A Dynamic Meta-Model Approach to Genetic Algorithm Solution of a Risk-Based Groundwater Remediation Design Model

[1]Shengquan Yan and [2]Barbara Minsker

[1] Research Assistant, Department of Civil and Environmental Engineering, University of Illinois, 205 North Mathews, MC-250, 4146 Newmark Civil Engineering Laboratory, Urbana, IL 61801-2352. (smyan@uiuc.edu)
[2] Associate Professor, Department of Civil and Environmental Engineering, University of Illinois, 205 North Mathews, MC-250, 3230 Newmark Civil Engineering Laboratory, Urbana, IL 61801-2352. (minsker@uiuc.edu)

Abstract
Approximation ("meta") models have been used in coupled water resources optimization and simulation models to improve computational efficiency. In most instances, multiple simulation runs have been done before the optimization, which are then used to fit an approximate model that is used for the optimization. In this study, we propose a dynamic meta-modeling approach, in which artificial neural networks (ANN) is embedded into a genetic algorithm (GA) optimization framework to replace time-consuming flow and contaminant transport models. Data produced from early generations of the GA are sampled to train the ANN. We propose a dynamic learning approach that periodically re-samples new solutions both to update the ANN and correct the GA's converging route. This allows the meta model to adapt to the area in which the GA is searching and provide more accuracy. The results show that a proper sampling strategy can benefit both GA's searching and ANN's retraining. In our test case, more than 90 percent of the numerical model calls were saved with no loss in accuracy of the optimal solution.

Introduction

Groundwater management models often involve coupling complex simulation models with optimization models to achieve management goals. For the risked-based remediation design model studied in this paper (Smalley and Minsker, 2000), a flow model, contaminant transport model, chemical reaction model, and risk evaluation model are combined with optimization to identify optimal solutions. The combination of those different complex sub-models makes traditional optimization approaches difficult for finding a global optimum. Genetic algorithms are well suited for solving such problems, but can require substantial computational resources when each objective function (fitness) evaluation involves solving time-consuming contaminant transport and chemical reaction models (PDEs). The GA optimization process needs to evaluate the fitness function thousands of times before it can converge to the global optimum. This is a significant limitation when applying the GA optimization framework to more complex remediation cases, even with the help of parallel computation.

To alleviate the computational burden, many less accurate but much more computationally efficient approximation methods have been explored. Cooper et al. (1998) used curve-fitting methods to approximate the response surface. Aly and Peralta (1999) and Rogers et al. (1995) used artificial neural network (ANN) and GA to optimize groundwater problems. While these methods can be helpful in GA optimization, they used a static response surface that was trained using a well designed off-line data set before the optimization began. This requires the off-line data set to be carefully selected so that it is representative of the problem domain, which can be a time-consuming process by itself. Furthermore, the approximated static response surface, albeit accurate at the beginning, may be less and less representative when the GA population converges to other local areas later in the run. As a result, the GA may prematurely converge to regions containing local minima. If the problem has strict constraints, a small error in the constraints may incur large penalties or the large penalties may be ignored if ANNs underestimate the constraints. This makes the situation even worse because the GA may follow the false fitness to an infeasible area.

A few studies have proposed adaptive response surface approaches in recent years. Brooker (1998) proposed a framework for generating and managing a sequence of surrogates to the objective function. Jin et al. (2002) used Evolutionary Algorithms (EAs) with ANNs that adapt to sampling model errors to accelerate aerodynamic design problems. However, none of these studies have focused on surrogate functions within GAs, and using sampling model errors to adjust sampling rates (i.e., how often the real function is used) in a GA is less stable when the sampling rate is low.

This paper proposes a unique dynamic approach to create and adjust the response surface within a GA optimization framework. The algorithm periodically samples data points from the GA population based on population similarity and the fitness estimations by ANNs and uses that information to synchronize the response surface with the current GA population. The sampled solutions are also used to update the ANN's estimations to maintain accurate solutions within the population. As the GA converges to the optimal solution in a particular local region, less sampling is required because the ANNs become more accurate on local approximation due to the retraining. Different sampling and retraining methods, including a chromosome caching technique that can further reduce numerical model evaluations and increase accuracy are discussed in this paper.

