Comparison of Derivative-Free Optimization Methods for Groundwater Supply and Hydraulic Capture Community Problems

K. R. Fowler [a,*] J. P. Reese [b] C. E. Kees [c] J. E. Dennis, Jr., [d] C. T. Kelley [e] C. T. Miller [f] C. Audet [g] A. J. Booker [d] G. Couture [g] R. W. Darwin [e] M. W. Farthing [c] D. E. Finkel [i] J. M. Gablonsky [d] G. Gray [h] and T. G. Kolda [h]

[a] *Department of Mathematics and Computer Science, Clarkson University Potsdam, NY 13699-5815, USA*

[b] *School of Computational Sciences, Dirac Science Library, Florida State University, Tallahassee, FL 32306-4120 USA*

[c] *US Army Engineer Research and Development Station, ATTN: CEERD-HF-HG, 3909 Halls Ferry Rd., Vicksburg, MS 39180-6133, USA*

[d] *The Boeing Company, PO Box 24346, MS 7L 21, Seattle, WA 98124-0346, USA*

[e] *Department of Mathematics, North Carolina State University, Raleigh, NC 27695-8205, USA*

[f] *Department of Environmental Sciences and Engineering, University of North Carolina, Chapel Hill, NC 27599-7400, USA*

[g] *Ecole Polytechnique de Montréal - GERAD, C. P. 6079, Succ. Centre-ville, Montréal, Québec, H3C 3A7, Canada*

[h] *Sandia National Labs, Livermore, CA 94551-9159, USA*

[i] *MIT Lincoln Laboratory, 244 Wood Street, Lexington, MA 02420-9108 USA*

Abstract

Management decisions involving groundwater supply and remediation often rely on optimization techniques to determine an effective strategy. We introduce several derivative-free sampling methods for solving constrained optimization problems that have not yet been considered in this field, and we include a genetic algorithm for completeness. Two well-documented community problems are used for illustration purposes: a groundwater supply problem and a hydraulic capture problem. The community problems were found to be challenging applications due to the objective functions being nonsmooth, nonlinear, and having many local minima. Because the results were found to be sensitive to initial iterates for some methods, guidance is provided in selecting initial iterates for these problems that improve the likelihood of achieving significant reductions in the objective function to be minimized. In addition, we suggest some potentially fruitful areas for future research.

*K*eywords: Sampling methods, genetic algorithm, local minima, nondifferentiable objective function

* Corresponding author

Email addresses: kfowler@clarkson.edu (K. R. Fowler), jreese@scs.fsu.edu (J. P. Reese), Christopher.E.Kees@erdc.usace.army.mil (C. E. Kees), dennis@rice.edu (J. E. Dennis, Jr.,), tim_kelley@ncsu.edu (C. T. Kelley), casey_miller@unc.edu (C. T. Miller), charlesa@gerad.ca (C. Audet), andrew.j.booker@boeing.com (A. J. Booker), Gilles.Couture@gerad.ca (G. Couture), rwdarwin@unity.ncsu.edu (R. W. Darwin), matthew.w.farthing@erdc.usace.army.mil (M. W. Farthing), dfinkel@ll.mit.edu (D. E. Finkel), joerg.m.gablonsky@boeing.com (J. M. Gablonsky), gagray@sandia.gov (G. Gray), tgkolda@sandia.gov (and T. G. Kolda).

1 Introduction

Problems involving the design of groundwater supplies and contaminant containment and removal from subsurface systems can be difficult to solve in anything approaching an optimal fashion. The objective function of interest is often discontinuous, nonlinear, nonconvex, and replete with local minima. Moreover, evaluation of the objective function often requires the solution of an approximate numerical simulation model, which can be both expensive and subject to poor resolution of the physical phenomena of concern. Thus, the difficulties of achieving an optimal solution for groundwater supply and contaminant transport problems have their roots in physical aspects of the problems of concern, which are manifest in terms of a challenging set of mathematical characteristics.

Two additional impediments to the advancement of optimal design approaches exist for this class of problems. First, many potential methods exist, but most work focuses on only a small number of available methods for an idealized example problem, which may not have the same range of difficulty as the real class of problems of concern. Second, many optimization methods exist that have yet to be compared and in some noteworthy cases have yet to even be considered by the water resources community.

In response to these observations Mayer et al. [58] proposed a set of so-called "community problems" (CPs), which included a range of supply and remediation problems. The CPs offer a set of challenging and realistic applications to support methods comparison and advancement. An additional hope in introducing the CPs was that the existence of these problems would catalyze the introduction of new methods into the water resource field and perhaps unite subsets of the optimization community by stimulating the joint solution of interesting and difficult problems with a range of methods, which in total would be beyond the reach of any single research group in a reasonable length of time. Overall, it was hoped that the CP's would serve to hasten the rate of maturation of optimization methods for important water resources problems and improve the community's ability to arrive at effective designs for realistic problems.

Global solutions to the CPs have not yet been determined or even shown to exist. However, the CPs have received consideration in the literature [32, 33], and interest in these problems appears to be increasing in scope and frequency [39, 43, 44, 57]. Two areas in which the CPs have yet to be successful are the introduction of broad new classes of methods into the water resources field by experts in mathematical optimization and comparisons of significant sets of methods for the same problem.

Because the CPs are realistic, they possess many of the mathematical difficulties previously alluded to: they have nonsmooth, nonlinear, discontinuous, nonconvex objective functions that have many local minima. Derivative-based optimization methods are well known to perform poorly on problems with these characteristics, which has given rise to an increase in popularity of genetic algorithms (GAs) [38, 45, 46] and simulated annealing methods [51] in the water resources field [19, 27, 29, 63], which do not require the evaluation of derivatives of the objective function with respect to decision variables.

