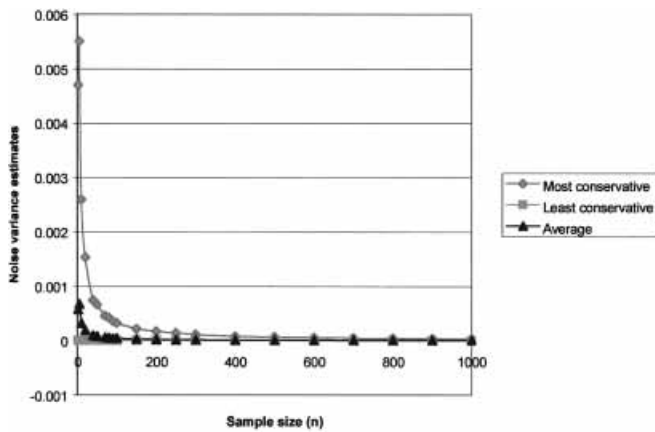
Figure 2 | Effect of sampling on the standard deviation.

In order to demonstrate the effect of the noise variance on the sampling strategy, the highest value, the lowest value and a randomly chosen 'typical' value of the noise variance (see Table 2) were used to estimate the optimal sample size as shown in the following subsections. The highest value of the noise variance was estimated as the maximum value found for the most conservative member in Figure 2; the lowest value of the noise variance was approximated by the minimum value found for the least conservative member in Figure 2; and the 'typical' value of the noise variance was estimated as an average value for the 'average' member in Figure 2. These estimates span the range of values for the noise variance shown in Figure 2.

Identify the optimal sample size

Once the population size and the noise variance are estimated, a range of sample sizes can be determined using theoretical relationships from Miller (1997). In developing a predictive model for the optimal sample size, Miller (1997) assumed that the computational time would be fixed and the optimal sample size would be that which achieved the best performance (measured in terms of percent convergence) within that timeframe. However, each sample size has a different convergence rate and hence a different convergence model. Also, each sample size produces a fitness function that is Pareto-optimal

in terms of speed and accuracy (Miller & Goldberg 1996). Hence, any of the sample sizes could result in the GA performing optimally within a given timeframe. Identifying the optimal sample size would then require an exact convergence model as a function of sample size, which does not exist for most problems.

One of the methods investigated by Miller (1997) to determine the optimal sample size was simple enumeration of all of the sample sizes from 1 onwards until the optimal sample size was identified. In order to avoid the costly trial and error experimentation involved in this method, Miller (1997) presented methods for identifying lower and upper bounds on the sample size. By bounding the range of sample sizes considered for the sampling fitness function, the computational effort involved in identifying appropriate sample sizes to achieve a desired level of reliability can be significantly reduced.

Lower bound sample size determination

To develop an estimate for the lower bound sample size, Miller & Goldberg (1996) assumed that any improvement in convergence rate due to increased sampling and the resulting lower noise variance can be ignored. Under this assumption, the same convergence rate can be assumed for GA runs with all sample sizes, which means that the lower bound for the sample size can be estimated as that sample size which maximises the ending generation. Miller & Goldberg (1996) justified the use of this estimated lower bound because faster convergence and decreased population sizes when higher sample sizes are used will result in improved performance of the GA and hence the optimal sample size must be at least as large as the lower bound.

To derive the lower bound, an equation must be developed for the ending generation. Fitzpatrick & Genfenstette (1988) developed a model for the total time T required by the GA as

$$T = (\alpha' + \beta n)GN \tag{9}$$

where G is the total number of generations, N is the population size and n is the sample size of the sampling

Table 2 │ The lower bound values, n_{lb}

Parameter	Most conservative estimate	Least conservative estimate	Typical estimate
Fixed GA overhead time (α') (s)	1.07	1.07	1.07
Single fitness evaluation time (β) (s)	7.46	7.46	7.46
Standard deviation for noise (σ_N)	0.188	1.97E-03	0.023
Standard deviation for fitness (σ_F)	1.83	1.83	1.83
Lower bound	0.04	0.0004	0.005

fitness function. The variable α' represents the fixed amount of GA overhead time per individual per generation, which includes the time required for selection, crossover and mutation but not for the fitness function evaluation. The variable β represents the time required for a single fitness function evaluation. The costs of generating the initial population have been ignored because they are negligible when compared to the costs of running the GA.