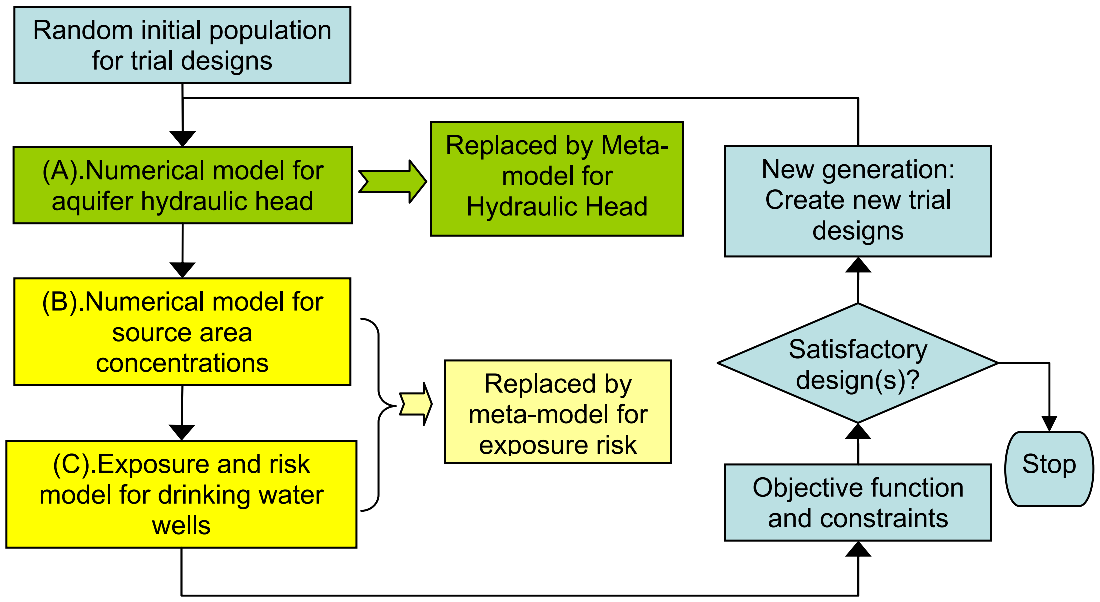
Risk-Based GA Optimization Framework

The risk-based GA optimization framework developed by Smalley et al. (2000) is depicted in Figure 1, along with notations for where the meta-models would replace the simulation models in this paper. First, an initial population of trial designs is generated randomly. Second, the numerical groundwater flow model (component A, MODFLOW) is called to simulate the flows in the source area for the specified set of parameters. Third, the chemical fate and transport model (component B, RT3D) is called to estimate the contaminant plume spreading over time. The exposure and risk assessment model (component C) estimates the human health risk at drinking water wells down gradient from the contaminant source. The risk information is incorporated into the GA to produce penalties if any of the risk standards are violated. The GA then uses selection, crossover, and mutation to create new trial designs for the next iteration.

The objective function, given by Equation 1, represents the total cost of a remediation design, 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. The details are presented by Smalley et al. (2000). The goal of the optimization framework is to find the least-cost solution without violating the pumping rate and hydraulic head limits and risk criteria. The GA repeatedly tests possible locations to install pumping wells and pumping rates for the remediation process until the global or near-global optimum is found.

$$MinC_{TOT} = C_{REM} + C_{MON} + C_{SYST} \qquad (1)$$

Although the GA can handle this typical mixed-integer problem, it is computationally expensive if all of the numerical sub-models are fully realized throughout the GA optimization process. MODFLOW, the groundwater flow model, and RT3D, the chemical transport and fate model, can be time-consuming for complex sites. Instead of evaluating the numerical models, two meta-models created using ANNs are embedded into the framework to replace the numerical models. The first one replaces MODFLOW (Component A). The second one replaces the combination of RT3D and the risk evaluation model (component B plus component C). If the two meta-models are adequately trained and updated, the optimization process can be greatly accelerated without excessively compromising accuracy.