Such optimization problems arise in many other areas of science as well [9, 15, 72]. The mathematical optimization community frequently uses a class of deterministic methods which we refer to here as sampling methods to approximate the solution of such problems [9, 15, 72]. Sampling methods do not require derivatives of the objective function and in general rely upon a direct search of the decision space guided by a pattern or search algorithm. Deterministic sampling methods are a potentially important class of optimization methods which have received only limited use in the water resources literature [10, 32, 33, 39, 43, 68], and most such sampling methods have yet to be considered at all by the water resources community. These methods are different from commonly used sampling approaches such as GAs in that there is no randomness in the method, and there are rigorous convergence results. We include a very robust GA in the results of this work, so that the deterministic sampling methods can be compared to an approach that is more commonly used in water resources.

The overall goal of this work is to introduce and evaluate several members of an important class of optimization method by solving a subset of the CPs. The specific objectives of this work are: (1) to detail several sampling methods suitable for solving challenging water resources problems, such as the CPs; (2) to evaluate the performance of the sampling methods in terms of the solution achieved and computational effort required for a subset of the CPs as a function of the problem specification and initial conditions; and (3) to provide guidance for selecting an initial iterate for the CPs that improves the performance of the optimizers.

2 Model Problems

2.1 Overview

The CPs of concern in this work are a subset of a broad class of problems described by Mayer et al. [58, 59]. The CPs consist of model formulations and a wide range of physical domains, objective functions, and constraints for a total of 30 design applications. In the sections that follow, we describe the model problems of focus in this work and specify the hydrologic setting, objective function, constraints, simulator, and method details and links.

2.2 Model Problems

We consider two CPs, a water supply problem and a hydraulic capture problem, which are described in Mayer et al. [58, 59]. The water supply problem is also described by Fowler et al. [33]. The objective of the water supply problem is to minimize the cost to supply a specific quantity of water subject to a set of constraints. The cost involves installation and operation cost for a set of extraction wells subject to constraints on the net extraction rate, pumping rates, and hydraulic head. The decision variables are the $\{(x_i, y_i)\}_{i=1}^{n}$ locations and pumping rates $\{Q_i\}_{i=1}^{n}$, of the wells, and the number of wells n. We also considered a case in which only the locations of a fixed number of wells pumping at a specified rate were decision variables.

The objective of the hydraulic capture CP is to minimize the cost needed to prevent an initial contaminant plume from spreading by using wells to control the direction and extent of advective fluid flow. Several approaches exist to model the migration of a contaminant plume, including particle-tracking advective control, flow-based gradient control, and constraining a target concentration contour [2, 4]. We use a gradient control approach, which only relies upon information from a flow solve and is common in practice [3]. To capture the plume with the gradient control method, we impose constraints on head differences at certain points around the plume. The decision variables for this problem are the pumping rates $\{Q_i\}_{i=1}^{n}$, the well locations $\{(x_i, y_i)\}_{i=1}^{n}$, and the number of wells $n \leq N_w$. Here n is the number of wells in the final design. Since it is not clear how many wells will be needed to contain the plume, we start with a set of N_w candidate wells and include the number of wells implicitly as a decision variable.

2.3 Hydrological Setting

We consider a physical domain $\Omega = [0, 1000] \times [0, 1000] \times [0, 30]$ m. Flow in saturated porous media is described by

$$S_s \frac{\partial h}{\partial t} = \nabla \cdot (K \nabla h) + \mathcal{S}, \qquad (1)$$

where $S_s = 2.0 \times 10^{-1}$ 1/m is the specific yield, h is the hydraulic head, and K is the hydraulic conductivity. We consider homogeneous aquifers with

$K = 5.01 \times 10^{-5}$ m/s. Here the source term $\mathcal{S}$ represents a well model that satisfies

$$\int_{\Omega} \mathcal{S}(t) d\Omega = \sum_{i=1}^{n} Q_i. \tag{2}$$

The source $\mathcal{S}$ is where the decision variables enter the state equations. To describe the unconfined aquifer that applies to both CPs, we use the following boundary and initial conditions:

$$\left.\frac{\partial h}{\partial x}\right|_{x=0} = \left.\frac{\partial h}{\partial y}\right|_{y=0} = \left.\frac{\partial h}{\partial z}\right|_{z=0} = 0, t > 0 \tag{3}$$

$$q_z(x, y, z = h, t > 0) = -1.903 \times 10^{-8} \text{ m/s}, \tag{4}$$

$$h(1000, y, z, t > 0) = 20 - 0.001y \text{ m}, \tag{5}$$

$$h(x, 1000, z, t > 0) = 20 - 0.001x \text{ m}, \tag{6}$$

$$\mathcal{S}(x, y, z, t = 0) = 0.0 \text{ m}^3/\text{s}, \tag{7}$$

$$h(x, y, z, 0) = h_s. \tag{8}$$

Here q_z evaluated at $z = h$ is the Darcy flux out of the domain (the negative value specified represents recharge into the aquifer), and h_s is the steady state solution to the flow problem without wells. The ground surface elevation for the unconfined aquifer is $z_{gs} = 30$ m.

