Hybrid Optimization Schemes for Simulation-Based Problems

G. A. Gray¹, K. Fowler², and J. D. Griffin³

¹Sandia National Laboratories, Livermore, California, United States, gagray@sandia.gov
²Clarkson University, Potsdam, New York, United States, kfowler@clarkson.edu
³SAS Institute, Cary, North Carolina, United States

Abstract

The inclusion of computer simulations in the study and design of complex engineering systems has created a need for efficient approaches to simulation-based optimization. For example, in water resources management problems, optimization problems regularly consist of objective functions and constraints that rely on output from a PDE-based simulator. Various assumptions can be made to simplify either the objective function or the physical system so that gradient-based methods apply, however the incorporation of realistic objection functions can be accomplished given the availability of derivative-free optimization methods. A wide variety of derivative-free methods exist and each method has both advantages and disadvantages. Therefore, to address such problems, we propose a hybrid approach, which allows the combining of beneficial elements of multiple methods in order to more efficiently search the design space. Specifically, in this paper, we illustrate the capabilities of two novel algorithms; one which hybridizes pattern search optimization with Gaussian Process emulation and the other which hybridizes pattern search and a genetic algorithm. We describe the hybrid methods and give some numerical results for a hydrological application which illustrate that the hybrids find an optimal solution under conditions for which traditional optimal search methods fail.

Keywords:
simulation-based, derivative-free, hybrid, hydraulic capture, pattern search

1. Introduction and Motivation

Comprehensive studies of complex systems in science and engineering benefit from the inclusion of physical experimentation, but in many cases, such experiments are prohibitively expensive or even impossible to perform. To overcome such limitations, scientists and engineers take advantage of computer models and use simulation as an alternative or augmentation to experimental data. Moreover, numerical models can be used to simulate environments that cannot be reproduced or controlled in physical experiment. They can provide higher visibility into system components than physical tests. When physical tests are performed, they are often time consuming, expensive, or only for limited testing conditions. In contrast, computer simulations can be processed in parallel in a fraction of time, and they can also systematically sample the entire test or application space.

The inclusion of computer simulations in the study and design of complex systems has created a need for tools to help gage and improve the predictive capabilities of numerical models. Thus, the pairing of optimization methods and simulation has become essential and the class of “simulation-based” problems has become a focus of optimization...
research. The underlying formulation for simulation-based optimization problems relies on the output from sophisticated simulation programs in the form of numerical solutions for governing conservation laws or systems of PDEs. Often, off-the-shelf software, developed in the academic community or by government agencies, is used to represent the physical phenomena of interest. Hence, optimization algorithms must work in conjunction with pre-existing, or “legacy,” software.

In this paper, we consider the general nonlinear simulation-based optimization problem of the form

\[
\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad x \in \Omega
\end{align*}
\]

where \( f: \mathbb{R}^n \rightarrow \mathbb{R} \) and \( \Omega \subset \mathbb{R}^n \) denotes set of feasible points. Within the simulation-based framework, the objective function to be minimized is often complicated in the sense that derivatives are unavailable and approximate derivatives are often unreliable. Moreover, the objective function landscapes can be disconnected, nonconvex, nonsmooth, or contain undesirable, multiple local minima due to the complicated physical phenomena being modeled. In addition, the goal in solving a simulation-based optimization problem may not be to merely find the classical “best” solution (i.e. the point that results in the smallest function value) but may also include finding a solution which is “robust” with respect to the simulator. For example, it may be the case that answer \( A \) gives a smaller objective function value, but that answer \( B \) is preferable in real world situations because it allows for some error without catastrophic failure.