The above equation can be used to determine the ending generation G as a function of T, α', β, n and N. From Equation (8), it can be seen that the population size N is also a function of the sample size n. Assuming a fixed amount of available computational time T, the value of the ending generation reduces to a function of the sample size n because all of the other values are assumed constant. Combining Equations (8) and (9) and rearranging, Equation (10) can be obtained:

$$G(n) = \frac{T}{(\alpha' + \beta n)(-2^{k-1}) \ln \left(\alpha \sqrt{\pi \left(\sigma_F^2 + \frac{\sigma_N^2}{n} \right)} \right)} . \quad (10)$$

Miller & Goldberg (1996) use the assumptions mentioned earlier to show that the ending generation, given in Equation (10), can be maximised by setting $dG/dn = 0$ and solving for n, which is the lower bound for the optimal sample size (n_{lb}) shown in Equation (11):

$$n_{lb} = \sqrt{\frac{\alpha'}{\beta}} \sqrt{\frac{\sigma_N^2}{\sigma_F^2}} \quad (11)$$

where σ_F^2 is the true fitness variance and σ_N^2 is the noise variance. The values for the variances were determined earlier and the other variables α' and β can be obtained from the trial runs of the noisy GA for the population sizing. Using these values, the lower bound of the sample size can be determined.

To estimate the lower bound for this application, Equation (11) was applied for the most conservative, least conservative and typical estimates of the noise variance from Step 2. The parameters and results are shown in Table 2. For a computationally intensive problem, such as the one studied in this paper, the value of β is typically much greater than α'. In such a case, unless the noise variance is substantially greater than the fitness variance (implying that the actual value of the fitness cannot be estimated to any degree of accuracy), the lower bound as estimated from Equation (11) will be less than 1. When this occurs, the lower bound of the sample size can be set equal to 1.

In this particular case study, the noise variance is much less than the true variance of the fitness and hence the lower bound found from Equation (11) is less than 1 for all of the cases. This indicates that the noise variance does not have an effect on the lower bound and the lower bound of the sample size can be set to 1.

Table 3 | Convergence model parameters for selected members

Parameter	Most conservative estimate	Least conservative estimate	Typical estimate
Standard deviation for noise			
(σ_N)	0.188	1.97E-03	0.023
$\bar{f}(t = 0)$	0.0133	0.0133	0.0133
$\bar{f}(t = 8\ h)$	9.33E-01	0.933	0.933
x	0.163	0.0017	0.019
c	-4.30	-4.30	-4.30

Upper bound sample size determination

To derive an upper bound sample size, Miller (1997) suggests developing an approximate convergence model that converges faster than the actual GA at the same noise level (sample size). The approximate convergence model is used to identify a sample size that maximises the performance of the GA within the given computational time. Given that the convergence model overestimates performance, the sample size determined with that model is an upper bound on the optimal sample size. Miller (1997) suggests the following approximate convergence model, which is representative of GA convergence in most domains:

$$\bar{f}(t) = \frac{\exp\left(\dfrac{x}{\sigma_N}t + c\right)}{1 + \exp\left(\dfrac{x}{\sigma_N}t + c\right)} \qquad (12)$$

where the fitting parameter x is a function of the convergence rate, parameter c is determined by the starting population fitness, t is the time in consistent units and $\bar{f}(t)$ represents the percent convergence, which is defined to be the percentage of alleles in the string that have converged to the optimal solution at time t. Convergence is obtained when most of the members of the final generation have the same set of decision variables, i.e. the same RBCA design for this study.