2.4 Objective Function

We consider a capital cost f^c to install a well and an operational cost f^o to pump a well, and we seek to minimize the total cost $f^T = f^c + f^o$. A negative pumping rate means that a well is extracting and a positive pumping rate means that a well is injecting. Our simulation time is $t_f = 5$ years. The objective function, as proposed in [58, 59] is given by

$$f^T = \underbrace{\sum_{i=1}^{n} c_0 d_i^{b0} + \sum_{i, Q_i < 0.0} c_1 |Q_i^m|^{b_1} (z_{gs} - h^{min})^{b_2}}_{f^c} \tag{9}$$

$$+ \underbrace{\int_0^{t_f} \left(\sum_{i, Q_i < 0.0} c_2 Q_i (h_i - z_{gs}) + \sum_{i, Q_i > 0.0} c_3 Q_i \right) dt}_{f^o},$$

where c_j and b_j are cost coefficients and exponents, $d_i = z_{gs}$ is the depth of well i, Q_i^m is the design pumping rate for which we use $Q_i^m = 1.5Q_i$ m^3/s, h^{min} is the minimum allowable head, and h_i is the hydraulic head in well i. Injection wells are assumed to operate under gravity feed conditions. In f^c the first term accounts for drilling and installing all the wells and the second term is an additional cost for pumps for the extraction wells. In f^o, the term pertaining to the extraction wells includes a lift cost to raise the water to the surface. The cost data is given in Table 1.

Table 1
Objective function parameters

Parameter	Value	Units
c_0	5.5×10^3	\$/m^{b_0}
c_1	5.75×10^3	\$/[(m^3/s)$^{b_1}\cdot$ m^{b_2}]
c_2	2.90×10^{-4}	\$/m^4
c_3	1.45×10^{-4}	\$/m^3
b_0	0.3	-
b_1	0.45	-
b_2	0.64	-
z_{gs}	30	m
d_i	z_{gs}	m
Q_i^m	$1.5Q_i$	m^3/s

2.5 Constraints

We constrain the pumping rates and hydraulic head for the objective function given in Eq. (9). The constraints are given by

$$Q^{emax} \leq Q_i \leq Q^{imax}, i = 1, ..., n, \tag{10}$$

$$h^{min} \leq h_i \leq h^{max}, i = 1, ..., n, \tag{11}$$

where Q^{emax} is the maximum extraction rate, Q^{imax} is the maximum injection rate, h^{max} is the maximum allowable head, and h^{min} is the minimum allowable head. Constraints (10) and (11) are enforced at each well. Constraint (10) reflects physical limits on the pumps and well design. Well designs are limited by the size distribution of the porous medium and the resulting size of the well screen. The upper bound in constraint (11) keeps the hydraulic head below the surface elevation, while the lower bound limits the allowable drawdown in a well. While onstraint (11) is a linear function of the pumping rates if an aquifer is confined (under the assumption that well losses are ignored), for this work we implement a nonlinear representation for an unconfined aquifer model. Moreover, constraint (11) is a highly nonlinear function with respect to the locations of the wells, since they are varying in the course of the optimization.

For the water supply CP, we define the total amount of water to supply as

$$Q_T = \sum_{i=1}^{n} Q_i \leq Q_T^{min}, \tag{12}$$

where Q_T^{min} is the minimum allowable total extraction rate. Although not specified in [58, 59],we require that the wells be situated at least 200 m from the Dirichlet boundaries, given by

$$0 \leq x_i, y_i \leq 800 \text{ m}. \tag{13}$$

We should note that optimzation landscapes for the water supply problem are shown in [33] which demonstrate the nonconvexity in (9) and also the disconnected feasible region defined by the constraints.

For the hydraulic capture CP, we constrain the net pumping rate with

$$Q_T = \sum_{i=1}^{n} Q_i \geq Q_T^{max}, \tag{14}$$

where Q_T^{max} is the maximum allowable total extraction rate.

In Mayer et al. [58] the authors leave it to the reader to choose the concentration that defines the plume boundary and to choose the constraint to capture the plume. For this work, we chose the 5×10^{-5} kg/m^3 concentration contour line as the boundary of the plume. We used a gradient control approach to ensure capture. Although a transport simulator was used to create the initial plume and to verify the effectiveness of the optimal point, only a flow simulator was required for the optimization. A gradient constraint was formulated as a constraint on the difference in hydraulic head values at specified locations, such that

$$h_j^k - h_{j+1}^k \geq d, k = 1 \ldots M, \tag{15}$$

where h_j, h_{j+1} are hydraulic head values at adjacent nodes and d is the bound on the difference. Here M is the number of gradient constraints imposed around the boundary. If h_j, h_{j+1} are aligned in the $x-$directions, dividing Eq. (15) by Δx and multiplying by the hydraulic conductivity K yields

$$K \left(\frac{h_j^k - h_{j+1}^k}{\Delta x} \right) \geq K \frac{d}{\Delta x}, \tag{16}$$

where the term on the left coincides with the x component of the Darcy velocity of the fluid. Constraint (15) results in a disconnected feasible region since this constraint in sensitive to the varying well locations. Table 2 shows the constraint data for the two applications.

Table 2
Constraint parameters

Parameter	Value	Units
Q_T^{min}	-3.2×10^{-2}	m^3/s
Q_T^{max}	-3.2×10^{-2}	m^3/s
Q^{emax}	-6.4×10^{-3}	m^3/s
Q^{imax}	6.4×10^{-3}	m^3/s
h^{min}	10	m
h^{max}	30	m
d	10^{-4}	m

The installation cost of an extraction well is roughly \$20,000 while the annual operational cost is approximately \$1,000. Having a pumping rate of zero indicates that there is no need to install a well. However, it is unlikely that a method will choose exactly zero. Instead, we set a threshold on the pumping rate and we remove a well from the design if the pumping rate falls below the threshold. The resulting objective function is discontinuous but the ability to remove a well can greatly reduce the cost of the design.