Random initial population for trial designs
(A).Numerical model for aquifer hydraulic head
Replaced by Meta-model for Hydraulic Head
New generation: Create new trial designs
(B).Numerical model for source area concentrations
Replaced by meta-model for exposure risk
Satisfactory design(s)?
(C).Exposure and risk model for drinking water wells
Objective function and constraints
Stop

Firgure 1. NNGA optimization framework

Meta-Model Methodology

Artificial Neural Network (ANN). ANNs have been successfully applied to numerous applications, including Rogers, 1995, and Aly, 1999 in the water resources field. Their widespread application in different disciplines can be attributed to their ability to approximate almost any linear or non-linear problems. An ANN has many neuron-like units. Each unit accepts input and gives output according to its activation function. The interconnections among the neurons in an ANN have weights associated with each connection, which compose a large parameter set. By gradually tuning the weights associated with each interconnection, a supervised learning algorithm can learn the mapping relationship between the inputs and the outputs sampled from a training set. Then the trained ANN is used to make a prediction.

We adopt a two layer, feed forward neural network for our meta-models. As shown in Figure 2, the input data are directly sent to the hidden layer. After the hidden layer transformation, the output layer has the final predicted results. The activation function of each neural unit is the sigmoid function, which is convenient for derivative calculation for the training algorithm. The training algorithm is Levenberg-Marquardt backpropagation method, which is much faster than steep descent backpropagation with affordable memory requirement. Each training episode has a training set and a smaller validation set. The training iterations terminates when the Mean Square Error (MSE) on the validation set stops decreasing, which avoids overfitting.

ANN Approximation Properties. The response surface of an ANN usually smoothly interpolates all of the points in the training set. If the actual function has many local minima, the response surface trained on a global sampling may smooth out the local regional details to maintain a small MSE on the global area. To illustrate this property, Figures 3 and 4 show the 2 dimensional Ackley function (Jin, 2002) surface and its response surface from an ANN trained on 150 sampling points within the global region ([-4, 4] for both x and y). The ANN has 2 inputs for x and y, 8 nodes in the hidden layer and one output. The mathematical formula for the Ackley function is:

$$f(x) = 20 + e - 20\exp\left[-0.2\exp\left(\sqrt{\frac{1}{n}\sum x_i^2}\right) - \exp\left(\frac{1}{n}\sum \cos(2\pi x_i)\right)\right] \qquad (2)$$

If the above ANN is used to replace the Ackley function in a GA framework, the GA’s population will quickly converge to a local region within [-1, 1] because the surrogate correctly captures the backbone of the surface despite the error on specific local points. After the GA finds the right local region, however, it will converge to a false optimum. Figures 4 and 5 show the local detail of the Ackley function and the ANN’s response surface. The true global minimum is at (0,0) which the surrogate misses.

Increasing training points can alleviate the problem for a simple surface, but for a complex surface like the Ackley function, it is very inefficient or even infeasible. A better way to improve the local approximation is by changing the sampling points in the training set. If we sample 100 of the 150 points from the local region [-1, 1] and leave the other 50 untouched, then the local approximation of the same ANN has a much better result (Figure 6). The new response surface correctly captures the local topology and even the global minimum point.

The property that the same ANN can have either global or local approximation ability only by changing the sampling to create the training set is beneficial when applied to a GA framework. At early generations, it is more critical to identify where the promising regions are, so a random sampling on the global region should suffice. After the population moves into local regions, adaptive sampling in those areas can update the response surface to give better identification of the local shapes and locations of minima. Note that at early generations, when the GA is moving quickly from region to region, more sampling and retraining is required. At later generations, less and less sampling is required because the local region is relatively smooth and the ANN is already sufficiently accurate.