A variety of derivative-free optimization methods have emerged and matured over the years to address simulation-based problems in general. They are theoretically supported and have established convergence criteria. To name a few, there are global and local methods, population-based methods (e.g. genetic algorithms, evolutionary algorithms, particle swarm optimization), single search methods (e.g. generating set search, simulated annealing, Nelder-Mead), heuristic methods (i.e. involving stochastic optimization parameters), deterministic (i.e. providing the same solution for the same set of optimization parameters). Each has its own strengths and weakness and often these strengths and weaknesses are manifested in the problem being studied. Recently, hybrid optimization techniques have emerged as a means of exploiting the strengths of some methods while overcoming the weaknesses in others. In this paper, we describe two hybrid methods, one for general nonlinear problems with continuous variables and one designed specifically for mixed integer problems.

2. Derivative Free Optimization Methods

To obtain a solution to a simulation-based problem, one seeks an optimization algorithm that is (1) reliable in the sense that similar solutions can be obtained using different initial points or optimization parameters, (2) accurate in that a reasonable approximation to the global minimum is obtained, and (3) efficient with respect to finding a solution using as few function calls as possible. The role of efficiency is particularly important in simulation-based applications because the optimization is guided solely by function values (i.e. not by derivative information) and these function values require the solution of a simulation, usually via a black-box structure. In practice, such simulations can take anywhere from a few seconds to a few days depending on the application and problem size. Thus, parallel implementations of both the simulator and the optimization methods are often essential for obtaining tractable solutions. In this section, we give examples of four derivative-free optimization methods. The first three were used in the hybrid schemes and the fourth was used for comparison tests.

2.1. Pattern Search

For pattern search, we specifically consider Asynchronous Parallel Pattern Search (APPS) [1, 2]. Like all pattern search algorithms, APPS is a direct search method which uses a predetermined pattern of points to sample a given function domain. It has been shown that if certain requirements on the form of the points in this pattern are followed and if the objective function is suitably smooth, convergence to a stationary point is guaranteed [3, 4].

The majority of the computational cost of pattern search methods is the function evaluations, so parallel pattern search (PPS) techniques have been developed to reduce the overall computation time. Specifically, PPS exploits the fact that once the points in the search pattern have been defined, the function values at these points can be computed simultaneously [5, 6]. The APPS algorithm is a modification of PPS that eliminates the synchronization requirements. It retains the positive features of PPS, but reduces processor latency and requires less total time than PPS to return
results [1]. Implementations of APPS have minimal requirements on the number of processors and do not assume that the amount of time required for an objective function evaluation is constant or that the processors are homogeneous.

We consider the specific APPS algorithm as described in [2] which has been implemented in an open source software package called APPSPACK. The details of the implementation are described in detail in [7]. Note that APPSPACK was designed for and has been shown to work well for problems with less than 100 variables (see [7] and references therein).

2.2. Treed Gaussian Process (TGP)

An alternative approach to optimization is statistical emulation, wherein the previous runs of a computer code are used to train a statistical model; and the model is used to draw inferences about the location of the optimum. In particular, a Gaussian Process (GP) is typically used for the emulation of computer simulators [8, 9]. The output of the simulator is treated as a random variable that depends on the input vector such that the response varies smoothly. This smoothness is given by the covariance structure of the GP. The mean and covariance functions determine the characteristics of the process, as any finite set of locations has a joint multivariate Gaussian distribution [10, 11]. A Bayesian approach allows full estimation of uncertainty, which is useful when trying to determine the probability that an unsampled location will be an improvement over the current known optimum. GP models can be used to build a globally oriented pattern of input locations. The selection of points for inclusion in this search pattern is based on predicted improvement statistics at candidate input locations and the current best point (i.e. the point which is closest to optimal). In particular, a predetermined number of locations can be chosen for evaluation through maximization (over a discrete candidate set) of expected multi-location improvements using an expected improvement criterion [12].

Standard GP models have several drawbacks, including strong assumptions of stationarity and poor computational scaling. To reduce these problems, treed Gaussian process (TGP) models partition the input space using a recursive tree structure; and independent GP models are fit within each partition [13, 14]. Such models are a natural extension of standard GP models, and combine partitioning ideas with Bayesian methods to produce smooth fitted functions [15]. The partitions can be fit simultaneously with the parameters of the embedded GP models using reversible jump Markov chain Monte Carlo [16].