If in the course of the optimization, a well rate satisfies

$$|Q_i| < 10^{-6} \text{ m}^3/\text{s}, \tag{17}$$

then it is removed from the design space and not included in the flow simulation or cost calculations. In [43], the authors compare this approach with the multiplicative penalty coefficient from [64] and a branch-and-bound approach using a surrogate model on the hydraulic capture problem described above in § 2. The results obtained when using the inactive-well threshold and the approach from [64] did not differ significantly in terms of the optimal point found or the computational expense. Note that although the multiplicative approach leads to a continuous problem, finite-difference derivatives still were used in [64] due to the black-box formulation of the problem. By black-box formulation, we mean a decoupling of the numerical method used to approximate the physics of concern from the optimization approach that is used to approximate the design solution.

2.6 Simulator

Note that to evaluate Eq. (9) the values of the hydraulic head in the wells, h_i, must be computed for a given set of pumping rates $\{Q_i\}_{i=1}^n$ at locations $\{(x_i, y_i)\}_{i=1}^n$. Obtaining the head values requires a call to a groundwater flow simulator for a solution to Eq. (1). For this work, we used the U.S. Geological Survey code MODFLOW-96 [61, 62]. MODFLOW is a widely used and well supported block-centered finite difference code that simulates saturated groundwater flow.

A MODFLOW simulation that involves wells requires a well input data file containing the grid locations of the wells and the pumping rates. For this work, the locations and pumping rates are decision variables and hence change as the optimization progresses. Moreover, the optimization techniques used here output real-valued well locations, while the version of MODFLOW used expects node numbered grid locations for sources and sinks. Hence each function evaluation required rounding the well locations to grid points, which creates a step function, and writing a new well file containing the current well locations and pumping rates. Once the well file was created, a call to the MODFLOW executable simulated the flow system and the values of h_i were extracted to evaluate Eq. (9).

To generate the initial plume, as described in [59], we simulated plume development from a finite source for $t \in [-t_s, 0], t_s = 1.58 \times 10^8$ s, with a source concentration of 1 kg/m^3 located physically in the region bounded by $[(200, 225); (475, 525); (h, h{-}2)]$ m. To simulate contaminant transport we used MT3DMS [73], a widely used contaminant transport package that is designed to interface with MODFLOW flow.

3 Optimization Methods

All of the optimization methods we consider in this paper use only function values to guide the minimization of Eq. (9). By this, we mean that the optimization is controlled by evaluating the objective function and constraints at points in design space, and those evaluations are used to decide what to do next. All but the GA have an explicit resolution or level, which changes as the optimization progresses. In most of the methods, this means that the size of a stencil or pattern upon which the search is based is reduced.

The optimization methods considered in this paper have several common features and strategies, which we use as a way to describe the methods. This set of methods was chosen from a wide range of applicable derivative free methods with the goal of testing algorithms each with different strengths, which we emphasize in the discussion that follows. We will not describe detailed technical features or convergence theory of these method, but refer the reader to the papers we cite below.

The methods are:

- APPS (Asynchronous Parallel Pattern Search) [47, 52, 53, 54] from Sandia National Laboratories;
- DE (Boeing Design Explorer) [1, 9, 13, 18] from the Boeing Company;
- DIRECT (DIviding RECTangles) [48] with implementations from North Carolina State University [30, 34, 35, 36] and others [11];
- IFFCO (Implicit Filtering for Constrained Optimization) [16, 37, 50] from North Carolina State University;
- NOMAD (Nonlinear Optimization for Mixed vAriables and Derivatives) [8, 17] from Rice University, the Air Force Institute of Technology, and the Ecole Polytechnique de Montréal; and
- GA (NSGA-II: Non-dominated Sorting Genetic Algorithm) [25, 74] from Kanpur Genetic Algorithms Laboratory, Indian Institute of Technology.

Of these, only DE is a commercial product. The other codes we used in this paper are easily available. All of these codes are well-documented, being updated, and the versions the reader may download could well be more general, robust, and efficient than the versions we describe here.

3.1 Search-Poll Paradigm

We begin the discussion of the optimization methods by first considering the mesh/stencil based methods: NOMAD, IFFCO, DE, and APPS. These methods use a conceptual discretization of the space of decision variables into a stencil or pattern of points. For IFFCO and APPS, the discretization is only local, i.e. defined only near the current approximation to the solution, but for NOMAD and DE, it is global, i.e. a grid defined on all of design space. This distinction can be explained in terms of a search-poll paradigm. The general idea is that a search step allows a great deal of freedom in seeking a better point with the understanding that if this fails to produce improvement, then the algorithm will fall back on a local poll of nearby points before allowing a smaller step to be tried.

For our purposes, a *mesh* is an iteration dependent, global discretization of the decision variable space. The meshes in the structured algorithms must satisfy certain technical conditions [7, 56] in order for the convergence theory to hold. The most important of these is that the directions in the stencil be a positive spanning set: a set of $n + 1$ or more directions whose nonnegative linear combinations span the decision space [56]. Results here are given for versions of NOMAD using $n + 1$ and $2n$ such directions. NOMAD, IFFCO, and APPS sample points in a *search* phase seeking a better point (i.e., one with a lower objective function value).

If the search succeeds, the mesh/stencil stays the same size for the next iteration. But, if the search fails to find a better point, then a local stencil/mesh search is carried out (the *poll* step) to see if a better point can be found by steps of the current size in the current positive spanning set of directions. If so, the current sizing parameter stays the same or is increased for the next iteration. But otherwise, the sizing parameter is decreased, usually by a factor of two. DE uses surrogates during the search step to find promising points. These are found by solving several local optimization problems from different starting points on the surrogates, and by finding surrogate refinement points. The solutions of the local optimizations and the surrogate refinement points are then evaluated with the simulation. This is the search phase of DE. If no progress has been made as measured by a filter, DE performs a poll step on the $2n$ coordinate directions [9].