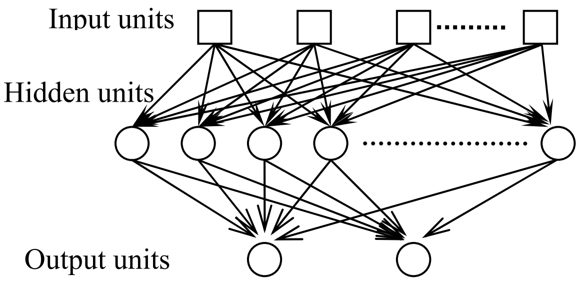


Figure 2. ANN architecture

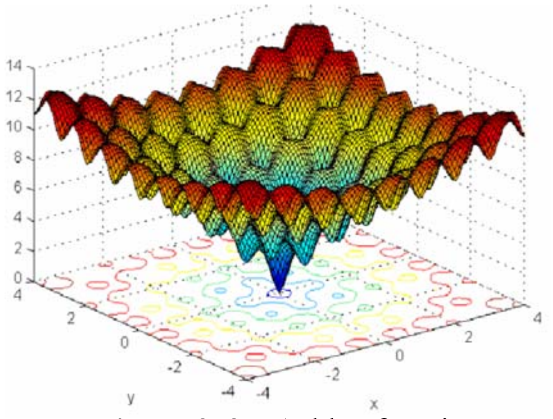


Figure 3. 2D Ackley function

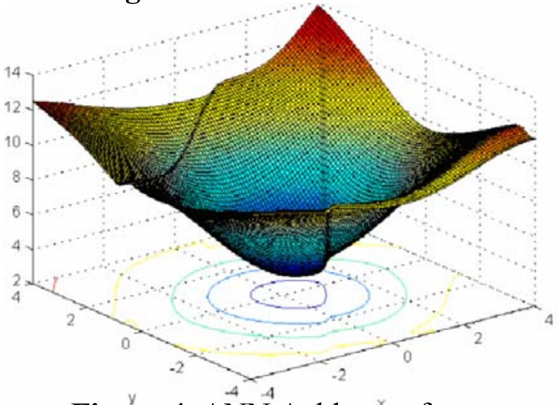


Figure 4. ANN Ackley surface

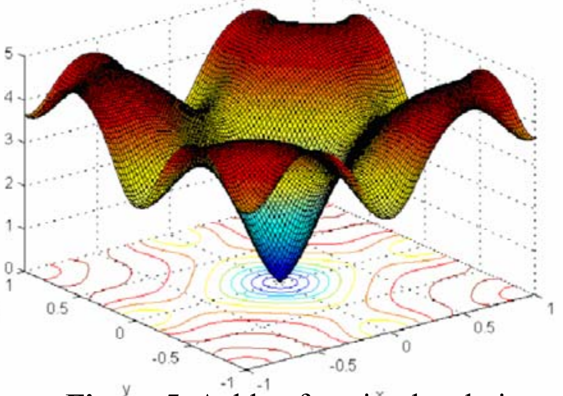


Figure 5. Ackley function local view

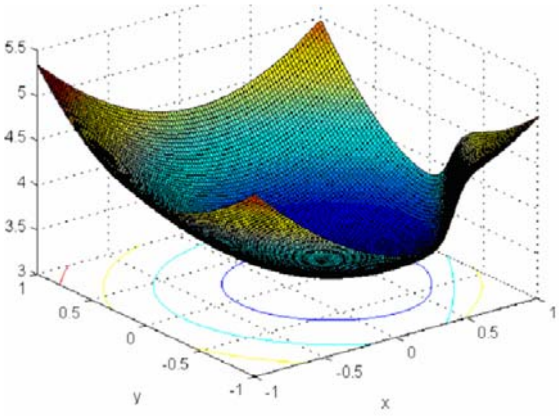


Figure 6. ANN surface local view

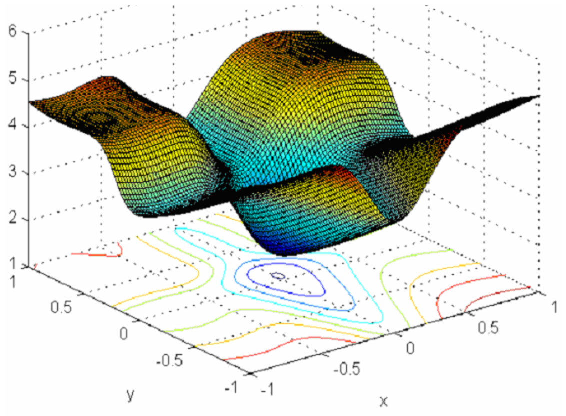


Figure 7. Retrained ANN local view

Caching techniques. Caching can improve GA's performance by storing the evaluated chromosomes into the memory and retrieve it quickly when the same chromosome reappears (Kratica, 1999). When the population becomes more homogeneous at late generations, caching can save a significant number of fitness evaluations through an efficient looking up algorithm in the memory.