2.3. Genetic Algorithms (GA)

Genetic algorithms [17, 18] are one of the best known derivative-free approaches to optimization. Commonly referred to as GAs, genetic algorithms are part of a larger class of evolutionary algorithms called population based global search heuristic methods [17]. GAs are based on biological processes such as survival of the fittest, natural selection, inheritance, mutation, or reproduction. Often, GAs are criticized for their computational complexity and dependence on optimization parameter settings, which are not known a priori [19, 20, 21]. However, if the user is willing to exhaust a large number of function evaluations, a GA can help provide insight into the parameter space and locate initial points for fast, local, single search methods.

2.4. Implicit Filtering

Implicit filtering, a projected quasi-Newton method, evaluates a stencil of points at each iteration used simultaneously to form finite difference gradients and as a pattern for direct search [22]. In contrast to classical quasi-Newton algorithms, implicit filtering uses a much large stencil to account for noise, while improving its direct search component. This step-size is reduced as the optimization progresses to take advantage of the fast convergence of quasi-Newton methods near a local minimum. Because it relies on finite difference gradients, only function values are needed to guide the minimization. For this work, we use a FORTRAN implementation called IFFCO, with the symmetric rank one quasi-Newton update [23]. We used the default optimization parameter settings. There are several convergence theorems for implicit filtering, which was particularly designed for the optimization of noisy functions [24].

3. Hybrid Optimization Schemes

Hybrid optimization is a popular approach in the combinatorial optimization community where metaheuristic methods are combined in an effort to take advantage of the distinct strengths of different approaches [25]. More recently, metaheuristics have been combined with deterministic methods to form hybrids that perform global and local searches [26, 27].
3.1. Oracle-Based Hybrid: APPS-TGP

Some optimization methods have introduced an oracle to predict points at which a decrease in the objective function might be observed. These points are given in addition to the iterates inherent to the optimization method itself. Analytically, an oracle is free to choose points by any finite process. (See [28] and references therein.) The addition of an oracle is particularly amenable to pattern search methods like APPS. The iterate(s) suggested by the oracle are merely additions to the pattern. Furthermore, the asynchronous nature of the APPSPACK implementation makes it adept at handling the evaluation of the additional points. We use the idea of an oracle as a basis for creating a hybrid optimization scheme which combines APPS and the statistical emulator TGP. Like APPSPACK, this algorithm was designed for derivative-free optimization and problems which have less than 100 variables.

To create the APPS-TGP hybrid, we use the TGP statistical model as our oracle in the hopes of adding robustness and introducing some global properties to APPSPACK. When the oracle is called, the TGP algorithm is applied to the set of evaluated iterates in order to choose additional candidate points. In other words, APPSPACK is still optimizing as normal, but throughout the optimization process, the iterate pairs are collected. Then, the TGP algorithm considers the current set of iterate pairs and recommends one or more new iterates. These points are then evaluated to see if one of them is a new best point. If not, the point is merely discarded. However, if a TGP point is a new best point, the APPSPACK search pattern continues from its location.

3.2. Mixed Integer Hybrid: GA-APPS

To address mixed-variable, nonlinear optimization problems of the form

\[
\begin{align*}
\text{minimize } & f(x, z) \\
\text{subject to } & c(x, z) \leq 0 \\
& z_\ell \leq z \leq z_u \\
& x_\ell \leq x \leq x_u.
\end{align*}
\]

where \(c(x): \mathbb{R}^e \rightarrow \mathbb{R}^a\), we consider a hybrid of a GA and a direct search. We are interested in simulation-based functions and thus have designed a derivative-free method suitable for problems with less than 100 continuous variables and 20 or less integer variables.