The situation is different for IFFCO. IFFCO *begins* an iteration by evaluating the function at all the points required for a poll step. This sounds profligate, but it is not. The idea is that this information provides an estimate of the gradient of the objective function. The Hessian estimate is provided by a quasi-Newton update [26, 50]. This provides a quadratic surrogate, or model, of the objective function that is used in the search step. If the search step fails, then the complete polling information is already at hand. If a better point was found in the poll, then it is taken, and otherwise the stencil size is reduced for the next iteration. In other words, the algorithm uses a large finite difference stencil at first and then reduces the stencil size as the optimization progresses to hopefully acheive the fast convergence of the underlying quasi-Newton method. APPS is a parallel generating search set method that, when running in parallel, reduces the step-size independently along each direction, enabling efficient utilization of parallel resources. For bound constrained problems, such as those considered in this paper, APPS and IFFCO use the coordinate directions to define the search pattern, and this gave them an advantage in some of the results we report in § 4.

3.2 Unstructured Searches

DIRECT is a deterministic sampling algorithm that was first introduced in [48], motivated by a modification to Lipschitzian optimization. It was created in order to solve difficult global optimization problems with bound constraints and a real-valued objective function. DIRECT systematically searches for the minimum by dividing the feasible region into hyper-rectangles. The algorithm continues the search by choosing some of the hyper-rectangles to sub-divide; a decision that is based on the size of the hyper-rectangle, and the value of the function at its center. After hyper-rectangles are subdivided, the new centers are sampled and a new iteration begins. The algorithm terminates when a given budget of function evaluations has been exhausted. A modified version of DIRECT, named DIRECT-L [30], is utilized in this study. The DIRECT-L algorithm biases DIRECT towards local searches (at the expense of the global search), and can improve convergence rates for some problems. A detailed description can be found in [30, 35, 48].

A genetic algorithm is a search technique that is inspired by evolutionary biological processes such as mutation, inheritance, selection, and crossover [38]. In this work, we use the non-dominated sorting genetic algorithm NSGA-II, which is described in [20, 23, 25, 74]. Although a variety of genetic algorithms exist, the NSGA-II has been applied to both single and multi-objective problems for a wide range of applications including those in water resources management [66]. Here, we consider a single-objective use of the NSGA-II, which incorporates both real- and binary-coded variables, and uses binary tournament selection [21]. For the real-coded variables, the simulated binary crossover (SBX) operator [21, 22] with polynomial mutation is used while single-point crossover with bitwise mutation are used for binary-coded variables.

Parameters like the population size, number of generations, as well as the probabilities and distribution indexes chosen for the crossover and mutation operators effect the performance of a GA [60, 67]. The population size of 30 was the lower bound of the suggested range, while a maximum of 30 generations were allowed. The crossover and mutation operator parameters were chosen based on the performance of NSGA-II for a multi-objective test problem with several local Pareto-optimal fronts [24]. We performed a limited number of experiments with other crossover and mutation operator parameter settings, but found no combination that gave better performance across the test problems considered here.

3.3 Constraints

We differentiate between three classes of constraints present in the problems. There are bound constraints such as the limits on the pumping rates and locations given by Eq. (10) and (13). All the methods here incorporate bound constraints into definition of the algorithms.

The next class of constraints are simple linear constraints such as the limit on the net pumping rates given by Eq. (12) or Eq. (14). The versions of the stencil-based codes (APPS, DE, IFFCO, and NOMAD) used in this paper handle linear and nonlinear constraints by the simple expedient of rejecting any infeasible point as a possible next iterate without even evaluating the objective function. DIRECT also handles linear constraints in this way. In the

DE search step, the local optimization method on the models enforces linear constraints, and surrogate refinement points are also required to satisfy linear constraints. In the poll step, points violating linear constraints are handled the same way as by the other stencil based methods.This approach has been rigorously justified for linear constraints in the cases of NOMAD, DE, DIRECT, and APPS, given certain conditions on the pattern [14, 30, 31, 42, 55]. This result is almost surely true for IFFCO as well, but the analysis has not been reported.

There are also more complex nonlinear inequality constraints as in Eq. (11). Of the specific codes tested here, only DIRECT [30, 31], can be shown rigorously to work for nonlinear constraints by the above simple "barrier" approach of declaring an infeasible point to be unacceptable as a next iterate. Rigorous analysis, while not directly linked to performance, is important for understanding behavior of the algorithm. Such analysis can assist in evaluation, tuning, and debugging by eliminating possible failure modes. The results in [30, 31] use the analytical methods of [8].

Another way some of the methods treat nonlinear constraints is by replacing the objective function by a penalty function, i.e., by minimizing an unconstrained objective consisting of the objective function plus a penalty constant times a measure of the aggregate constraint violation. The choice of the penalty constant is problematic, especially for the methods here which do not use any constraint derivative information. Furthermore, this procedure vaguely requires the penalty function to be "sufficiently large" for the ℓ_1 norm of the constraint violations, and it is required to increase to infinity for the ℓ_2 norm aggregate constraint violations. Both the GA and DIRECT used this approach to handle constraints for this work. DE uses surrogates of the constraints in the search phase for both the local search on the surrogates, and the surrogate refinement.

All of the methods have some way to deal with a point at which the evaluation of the constraints or objective function fails. These methods are described in the references for the various algorithms, and are important parts of the optimization, since these failures are not uncommon. Such a failure could be caused by a failure to converge for an internal iteration in the simulator, for example. While we saw no such failures in the computations reported in this paper, we used this feature in the codes to efficiently handle the linear constraints on the pumping rates. That is, if an optimizer selects a set of pumping rates that does not meet the net pumping rate constraint, then the simulator was not called and the function evaluation was declared a failure.