This model can be calibrated by determining the initial value of the percent convergence, $\bar{f}(0)$, and the final value of the percent convergence, $\bar{f}(T)$, at the end of one sample GA's run in order to compute x and c respectively. The sample runs used for population sizing can be used for this purpose so that no additional runs are required. For this application, the model was calibrated using an initial run with a population size of 75 and an ending generation of 50, which was used earlier to establish the correct value of the population size. Convergence was assumed when at least 90% of the alleles had converged to the same value in the final generation. The values of the model parameters x and c in Equation (12) determined for this application are shown in Table 3 for the three noise variance estimates used previously.

As expected, there is wide variation in the model parameters for the estimate with the most noise (most conservative), the estimate with the least noise (least conservative) and the estimate with the typical value of the noise. However, when the convergence model was developed and plotted as shown in Figure 3, the model was the same for all of the cases. This is reasonable because the model parameters x and c are scaled using the noise variance. This result indicates that the approximate convergence model in Equation (12) is a robust model and does not depend on the amount of noise present in the system. The main factors influencing the model are the
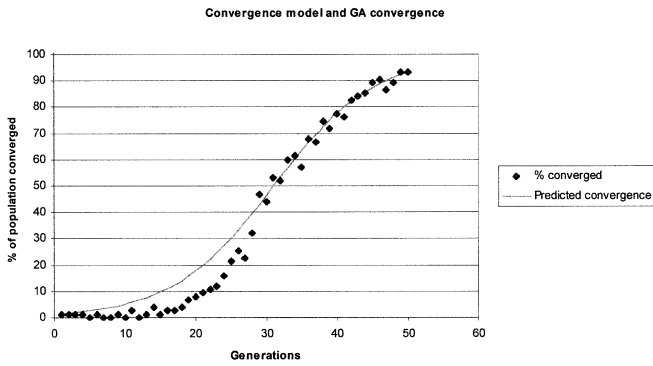
Figure 3 │ Convergence model.

convergence criteria and the time the GA takes to converge to a solution. Hence, any noise variance estimate can be used without affecting the convergence model estimate.

As can be seen from Figure 3, the model converges at a slightly faster rate than the GA does. This result is appropriate because, as stated earlier, the upper bound model is valid only when this condition holds.

Using this convergence model, the upper bound of the optimal sample size can be calculated by finding the sample size that maximises convergence in the ending generation G, $\bar{f}(G)$. As with the lower bound derivation, the ending generation G can be determined as shown in Equation (10). The final convergence, $\bar{f}(G)$, is calculated by combining Equation (10) with Equation (12). The upper bound is then determined by maximising the performance of the GA $(d\bar{f}(G)/dn = 0)$ and solving for the

value of the upper bound (n_{ub}). The differentiation was performed using Mathematica and the resulting equation for the upper bound was found to be

$$An_{ub}^3 + Bn_{ub}^2 + Cn_{ub} + D = 0 \tag{13}$$

where

$$A = \sigma_F^2 \beta^2$$
$$B = 2\alpha'\beta\sigma_F^2 + \sigma_N^2\beta^2$$
$$C = \alpha'^2 \sigma_F^2 + 2\alpha'\beta\sigma_N^2 - \frac{x^2 T^2}{(-2^{k-1}\ln(\alpha)\sqrt{\pi})^2\sigma_N^2 c^2}$$
$$D = \alpha'^2\sigma_N^2.$$

The polynomial function 'roots' in MATLAB was used to solve the cubic equation for the upper bound. The upper bounds calculated using this approach are given in the following tables for various parameter values.

Table 4 shows that the noise variance had minimal effect on the upper bound, as expected given that the convergence model is the same for all three cases. This result and the similar lower bound result found previously indicate that it is not necessary to determine the noise variance accurately in order to find the bounds on the sample size. This finding means that the noise variance can be estimated with small sample sizes, resulting in significant computational savings. Because the variance does not have much effect, the maximum noise variance of 0.188 will be used for subsequent sensitivity analyses.