A GA carries forward a population of points that are iteratively mutated, merged, selected, or dismissed. However, individuals in the population are not given the opportunity to make intergeneration improvements. In the real world, an organism is not constant throughout its life span, but instead can grow, improve, or become stronger. This behavior is introduced in the GA-APPS hybrid. The GA governs point survival, mutation, and merging as an outer iteration. Then, during an inner iteration, individual points are improved via APPS applied to the real variables, with the integer variables held fixed. Of course, to allow the entire population to grow as such, would be computationally prohibitive. Thus GA-APPS employs a ranking algorithm that takes in to account individual proximity to other, better points. The goal is to select promising individuals from distinct subdomains.

4. The Hydraulic Capture Problem

To demonstrate the applicability of APPS-TGP and GA-APPS, we consider the hydraulic capture (HC) problem, whose objective is to control the migration of a contaminant plume. To reach this goal, a set of pumping wells is installed with specified locations and rates that result in a flow direction and magnitude that defines the extent of the plume. Additionally, the well-field is designed to minimize associated costs. Figure 1 is an illustration of the problem. The oval shows the relative location of the plume within a square aquifer. The circles around the exterior of the plume represent the constraints that force the flow towards the interior of the plume and prevent migration. Groundwater flow in the natural system in the absence of wells is towards the northeast, indicated by the arrow in the lower left corner. A reasonable configuration of four wells–two extracting water inside the plume and two injecting water outside the plume–is also included where the wells are represented by stars.

In this study, two models of hydraulic capture are considered: transport-based concentration control (TBCC) and flow-based hydraulic control (FBHC). For both models, the decision variables are the number of wells, \(n \leq N\), the pumping rates, \(\{Q_i\}_{i=1}^n\), and the well locations, \(\{(x_i, y_i)\}_{i=1}^n\). Here, \(N\) is the maximum possible number of wells in the final design.
The objective function is the sum of the capital cost and the operational cost and can be stated as follows [29, 30]:

\[
\sum_{i=1}^{n} c_0 d_i + \sum_{Q_i < 0} c_1 Q_i^{b_1} (z_{gi} - h_{\text{min}})^{b_2} + \int_0^{t_f} \left( \sum_{i:Q_i < 0} c_2 Q_i (h_i - z_{gi}) + \sum_{i:Q_i > 0} c_3 Q_i \right) dt.
\]

The first term accounts for drilling and installing each well, and the second term represents the additional cost for pumps for extraction wells. The third term includes the lift cost associated with raising the water to the surface. Note that a negative pumping rate indicates a well is extracting and a positive pumping rates corresponds to an injection well. Injection wells are assumed to operate under gravity feed conditions. Specifically, \( c_j \) and \( b_j \) are cost coefficients and exponents, respectively, \( d_i = z_{gi} \) is the depth of well \( i \), \( Q_i^m \) is the design pumping rate, and \( h_{\text{min}} \) is the minimum allowable head.

The hydraulic heads, \( h_i \) for well \( i \), vary with the decision variables, and obtaining their values at each iteration requires a solution to equations that model saturated flow. Numerically, the simulator MODFLOW2000 [31] is used to find these values. In this study, the simulation is done on a 100x100x10 grid and the steady state solve for the hydraulic head values is on the order of seconds (roughly 5).

Constraints on the decision variables typically include bounds on the well capacities and the hydraulic head at each well location. We incorporate the two sets of inequalities

\[
Q_{i}^{\text{max}} \leq Q_i \leq Q_{i}^{\text{max}}, i = 1, ..., n
\]

\[
h_i^{\text{max}} \geq h_i \geq h_i^{\text{min}}, i = 1, ..., n
\]

where, \( Q_{i}^{\text{max}} \) is the maximum extraction rate at any well, \( Q_{i}^{\text{max}} \) is the maximum injection rate at any well, \( h_i^{\text{max}} \) and \( h_i^{\text{min}} \) are the maximum and minimum allowable head, respectively. In addition, a constraint on the net pumping rate to avoid dewatering the aquifer is given by

\[
Q_T = \sum_{i=1}^{n} Q_i \geq Q_T^{\text{max}},
\]

where \( Q_T^{\text{max}} \) is the maximum allowable total extraction rate.