3.4 Surrogates

A surrogate is a function that can be used as a stand-in during the search phase for the expensive objective function and constraints used to define the optimization problem. The idea is to use a surrogate that is much cheaper to evaluate that the real objective function. We do not call it an approximation because this implies some effort to make the surrogate approximate the function, and yet popular surrogates can not be guaranteed to be close to the function they replace in the search phases of the optimization. NOMAD can be used with surrogates, but this was not done for the results in this paper.

DE uses Kriging models as surrogates. To build these models, the simulation is evaluated at points given by a DACE (Design and Analysis of Computer Experiments)[12, 13] experiment. These models interpolate the true responses at points used to build the model, and indicate global trends over the full design space. DE updates the Kriging models with new data during an optimization. To prevent problems with conditioning when points accumulate near each other during an optimization a modification of the standard Kriging models is used in DE [1].

At the other end of the spectrum are the local Taylor series based quadratics used by quasi-Newton methods. There, one matches the true gradient with the surrogate gradient at the current point, and so the surrogate, which is generally called a local model in this case, resembles the objective function with increasing accuracy as the distance to the current point decreases.

Between these two extremes lie the quadratic surrogates used by IFFCO. IFFCO begins each iteration by evaluating the objective on a stencil defined by a current stencil size parameter and a positive spanning set of directions. These values are used to build the gradient of the quadratic surrogate. The Hessian is updated by either the SR1 or BFGS update [26]. As the iteration closes in on an optimal solution, the stencil size becomes smaller, and so the approximation becomes more like the Taylor series local model, when it exists.

3.5 Termination

All of the methods will terminate if a budget of function evaluations has been exhausted. In most instances, the total number of function evaluations is checked against the budget only after an iteration is complete, so the final number of function evaluations can be over the budget.

The stencil-based methods (APPS, IFFCO, NOMAD, DE) also terminate when the stencil reaches a minimum size. The defaults in the codes vary.

However, for this application, the codes were set to terminate when the stencil size was equivalent to the resolution of the spatial grid in the simulator.

3.6 Scaling

The components of the vector of decision variables u can differ significantly in magnitude. The five orders of magnitude difference between pumping rates and physical locations is one example. Such poor scaling can cause the optimization to stagnate well before finding a good solution. To remedy this, most of the methods scale u to a reference domain. For example, IFFCO, APPS, and DIRECT scale u so that all lower bounds are 0 and all upper bounds are 1. DE requires the user to provide an appropriate scaling, both for the variables, and also for the objective and all constraints.There can also be scaling issues with the constraints, but we did not see problems with constraint scaling in this work. As a general rule, one should try to scale the constraints so that the constraint violations reflect the relative amounts of infeasibility one feels are appropriate for the particular initial iterate.

4 Results

4.1 Overview

A variety of numerical experiments were performed to evaluate the derivative-free optimization methods of focus in this work. Specifically, we considered the water supply CP and the hydraulic capture CP. We also considered alternative approaches to posing the optimization problem by varying the set of design parameters and the initial iterates. Initial iterates were of interest because of the complex nature of the feasible solution space and the need of APPS and IFFCO for a feasible initial iterate. Although in practice, determining a feasible initial iterate can prove challenging since the knowledge of a feasible solution is not even known a priori, we do not consider it a significant limitation for the applications of concern here.

When comparing optimization methods, multiple aspects of the solution are important, which complicates such comparisons. An obvious metric of interest is the quality of the solution obtained, which in this case is measured by the value of the cost function that is minimized for each application. Exact global solutions to the community problems have not been published although local minima have been shown to exist for the water supply problem [33]. Assuming all methods achieve the same minimum cost, the computational effort needed to achieve the solution, which we measure in terms of function evaluations, is another metric of primary importance. For difficult problems such as those considered in this work, it is not expected that all methods will obtain the identical cost function value, which complicates the comparison of methods. To aid such comparisons, we present results in terms of the cost profile as a function of the number of function evaluations.

In addition, the initial feasible iterate, problem formulation, and method specific settings can all affect the results achieved. We examined the effect of the initial iterate and the problem formulation, but we avoided detailed tuning of settings in the methods. Changes in parameter settings for the individual methods or changes in the algorithms for any of these methods could, of course, change the results. Indeed, one of the objectives of this work was to catalyze such algorithmic advancements. Since APPS was the only parallel implementation of the algorithms tested, we do not compare run times, rather the parallel version of APPS was used only to demonstrate the asynchronous nature of the search phase.

We used MODFLOW to simulate the unconfined flow field for the domain described in § 2.3 using an equally spaced $50 \times 50 \times 10$ grid in the $x \times y \times z$ directions. Wells were simulated by assigning a stress to a set of grid blocks corresponding to the rounded location of the wells. The initial conditions for both problems required a steady state simulation for an unconfined flow problem, which is depicted in Figure 1. The head field depicted in this figure corresponds to the value in the fourth layer from the top of the domain, since under these boundary conditions the top three layers are dry. These dry layers were included in the model to allow for local increases in head due to injection, which was needed to resolve the initial conditions for the hydraulic capture CP.