Table 4 │ Effect of changes in noise variance on upper bound sample size

Parameter	Most conservative member	Least conservative member	Typical member
Standard deviation for noise (σ_N)	0.188	1.97E-03	0.023
Total run time (T) (d)	1	1	1
Single fitness evaluation time (β) (s)	7.5	7.5	7.5
Upper bound (n_{ub})	31	31	31

Table 5 │ Effect of run time on the upper bound sample size

Parameter				
Total allowed run time (T) (d)	1	2	3	4
Standard deviation for noise (σ_N)	0.188	0.188	0.188	0.188
Single fitness evaluation time (β) (s)	7.5	7.5	7.5	7.5
Upper bound (n_{ub})	31	62	93	124

Table 6 │ Effect of function evaluation time on the upper bound sample size

Parameter			
Single fitness evaluation time (β) (s)	7.5	15	23
Standard deviation for noise (σ_N)	0.188	0.188	0.188
Total allowed run time (T) (d)	1	1	1
Upper bound (n_{ub})	31	15	10

One additional factor that should be considered in developing the upper bound is the effect of the allowed runtime (T in Equation (10)). Table 5 shows that the allowed runtime of the GA has a substantial effect on the upper bound. The upper bound increases almost linearly with an increase in the total time the GA is allowed to run. The upper and lower bound analyses assume that this parameter is set by the user to reflect computational limits.

Finally, it is interesting to note the effect of time taken for each function evaluation, β, on the upper bound. As shown in Table 6, when the time taken for each function evaluation is increased, a linear decrease in sample size is noted.

For this paper, a total run time of 1 day was chosen and the value of β is 7.5 s. Using these values, it can be seen from Table 4 that the upper bound on the sample size is 31.

Pareto-pruning

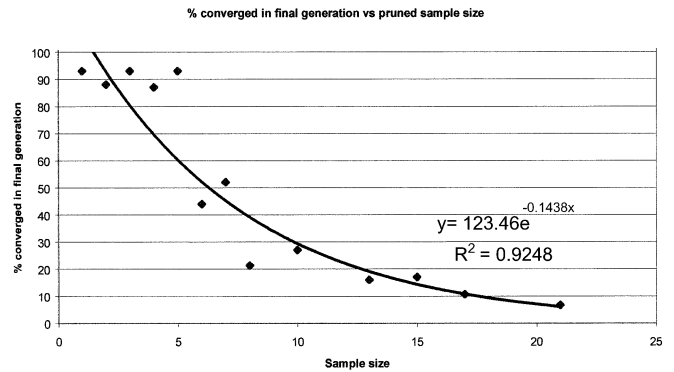In order to find the optimal sample size, each of the sample sizes between the lower and upper bound would theoretically have to be tested. However, this is not necessary because multiple sample sizes can result in the same ending generation. Table 7 shows the ending generation for each sample size in the RBCA design example; these figures were calculated using Equation (8). However, a fraction of a generation cannot be completed, so the ending generation is truncated to the next lower whole number. Miller (1997) showed that, when sample sizes have the same truncated ending generation, only the largest value of the sample size needs to be sampled because it maximises the performance of the algorithm. This technique is commonly referred to in the genetic algorithm literature as 'Pareto-pruning'. The pruned values of the sample sizes are given in Table 7. As seen from Table 7, the sample sizes that need to be considered are 1–13, 15, 17, 19, 21, 25 and 31.
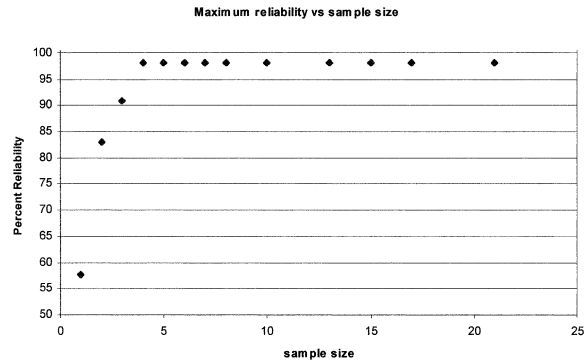
Sample size results