The physical domain is a 1000 × 1000 × 30 meter unconfined aquifer. The boundary conditions, initial conditions, and plume development are described in [32, 33]. The starting point includes two extraction and two injection wells for a total of \( N = 4 \) candidate wells and initial pumping rates of \( \pm Q_i = 0.0064 \), just as in Figure 1.

4.1. Flow-based Hydraulic Control (FBHC)

Flow-based hydraulic control is a technique for plume containment that enforces head gradient constraints around the perimeter of the plume. In the particular model we consider, a head gradient constraint is formulated as a constraint on the difference in hydraulic head values at specified locations. Consider

\[
h_i^k - h_i^k \geq d, k = 1 \ldots M
\]
where $M$ is the number of head gradient constraints imposed around the boundary, $h_1, h_2$ are hydraulic head values at specified, adjacent nodes for each constraint $k$, and $d$ is the bound on the difference. This set of constraints can be used to enforce head gradients vertically or horizontally.

The FBHC approach to the HC problem is minimizing the objective function subject to (4), (5), (6) and (7). Use of this method is attractive because it is relatively inexpensive. However, it requires (7) to be calibrated to ensure that the contaminant plume is properly captured and to avoid excessive pumping [34].

### 4.2. Transport Based Concentration Control (TBCC)

A direct approach for plume containment is to impose constraints on the concentration at specified locations. This constraint can be expressed as

$$C_j \leq C_{j}^{\text{max}}$$

where $C_j$ is the concentration at some observation node $j$, and $C_{j}^{\text{max}}$ is the maximum allowable concentration. Evaluation of this constraint requires a solution to the contaminant transport equation. Numerically, this is obtained using the simulator MT3DMS [35], which on a 100x100x10 grid takes roughly ten times longer than calculating the hydraulic head values.

Solving the HC problem using the TBCC model is minimizing the cost function (3) with respect to (4), (5), (6), and (8). Therefore, the TBCC approach is relatively expensive. Moreover, objective functions and constraints involving concentrations are nonconvex in some situations [36, 37], making the minimization problem more difficult.

### 5. Numerical Experiments

The HC application studied in this work was proposed in [29] as a benchmarking problem for the optimization and hydrology communities. Table 1 summarizes the history of solving the HC problem. The abbreviation MINLP refers to mixed-integer nonlinear problem and NLP refers to nonlinear problem. The difference in these two formulations is whether or not the number of wells in the design is explicitly included as a decision variable or not. In the MINLP, the number of wells is stated as an integer variable while the well locations and pumping rates are real-valued. In the NLP, only the real-valued well locations and pumping rates are included, and a well is removed from the design if the

**Table 1: A Summary of the Optimization Methods Applied to the HC Problem**

| Optimization Method                  | Formulation | References |
|--------------------------------------|-------------|------------|
| Genetic Algorithm (GA)               | NLP         | [32]       |
| GA                                   | MINLP       | [33]       |
| Particle Swarm Optimization (PSO)    | NLP         | [38,39]    |
| PSO with surrogates                  | NLP         | [39]       |
| Implicit Filtering                   | NLP         | [32,33,40] |
| Pattern Search                       | NLP         | [32,41]    |
| Pattern Search (Multifidelity)       | NLP         | [32]       |
| Branch and Bound with surrogates     | MINLP       | [33,40]    |
| Conjugate Gradient                   | NLP         | [38]       |
| Simulated Annealing                  | NLP         | [38]       |