4.2 Water Supply Problem

4.2.1 Five-Well Design

It is easy to prove by examination of the objective function and constraints that a feasible solution to the water supply CP requires a minimum of five wells, and the number of wells in the optimal solution is the minimum needed to obtain a feasible solution. For $N_w = 5$, the extraction rates of all wells must be $\{Q_i\}_{i=1}^{5} = -0.0064$ m^3/s to satisfy the constraints. Based upon these observations and the need of APPS and IFFCO for a feasible initial iterate, we searched a five-well design space using the fixed MODFLOW discretization summarized above. We found feasible solutions in this space, but the feasible region was sparse and the landscape complicated. Because the number of

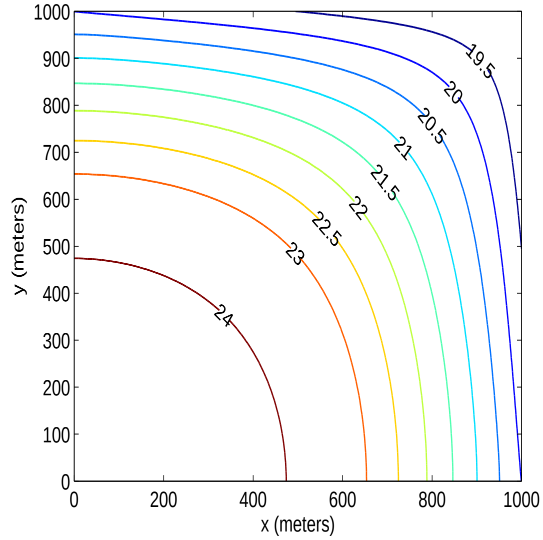


Fig. 1. Steady state head distribution in meters.

wells and their pumping rates are fixed in this scenario, ten decision variables remain, which are the optimal locations $\{(x_i, y_i)\}_{i=1}^{5}$ of the wells.

Table 3 shows the function value obtained at the initial iterate, the minimum function value, and the number of function evaluations performed by each optimizer before the termination criteria were met. Table 4 shows the initial (x, y) coordinates for the five wells and the optimal locations, which have been rounded to the nearest grid location. Figure 2 shows the value of the objective function as a function of the number of function evaluations for each of the methods considered. We used the initial iterate for APPS and IFFCO, which required such a point. We also used the initial iterate for the two versions of NOMAD, which was not strictly required. DIRECT-L, GA, and DE did not rely upon the initial iterate.

Table 3
Optimal solutions for the five-well water supply CP

Method	Minimum f	Number of Function Evaluations
Initial Iterate	\$ 127,421	—
IFFCO	\$ 125,129	165
DIRECT-L	\$ 125,085	648
NOMAD(2N)	\$ 124,386	539
NOMAD(N+1)	\$ 124,389	346
GA	\$ 124,386	925
DE	\$ 125,598	510
APPS	\$ 124,427	117

Table 4
Optimal well locations for the five-well water supply CP

Initial Iterate (m)	IFFCO	DIRECT-L	NOMAD (2N)	NOMAD (N+1)	GA	DE	APPS
x(1) 360	160	200	160	160	160	180	200
y(1) 720	800	800	800	800	800	780	800
x(2) 780	800	800	800	800	800	780	800
y(2) 7800	800	800	800	800	800	800	800
x(3) 680	320	560	460	460	460	580	520
y(3) 680	800	800	800	800	800	740	800
x(4) 200	680	280	800	800	800	780	800
y(4) 200	800	380	460	440	460	340	480
x(5) 720	760	800	800	800	800	680	800
y(5) 340	240	200	140	140	120	200	140

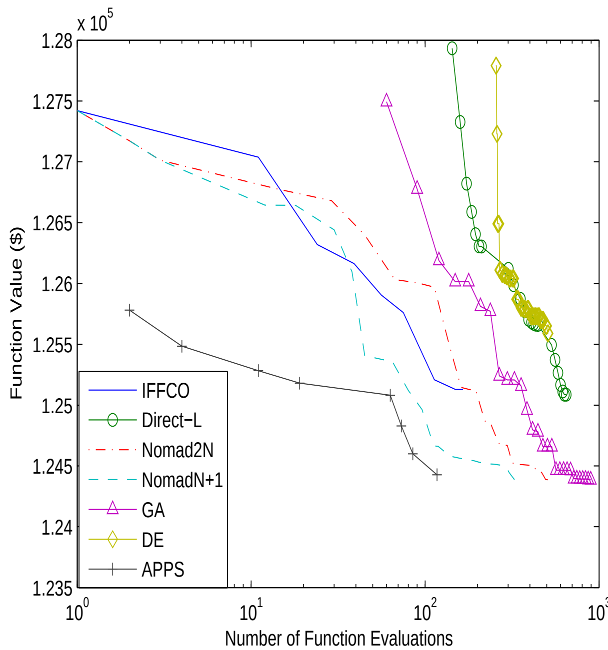


Fig. 2. Solution profiles for five-well water supply CP. Note that the GA is not seeded.

4.2.2 Five-well Discussion

If viewed in terms of the total cost, the optimal solutions differed by less than 1%. This is a bit misleading because the fixed cost represented about 80% of the total cost. The initial iterate was about 10% higher in operational cost than the best solution found. The magnitude of these numbers would of course be shifted if the design life was increased or if a different initial iterate was used. Thus care is needed in interpreting these results. Nonetheless, some aspects of these results warrant note.

First, NOMAD and the GA achieved the lowest cost, followed closely by APPS. DE returned the highest cost, but still a good design by most reasonable standards. IFFCO stagnated prematurely, exhausting its set of stencil sizes. Further numerical experiments showed that a single restart would remedy that stagnation. DIRECT and Design Explorer also terminate prematurely but did result in solutions with the wells moved to the prescribed head boundary conditions.

Second, the optimal designs returned by NOMAD and the GA differ by one grid block for two of the wells, but are otherwise the same. The APPS design, while nearly as good as the best solution found, differed in the location of some of the wells compared to NOMAD. This is consistent with our observation that within the feasible region, the objective function was not highly sensitive to the location of the wells, meaning relatively flat regions exist in the feasible region landscape. Designs returned by the other methods varied as well. However, APPS used the smallest number of function calls before exhausting its stencil budget. This demonstrates the strength in APPS asynchronous structure.