When incorporated into NNGA, caching technique plays a more active role. If a chromosome is already in the cache, NNGA uses the cached fitness instead of ANN's estimation. This can improve the searching accuracy in the local regions because GA's searching is based on the cached true fitness in the near homogenous population rather than estimations. In this study, we use an efficient caching algorithm called AVL tree.

Sampling. Although ANN can learn the global shape through random sampling at early generations, the surrogate is not error-free and it can occasionally lead GA to wrong local regions. Once GA moves into a wrong local region, it is much harder to jump out because ANN is also adapted to the same local region through retraining. The remedy for this problem is sampling. At each generation, a number of points are sampled by the numerical models (PDE calls) to replace the ANN's estimations. When these true fitness are mixed into the population, the GA has a much higher likelihood of finding the best local region. Sampling is more active at early generations because it improves GA's global searching ability. Once the GA has found the right region, caching and the adapted ANN become more active, thus less sampling is required. Note that the sampling points are also the retraining points that help the ANN to adapt to local regions.

Sampling strategy. Two sampling strategies are tested in this work, random sampling and best sampling. In random sampling, members of the population are randomly selected for PDE evaluations. For best sampling, the chromosomes are first passed to the ANN to estimate their fitness. Then the most promising points are given higher probability of selection for PDE evaluations.

Sampling rate. An adaptive NNGA adjust its sampling rate to reflect the current level of homogeneity in the population. A heterogeneous population needs a higher sampling rate to improve GA's global searching ability and increase the retraining frequency (assuming a fixed number of sampling points needed for retraining). To estimate the similarity of a population, we use a scaled average input standard deviation (*SD*), where the standard deviation of each input variable value is calculated (and scaled to the range [-1.0, 1.0] to make the method problem independent) and then the different standard deviations are averaged across input variables. *SD* is always changing from a value usually close to 0.58 in the first generation to a value close to 0 when converging. To make the estimation smoother in a dynamic environment, a smoothed scaled standard deviation (*SSD*) can be used instead.

$$SSD_i = SSD_{i-1} + \eta(SD_i - SSD_{i-1}) \qquad (3)$$

$$SD_i = \frac{1}{n}\sum_{k=1}^{n} std(X_k) \qquad (4)$$

X_k	— Scaled input variables, scaled to [0, 1]
n	— number of input variables (decision variables)
SD_i, SSD_i	— (smoothed) scaled standard deviation at generation *i*
η	— smoothing constant in [0, 1], eg. 0.5.

So the sampling rate is,

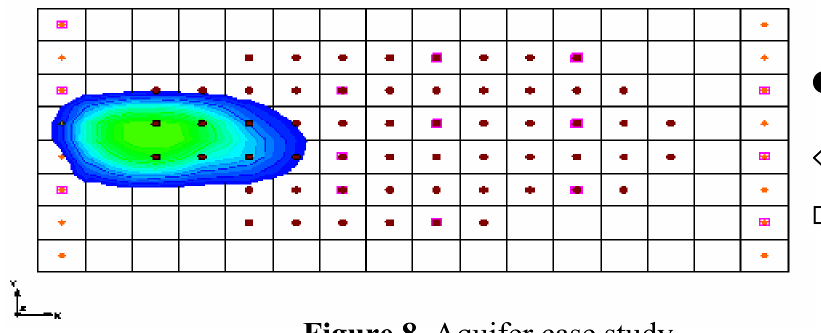
$$S_i = \frac{SSD_i}{SSD_0} S_0 \qquad (5)$$

where S_i is the sampling rate. S_0 is the maximum sampling rate.

The sampled solutions are put into a sampling pool with a user-specified capacity for samples (e.g., 100 solutions). When the sampling pool is full, a new retraining is triggered and the pool is emptied for the next retraining.

Case Study

A simple case study is used in the research (Figure 8). The aquifer plane is discretized into a 16 x 8 mesh grid. Due to regional flow and dispersion, the contaminant plume spreads slowly to the down gradient area. Three pumping wells are the candidates for pumping treatment. The decision variables are their locations in the mesh grid and the pumping rates. Monitoring wells are installed at the boundaries and several locations in the source area. The contaminant concentration readings are used to predict human health risk associated with drinking the water.