**Table 2: Solutions of the FBHC Problem Formulation**

| Optimization Method                  | Final Cost ($) | Simulator Calls (MODFLOW) |
|--------------------------------------|----------------|---------------------------|
| IFCCO                                | 23,421         | 385                       |
| DIRECT                               | 49,549         | 592                       |
| NOMAD (2N)                           | 50,797         | 168                       |
| NOMAD (N+1)                          | 50,574         | 94                        |
| APPS                                 | 25,018         | 111                       |
| APPS (+ surrogate)                   | 22,428         | 152 (+ 86 MT3D)           |
| GA (integer)                         | 24,934         | 930                       |
| GA (real)                            | 54,973         | 930                       |
| GA + Surrogate (MINLP)               | 23,491         | 22                        |

The HC application studied in this work was proposed in [29] as a benchmarking problem for the optimization and hydrology communities. Table 1 summarizes the history of solving the HC problem. The abbreviation MINLP refers to mixed-integer nonlinear problem and NLP refers to nonlinear problem. The difference in these two formulations is whether or not the number of wells in the design is explicitly included as a decision variable or not. In the MINLP, the number of wells is stated as an integer variable while the well locations and pumping rates are real-valued. In the NLP, only the real-valued well locations and pumping rates are included, and a well is removed from the design if the
pumping rate is near zero. Finally, it should be noted that all of the methods in Table 1 are single search methods, in that they suggest one new candidate at each iteration, except the genetic algorithm and the particle swarm optimization method which advance an entire set of points forward at each iteration. The later are considered population-based approaches. The term surrogate refers to the use of a stand-in for the true, expensive objective function.

In Table 2, we show the previously reported solutions [32, 33] of the FBHC formulation of the HC problem for a variety of traditional optimization methods. Of the methods in Table 2, only the integer-valued GA and the GA-surrogate MINLP approach were able to directly handle the number of wells in model. For the GA, once the correct number of wells is identified, the remaining function evaluation budget is exhausted while improving the real-valued variables. For the GA-surrogate MINLP approach, the correct number of wells in the optimal design is located quickly and then the optimization terminated based on a comparison to known published results. It is this termination criteria that keeps the number of calls to the MODFLOW simulator small. Note that solutions given in Table 2 vary despite the fact that all methods were given a specific starting point of 4 active wells and an initial cost of $78,587. The solutions which give the lowest costs correspond to optimization methods which successfully remove three wells from the original design.

To demonstrate the dependence of the results in Table 2 on the initial point, we created a space-filling design of 100 alternative starting points. Of these points, more than one-third were infeasible. Using the 62 feasible points as initial iterates, we reran APPS and IFFCO on the HC problem. As the results in Table 3 show, both algorithms were unsuccessful at finding the minimum in all cases. In contrast, the hybrids successfully found the optimal solution with only 1 well in the design. More specifically, the stand-alone algorithms were often unable to find an iterate superior to the initial guess. In other cases, they got stuck in local minima which correspond to the inability to remove wells from the current design. The hybrids were able to overcome these issues.

Because of starting point dependence, we opted to investigate the applicability of the hybrids to the HC problem given no starting point. We studied both the FBHC and the TBCC formulations. In both cases, the hybrids were able to find an optimal design with 1 active well. The final cost of the designs obtained are given in Table 4. For APPS-TGP, the total number of calls to the simulator was around 700 for each problem formulation, FBHC and TBCC. This may seem high in comparison to the results shown in Table 2. However, without a starting point, most of these methods quit without finding a reasonable solution. Moreover, when the real-valued GA was tested without a starting point, it exited without converging to a solution after over 1000 calls to the simulator. The GA-APPS hybrid was even more efficient making only about 500 calls to the simulator.