To evaluate the effects of sample size on performance, the genetic algorithm was run for a single day ($T = 1$ in Equation (9)) with the sample sizes shown in the previous

Table 7 | Sample sizes from Pareto Pruning

T (*days*)	1	
α' (s)	1.07	
β (s)	7.5	
N	75	
Sample size	**Ending generation**	**Truncated ending generation**
1	134.9	134
2	71.9	71
3	49.1	49
4	37.2	37
5	30.0	30
6	25.1	25
7	21.6	21
8	18.9	18
9	16.9	16
10	15.2	15
11	13.8	13
12	12.7	12
13	11.7	11
14	10.9	10
15	10.1	10
16	9.6	9
17	9.0	9
18	8.5	8
19	8.0	8
20	7.7	7
21	7.3	7
22	6.9	6
23	6.7	6
24	6.4	6
25	6.1	6
26	5.9	5
27	5.7	5
28	5.5	5
29	5.2	5
30	5.1	5
31	4.9	5

Figure 4 | Performance of the algorithm as a function of sample size.

Plot title: % converged in final generation vs pruned sample size; fitted curve $y = 123.46e^{-0.1438x}$, $R^2 = 0.9248$

Figure 5 | Design with the maximum reliability as a function of sample size.

Plot title: Maximum reliability vs sample size

section. The results indicated clear tradeoffs between convergence and reliability of the RBCA designs identified. Figure 4 shows how the performance (measured in terms of percentage of the population converged) varies for different sample sizes. Note that the sample sizes 1 and 2 converged before the total run time of 1 day and were stopped after 50 generations. The higher sample sizes ran for fewer generations and hence did not converge within the allowed run time of 1 day.

To evaluate reliability, the designs identified in the GA runs were tested using 5000 Monte Carlo simulations. Figure 5 shows the highest reliability designs that were identified for each sample size. The best design achieved a reliability of 98% (i.e. it satisfied the constraints for 98% of the 5000 simulations). This design consisted of a single extraction well with a pumping rate of 200 m³/d at Node 22 in Figure 1. However, the algorithm did not identify the
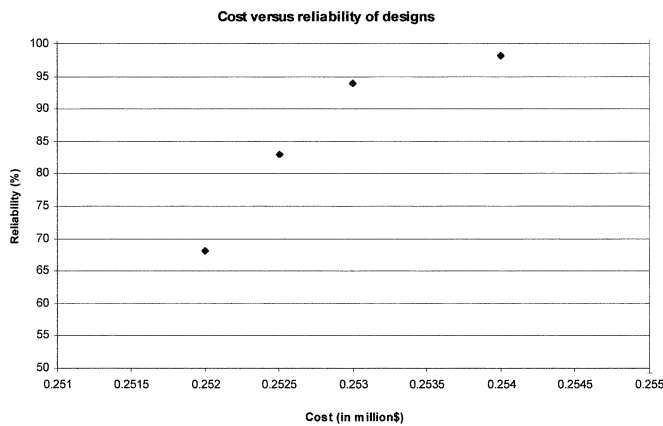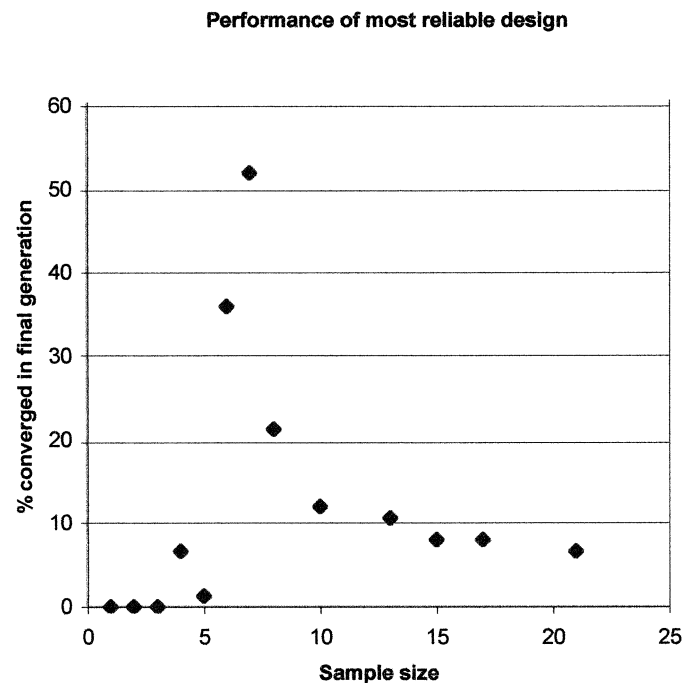
Figure 6 | Cost versus reliability for the primary designs identified.