● Possible remediation well locations

◇ Fixed head boundaries

□ Monitoring well location

Figure 8. Aquifer case study

The GA has a population of 100 individuals, which was determined using the population sizing approach developed by Reed et al. (2000) in previous work. The fitness evaluation of each individual requires about 1~2 minutes using the numerical models. The algorithm converges after about 96 generations. The global minimum found by SGA has the fitness close to 186.0, with no constraints violated. For the NNGA, the numerical models are called in the first several generations to generate the training data set. The GA is then suspended to wait for the meta-model training. After that, sampling, caching and the meta-model are used to evaluate the fitness.

The ANN meta-model has two sub-models. The first model uses the pumping well locations and the pumping rates as input to produce output hydraulic heads at the pumping wells. The pumping well locations, the pumping rates, and the hydraulic heads at those wells are then used to feed a second sub-model, which predicts the maximum log risk predicted at all monitoring wells. Log risk was used to avoid scaling problems associated with extremely small numbers.

The groundwater flow sub-model of ANN has 9 units in the input layer, 12 units in the hidden layer and 3 units in the output layer. The risk evaluation sub-model of ANN has 12 units in the input layer, 14 units in the hidden layer and 1 unit in the output layer. Inputs and outputs of the meta-models are scaled to the range *[-1.0, 1.0]*. Mean Square Error (MSE) is used to evaluate the predictive accuracy. All of the MSE results presented here are based on the scaled data.

Results.

To explore the performance of NNGA, we ran the test case on different combinations of the following three adjustable algorithm parameters:

Maximum sampling rate (S_0). This parameter determines the sampling rate of NNGA and the retraining frequency of the ANNs. NNGA automatically adjusts its sampling rate according to the population similarity and S_0, as shown in Equation 4. We tested NNGA performance for values of S_0 equal to 5, 10, 20 and 30.

Initial training generations (G_0). This parameter determines the initial training set size because the first ANN training is after G_0 generations. The training set size is G_0*N, where N is the population size. We tested the NNGA with values of G_0 equal to 3, 5 and 8, so the corresponding training set size is 300, 500 and 800.

Sampling strategy. We tested both the random sampling and best sampling strategies described previously.

To reduce the total number of parameter combinations, we adjusted one parameter and fixed the others in each run. For each parameter set, we tested ten random initial populations to get a better sense for algorithm performance.

Figure 9 shows NNGA performance for different values of variable S_0. The x axis is the run number, where each run number corresponds to a different initial population. The y axis is the converged fitness value. For S_0>10, the NNGA almost always converged to the expected global minima. When S_0 is decreased to 5, 3 of the ten runs were misled to local minima. This likely occurs because when the sampling rate is too low, the GA has insufficient true fitness values to ensure good global searching at early generations. Figure 10 shows the average PDE calls and average converged fitness values for the four sampling levels. As S_0 increases, the required PDE overhead increases at a much lower speed. For example, when S_0 is increased 6-fold from 5 to 30, the PDE calls increased only 2-fold). This is because a higher sampling rate at early generations causes quicker convergence. This in turn decreases SSD and the sampling rate more rapidly (see Equation 5).

Figure 11 shows NNGA performance for variable G_0. For all the candidate G_0, the NNGA converged to the expected global minima. Hence NNGA is quite insensitive to the size of the initial training set as long as it is not too small to cause over fitting on the first training. Figure 12 shows the average PDE calls and average fitness for the three values of G_0. The required PDE overhead is again increasing at a much slower speed as G_0 increases.

Figure 13 shows NNGA performance on different sampling strategies. It is clear that for both S_0=10 and 20, the best sampling strategy gives improved performance over the random sampling strategy, which is more likely to converge to local minima. When NNGA does best sampling, it first figures out the most promising individuals (low fitness individuals) using the ANN's prediction. Those promising chromosomes are also the most active chromosomes when GA applies selection and crossover (global search) to produce the next generation, which improves global searching. Furthermore, those chromosomes are also the most promising points for guiding the ANN to an improved local approximation in promising areas.

Figure 14 shows the proportions of PDE evaluations, ANN evaluations, and calls to cache in each generation. In the first half of the run, ANN replaces most PDE calls. In the second half of the run, caching is more active and replaces most PDE calls. Figure 15 shows a comparison of PDE calls on NNGA, SGA+caching and SGA. The figure shows that SGA needs about 9,600 PDE calls to converge, SGA+caching requires about half that many calls, and the NNGA needs only about 900 calls. Thus, the NNGA saved more than 90 percent of the PDE calls required to find the optimal solution.