### 6. Discussion and Future Work

We have shown how a hybrid optimization approach can be effectively applied to a problem from water resources management. Specifically, we have demonstrated how a hybrid can overcome issues such as mixed integer variables and dependence on initial point. The largest component of complexity associated with managing water resources stems from variability and uncertainty in the water supply. Hybrid optimization techniques present an opportunity to uncover information about worst case scenarios. Discerning risk in coupled water management models is an
optimization problem, however, due to the prevalence of inherent uncertainties, traditional optimization methods may not be applicable or appropriate. Instead, we will consider next generation optimal analysis tools that will allow model calibration in the presence of uncertainty and design of management solutions that are robust to uncertainty. By combining derivative-free optimization methods effective and efficient for finding solutions to simulation-based optimization problems with statistical sampling techniques which provide information about the uncertainty in the design space and the reliability of solutions, we should be able to provide a deeper understanding of the significance and sensitivities of model uncertainties while still limiting overall computational costs.

7. References

[1] P. D. Hough, T. G. Kolda, V. Torczon, Asynchronous parallel pattern search for nonlinear optimization, SIAM J. Comput. 23 (2001) 134–156.
[2] T. G. Kolda, Revisiting asynchronous parallel pattern search for nonlinear optimization, SIAM J. Optim. 16 (2) (2005) 563–586.
[3] E. D. Dolan, R. M. Lewis, V. Torczon, On the local convergence properties of parallel pattern search, Tech. Rep. 2000-36, NASA Langley Research Center (2000).
[4] V. Torczon, On the convergence of pattern search algorithms, SIAM J. Optim. 7 (1997) 1–25.
[5] T. G. Kolda, E. Dennis, Jr., V. Torczon, Direct search methods on parallel machines, SIAM J. Optim. 1 (1991) 448–474.
[6] V. Torczon, PDS: Direct search methods for unconstrained optimization on either sequential or parallel machines, Tech. Rep. TR92-09, Rice Univ., Dept. Comput. Appl. Math., Houston, TX (1992).
[7] G. A. Gray, T. G. Kolda, Algorithm 856: APPSPACK 4.0: Asynchronous parallel pattern search for derivative-free optimization, ACM TOMS 32 (3) (2006) 485–507.
[8] J. Sacks, W. Welch, et al., Design and analysis of computer experiments, Statist. Sci. 4 (1989) 409–435.
[9] T. Santner, B. Williams, W. Notz, The Design and Analysis of Computer Experiments, Springer-Verlag, 2003.
[10] N. A. C. Cressie, Statistics for Spatial Data, revised edition, John Wiley & Sons, 1993.
[11] M. L. Stein, Interpolation of Spatial Data, Springer, New York, NY, 1999.
[12] T. Hemker, K. R. Fowler, M. W. Farthing, O. von Stryk, A mixed-integer simulation-based optimization approach with surrogate functions in model calibration in the presence of uncertainty and design of management solutions that are robust to uncertainty.
[13] M. Schonlau, Computer experiments and global optimization, University of Waterloo, Waterloo, Canada, 1997.
[14] R. B. Gramacy, H. K. H. Lee, Bayesian treed Gaussian process models, Tech. rep., Dept. Appl. Math & Statist., Univ. Calif., Santa Cruz (2006).
[15] R. B. Gramacy, H. K. H. Lee, Bayesian treed Gaussian process models with an application to computer modeling, J. Amer. Statist. Assoc. 102 (2008) 1119–1130.
[16] H. Chipman, E. George, R. McCulloch, Bayesian treed models, Machine Learning 48 (2002) 303–324.
[17] P. Green, Reversible jump markov chain monte carlo computation and bayesian model determination, Biometrika 82 (1995) 711–732.
[18] D. E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison Wesley, 1989.