Figure 7 | Percentage of the 'best' design found in the final generation as a function of sample size.

'best' design at sample sizes of 1 and 2 because designs with lower reliability but lower cost took over the population. Note that the GA required 11,100 fitness function evaluations to identify the 'best' design for a sample size of 4.

Figure 6 shows the costs and reliabilities of the four primary designs identified by the algorithm for a sample size of 7. Primary designs were those designs where at least 1% of the members in the final population had the same design. Designs with a range of reliabilities between 68% and 98% were identified within a fairly narrow range of costs. These designs provide a useful range of alternative reliabilities from a single GA run.

Finally, as can be seen in Figure 7, the best design took over the population only for the sample size of 7. For smaller sample sizes, although it was identified, designs with lower reliability and cost took over the population. For sample sizes greater than 7, convergence was never achieved. These results are, of course, dependent upon the run duration specified. For longer durations, greater convergence would be achieved.

These results indicate that, although the lower and upper bound analysis may identify a large number of possible sample sizes, many will probably not need to be tested. The runs can be started with the lower bound sample size and then progressively increased until either the optimal designs stabilise (i.e. remain unchanged from one sample size to the next) or convergence is no longer

achieved. At that point, the reliability of the designs identified should be tested with full Monte Carlo simulations. If the reliability is too low and convergence is no longer achieved, then the allowed run time may need to be increased if possible. Even if the run time cannot be increased, the designs identified may be sufficiently reliable for screening purposes, identifying candidate types of designs for further analysis.

To test the optimality of the most reliable design identified in the preceding analysis, an additional run with a 2-day duration was completed with a sample size of 15. This run showed convergence to the same solution found previously, suggesting that optimal solutions can be identified even when full convergence cannot be achieved. Finally, it was noted that the fitness of the optimal design was sampled 2,130 times during this run, which explains how such a reliable solution can be identified with so little sampling. In order for the optimal design to survive to the ending generation, it must be able to perform well under most sampled conditions.