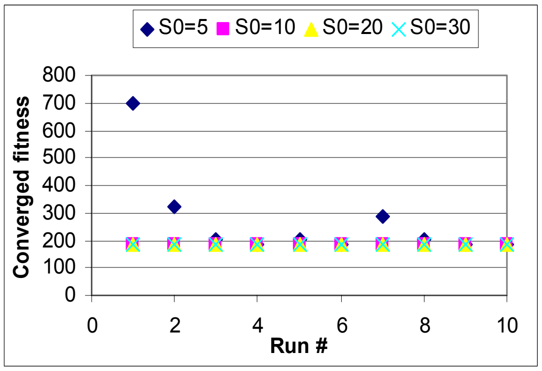


Figure 9 Results on different S_0

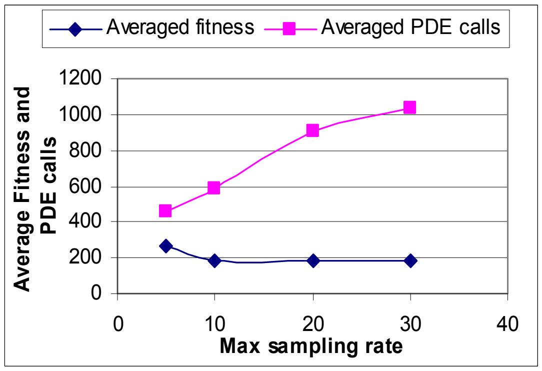


Figure 10 Averaged Results on different S_0

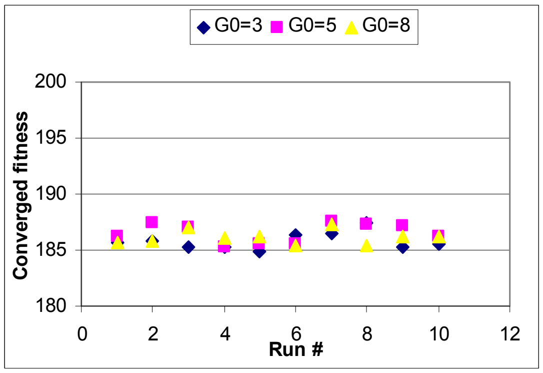


Figure 11 Results on different G_0

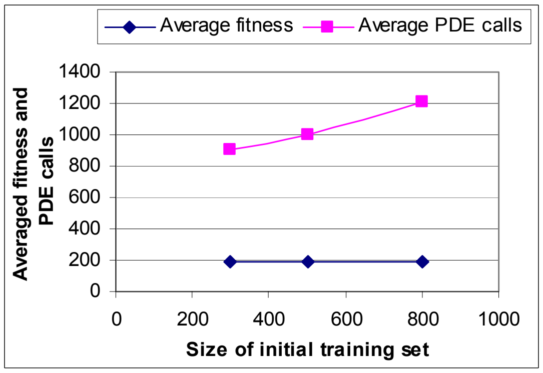


Figure 12 Averaged Results on different G_0

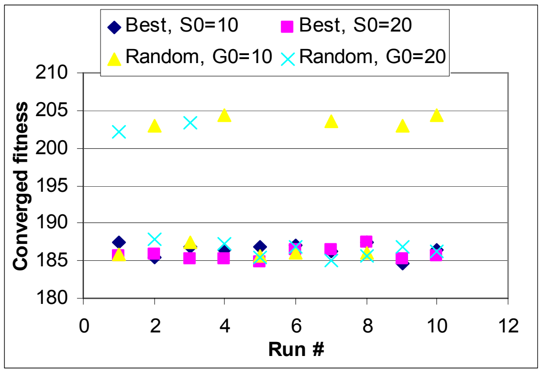


Figure 13 Results on sampling strategis

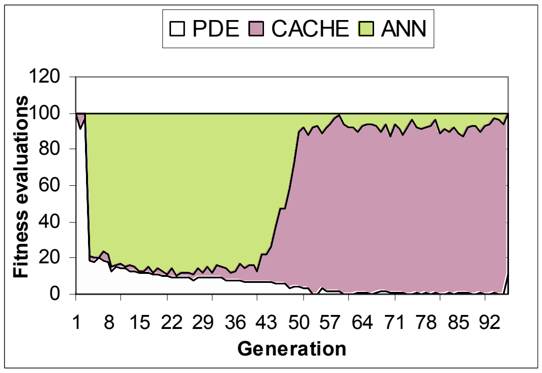


Figure 14 PDE, Cache and ANN evaluations

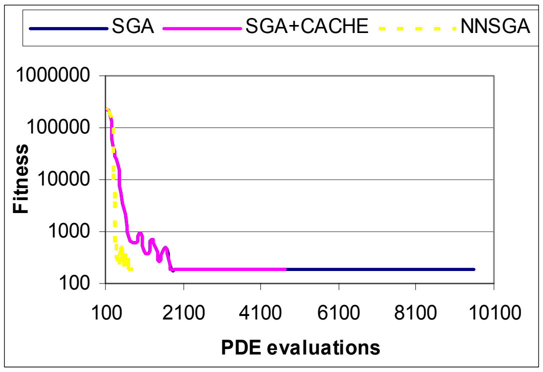


Figure 15 SGA, SGA+cache, NNGA comparison

Conclusion

A statically trained ANN is error-prone in the dynamic environment of a GA. If ANN is adaptively retrained, ANN can gradually move from global approximation to local approximation by injecting local sampling into the training set. When best sampling strategy is applied, ANN's estimation can be used to find the most promising points, which provide the best information for GA's searching and ANN's retraining. By combining ANN, caching and adaptive sampling, NNGA saved about 90% of the time-consuming fitness evaluations required to solve a test case while still achieving accurate solutions. Furthermore, the adaptive sampling algorithm used in the NNGA is insensitive to most parameter settings, alleviating need for extensive fine-tuning of parameters.

Acknowledgement

This material is based upon work supported by the U.S Army Research Office under grant number DAAD19-001-1-0025.

References

Aly, Alaa H., and Richard C. Peralta, *Optimal design of aquifer cleanup systems under uncertainty using a neural network and a genetic algorithm*, Water Resour. Res., 35(8), 2523-2532, 1999.