[19] J. H. Holland, Adaption in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, MI, 1975.
[20] K. A. DeJong, W. M. Spears, An analysis of the interacting roles of population size and crossover in genetic algorithms (1990).
[21] J. Grefenstette, Optimization of control parameters for genetic algorithms, IEEE Trans. Sys. Man Cybernetics SMC-16 (1) (1986) 122–128.
[22] F. G. Lobo, C. F. Lima, Z. Michalewicz (Eds.), Parameter settings in evolutionary algorithms, Springer, 2007.
[23] J. J. Eshiman, P. J. F. E. D. Miller, C. T. Kelley, J. M. Gablonsky, IFICO: Implicit Filtering for Constrained Optimization, Version 2, Tech. Rep. CRSC-TR99-23, CRSC (July 1999).
[24] C. T. Kelley, Iterative methods for optimization, SIAM, Philadelphia, PA, 1999.
[25] E. Alba, Parallel Metaheuristics, John Wiley & Sons, Inc., 2005.
[26] J. L. Payne, M. J. Eppstein, A hybrid genetic algorithm with pattern search for finding heavy atoms in protein crystals, in: Proc. 2005 Conference on Genetic and Evolutionary Computation, ACM, 2005, pp. 377–384.
[27] T. Zhang, et al., A hybrid surrogate and pattern search optimization method and application to microelectronics, Structural Multidisc. Opt. 32 (2006) 327–345.
[28] T. G. Kolda, R. M. Lewis, V. Torczon, Optimization by direct search: New perspectives on some classical and modern methods, SIAM Revie 45 (3) (2003) 385–482.
[29] A. S. Mayer, C. T. Kelley, C. T. Miller, Optimal design for problems involving flow and transport in saturated porous media, Adv. in Water Resources 12 (2002) 1233–1256.
[30] A. S. Mayer, C. T. Kelley, C. T. Miller, Electronic supplement to “optimal design for problems involving flow and transport in saturated porous media”, Adv. in Water Resources 12 (2002) 1233–1256.
[31] C. Zheng, M. C. Hill, MODFLOW2000, The USGS Survey Modular Ground-Water Model User Guide to the LMT6 Package, the Linkage With MT3DMS for Multispecies Mass Transport Modeling (2001).
[32] K. Fowler, et al., A comparison of derivative-free optimization methods for water supply and hydraulic capture community problems, Advanc. Water Resources 31 (5) (2008) 743–757.
[33] T. Hemker, K. R. Fowler, M. W. Farthing, O. von Stryk, A mixed-integration simulation-based optimization approach with surrogate functions in water resources management, Opt. & Eng. 9 (4) (2008) 341–360.
[34] A. E. Mulligan, D. P. Ahlfield, Advective control of groundwater contaminant plumes: Model development and comparison to hydraulic control, Water Resources Research 35 (8) (1999) 2285–2294.
[35] C. Zheng, P. P. Wang, MT3DMS: A Modular 3-D Multispecies Transport Model for Simulation of Advection, Dispersion, and Chemical Reactions of Contaminants in Groundwater Systems (1999).
[36] A. Battermann, J. M. Gablonsky, A. Patrick, C. T. Kelley, K. R. Kavanagh, Solution of a groundwater flow problem with implicit filtering, Opt. & Eng. 3 (2002) 189–199.
[37] D. P. Ahlfeld, M. P. Sprong, Presence of nonconvexity in groundwater concentration response functions, J. of Water Resources Planning and Management 124 (1) (1998) 8–14.
[38] S. L. Mattot, A. J. Rabideau, J. R. Craig, Optimization of pump and treat systems using analytic element flow models, Advances in Water Resources 29 (5) (2006) 760–775.
[39] M. D. Parno, K. R. Fowler, T. Hemker, Framework for particle swarm optimization with surrogate functions, Tech. Rep. TUD-CS-2009-0139, Technische Universität Darmstadt, Department of Computer Science (2009).
[40] T. Hemker, K. R. Fowler, O. von Stryk, Derivative-free optimization methods for handling fixed costs in optimal groundwater remediation design, in: Proc. of the CMWR XVI - Computational Methods in Water Resources, 2006.
[41] G. A. Gray, K. R. Fowler, Approaching the groundwater remediation problem using multifidelity optimization, in: Proc. of the CMWR XVI - Computational Methods in Water Resources, 2006.