CONCLUSIONS

This paper develops a three-step method for designing a noisy GA that is capable of identifying highly reliable designs. The three steps use theoretical relationships from the genetic algorithm literature to identify appropriate GA parameters and to develop optimal sampling strategies. Step 1 of the methodology provides valuable insights into developing population sizes and other standard GA parameters. Step 2 identifies noise variance estimates that are needed for the optimal sample size determination. Step 3 develops the optimal sampling strategy based on theoretical upper and lower bounds for the sample size. The three-step process was demonstrated using a RBCA design test case. Using this approach, the sampling strategy used for the test problem identified four candidate designs with 68–98% reliability for the optimal sample size of 7. The candidate designs were tested with full Monte Carlo simulations to identify the best design based on tradeoffs between cost and reliability. Relative to trial and error simulations, this GA-based approach may result in significant savings of computational effort because fewer designs may need to be evaluated using expensive Monte Carlo simulations. For example, for a sample size of 4, only 11,100 fitness function evaluations (simulations) were required to identify the optimal design with 98% reliability. Given that each design could take thousands of evaluations for full Monte Carlo simulations, it seems unlikely that equally optimal solutions could be identified by trial and error simulation with so few evaluations. These results indicate considerable promise for this method to identify highly reliable solutions to complex hydroinformatic problems without substantial computational effort beyond that required for a simple genetic algorithm without noise.

ACKNOWLEDGEMENTS

REFERENCES

Aksoy, A. & Culver, T. B. 2000 Effect of sorption assumptions on aquifer remediation designs. *Groundwater* **38**(2), 200–208.

Clement T. P., Sun, Y., Hooker, B. S. & Petersen, J. N. 1998 Modeling multispecies reactive transport in ground water. *Groundwater Monit. Remed.* **18**(2), 79–92.

Feyen, L., Bevan, K. J., De Smedt, F. & Freer, J. 2001 Stochastic capture zone delineation within the generalized likelihood uncertainty estimation methodology: conditioning on head observations. *Wat. Res. Res.* **37**(3), 625–638.

Fitzpatrick, J. M. & Grefenstette, J. J. 1988 Genetic algorithms in noisy environments. *Machine Learning* **3**, 101–120.

Goldberg, D. E. 1989 *Genetic Algorithms in Search, Optimizations and Machine Learning*. Addison-Wesley, New York.

Grefenstette, J. J. & Fitzpatrick, J. M. 1985 Genetic search with approximate function evaluations. In: *Proceedings of an International Conference on Genetic Algorithms and their Applications* (ed. Grefenstette, J. J.), pp. 112–120, Hillsdale, NJ.

Harik, G. R., Cantu-Paz, E., Goldberg D. E. & Miller, B. L. 1997 The gambler's ruin problem, genetic algorithms and the sizing of populations. In: *Proc. 1997 IEEE Conference on Evolutionary Computation*, pp. 7–12, IEEE Press, New York.

Miller B. L. 1997 Noise, sampling and efficient genetic algorithms, *IlliGAL Report* No. 97001, May.

Miller, B. L. & Goldberg, D. E. 1996 Optimal sampling for genetic algorithms. In: *Intelligent Engineering Systems Through Artificial Neural Networks (ANNIE '96)* (eds Dagli, C. H., Akay, M., Chan, C. L. P., Fernandez, B. R. & Ghosh J.), vol. 6, pp. 291–298, ASME Press, New York.

Patrick, R., Minsker, B. & Goldberg, D. 2000 Designing a competent simple genetic algorithm for search and optimization. *Wat. Res. Res.* **36**(12), 3757–3761.

RACER 99 1999 Remedial Action Cost Engineering and Requirements Users Guide. Talisman Partners Ltd.

Ritzel, B. J., Eeheart, J. W. & Ranjithan, S. 1994 Using genetic algorithms to solve a multiple objective groundwater pollution containment problem. *Wat. Res. Res.* **30**(5), 1589–1603.

Ross, M. S. 1985 *Introduction to Probability Models*, 3rd edn. Academic Press, London.

Smalley, J. B., Minsker, B. S. & Goldberg, D. E. 2000 Risk-based in situ bioremediation design using a noisy genetic algorithm. *Wat. Res. Res.* **36**(20), 3043–3052.

Wang, Q. J. 1991 The genetic algorithm and its application to calibrating conceptual rainfall-runoff models. *Wat. Res. Res.* **27**(9), 2467–2471.

Wang, M. & Zheng, C. 1997 Optimal remediation policy selection under general conditions. *Groundwater* **35**(5), 757–764.