Brooker , A. J., J. Dennis, P. D. Frank, D. B. Serafini, V. Torczon, and M. Trosset, *A rigorous framework for optimization of expensive functions by surrogates*, Struc. Optimiz., vol. 17, pp. 1–13, 1998.

Cooper, G., R. C. Peralta, and J. J. Kaluarachchi, *Optimizing separate phase light hydrocarbon recovery from contaminated unconfined aquifers*, Adv. Water Resour., 21(5), 339-350, 1998

Dougherty, D.E., and R.A. Marryott, *Optimal groundwater management 1. Simulated Annealing*, Water Resour. Res., 27(10), 2493-2508, 1991.

Fletcher, R. and C. M. Reeves, *Function minimization by conjugate gradients*, Comp. J. **7**-149, 1964.

Gopalakrishnan, G., B. S. Minsker, and D. Goldberg, *Optimal sampling in a noisy genetic algorithm for risk-based remediation design*, J. of Hydroinformatics, 2003.

Jin, Yaochu, Markus Olhofer, and Bernhard Sendhoff, *A Framework for Evolutionary Optimization With Approximate Fitness Functions*, IEEE Transactions on Evolutionary Computation, 6(5), 481-494, 2002

Kratica , J, *Improving performances of the genetic algorithm by caching*, Computers and Artificial Intelligence, 18(3):271—283., 1999.

Reed., P. M., B. S. Minsker, and D. E. Goldberg., *Designing a Competent Simple Genetic Algorithm for Search and Optimization*, Water Resour. Res., 36(12), 3757-3761, 2000.

Rogers, Leah L., Farid U. Dowla, and Virginia M. Johnson, *Optimal Field-Scale Groundwater Remediation Using Neural Networks and the Genetic Algorithm*, Environ. Sci. Technol. 29, 1145-1155, 1995

Russel, Stuart, Peter Norvig, *Artificial Intelligence-A Modern Approach*, Prentice Hall, 1995.

Smalley, J. Bryan, Barbara S. Minsker, and David E. Goldberg, *Risk-based in situ bioremediation design using genetic algorithm*, Water Resour. Res., 36(10), 3043-3051, 2000