Third, the solution profiles shown in Figure 2 show that up to the termination point of APPS the lowest cost design for a given number of function evaluations was achieved by this method. NOMAD and the GA achieved slightly lower objective function values but required significantly more function evaluations to do so. It can also be observed from this figure that in general the solution profiles for the methods that were seeded with a feasible initial iterate showed a more efficient solution than the methods that were not seeded with the initial iterate. By efficiency we mean the objective function value achieved for a specified number of function evaluations. This appears to be a slight advantage over DIRECT, DE, and GA which spend a considerable time searching for the feasible region during the preliminary phases of the optimzation.

4.2.3 Six-Well Design

We also considered a problem formulation that did not rely on the observation that five wells operating at the maximum extraction rate satisfies the water supply constraint exactly. Specifically, we considered the case in which the initial design consisted of six wells. In order to find a solution that was competitive with the five-well design, it would be necessary to eliminate one of the wells from the design, thus recovering the discrete capital cost. This proved to be an extremely difficult problem. Rather than reporting the results in detail for this work, we note some general findings.

The six-well formulation is particulary sensitive to the well locations in terms of violating the drawdown constraint given by Eq. (11). To understand the properties of a good initial iterate, we first considered the water supply CP with the confined aquifer as described in [58]. In contrast, this hydrological setting results in fewer constraint violations but maintains the challenge of identifying the five-well solution in the course of the optimization [33]. We generated a suite of other initial iterates using the DIRECT algorithm and a cluster analysis. To generate that set of starting points, the problems were reformulated so the objective was feasibility. Given a function evaluation budget of 50,000, DIRECT found approximately 5,000 feasible points. This set of data was then analyzed with the Agglomerative Nesting (AGNES) clustering algorithm to determine a smaller set of representative feasible points [49]. All methods that did not rely upon or were not supplied with an initial iterate failed to find a five-well solution. For the methods that were provided with an initial iterate, (APPS, IFFCO, and NOMAD) no method succeeded in finding a five-well solution for most of the 135 initial iterates that we investigated.

It was straightforward to include an integer variable in the GA formulation, the value of which determined if five or six wells were active and which well was excluded in the five well case. With this simple change, the GA was able to obtain a good five-well solution when seeded with a good initial iterate.

The notion of a good initial iterate requires some discussion. Because of the complexity of the landscape and the need to eliminate one well from the six-well initial design, naive attempts at initial iterates will fail in most cases; we tried 135 cases that failed in this work. If one starts with pumping rates that meet the minimum quantity constraint, which seems sensible, then in order to eliminate a well five of the rates must increase and the remaining rate must approach zero. This will cause an increase in the objective function that cannot be characterized as high frequency, low amplitude noise. Thus sampling methods are not likely to find the solution region while searching among the small, disconnected feasible regions.

Based on these observations, our approach to providing a good initial iterate is to specify at least five of the wells with the maximum possible rate. Thus to eliminate a well from the design only one of the wells needs to be reduced in the rate of pumping. Put another way, such an iterate positions one on a continuous portion of the feasible region with a downward path toward the optimal region. We tried a small set of initial iterates that met this criterion for the unconfined six-well formulation and in each case APPS and IFFCO returned a good design while NOMAD succeeded in some of the cases. These solutions were obtained typically within 200 function calls for APPS, IFFCO, and NOMAD while the GA was usually higher. This supports the observation made in the five-well discussion that methods using a feasible initial iterate, which is chosen with attention to the nature of the objective function, can result in a more efficient optimization performance.

The six well water supply formulation is more challenging since minimization relies on satisfying the inactive-well threshold and thereby removing a well from the design space. This challenge is also addressed in [33] in which initial designs containing up to 16 candidate wells were considered for the water supply CP. The GA is the only optimizer used in this study that has a direct method for handling the integer variable in determining the appropriate number of wells. However, given an initial iterate as described above, APPS, IFFCO, and NOMAD are competitive.

4.3 Hydraulic Capture Problem

4.3.1 Baseline Iterate

For the hydraulic capture problem, we sought to minimize Eq. (9) over all the possible decision variables, $n, \{Q_i\}_{i=1}^n$, and $\{(x_i, y_i)\}_{i=1}^n$. We imposed $M = 5$ head difference constraints in Eq. (15) around the perimeter of the plume in the fourth layer from the top of the domain and used a value of $d = 10^{-4}$ m/s. We used a 100×100 grid for this MODFLOW implementation. The relative (x, y) locations of the constraints are found in Table 5 and are shown in Figure 3. Previous work has also used a similar gradient-based constraint approach [32, 39, 43, 57].

Table 5
Gradient constraint locations

x (m)	y (m)
180	730
240	770
330	740
390	650
380	540

The initial iterate for the hydraulic capture CP is shown along with the initial plume location from the MT3D simulation in Figure 3. This iterate included two injection and two extraction wells for $N_w = 4$ candidate wells. The injection wells were initialized at $Q^{imax} = 0.0064$ m^3/s and the extraction wells were initialized at $Q^{emax} = -0.0064$ m^3/s. We located the extraction wells within the interior of the plume and the injection wells down gradient of the plume as noted in the figure and listed in the first column of Table 6. The initial iterate was verified to meet the constraints and the objective function of the design was evaluated. The (x, y) coordinates of the initial well design are found in Table 7.

Table 6 shows the function value obtained at the baseline iterate, the minimum function value and the number of function evaluations performed by each optimizer before the termination criteria were met. Table 7 shows the initial

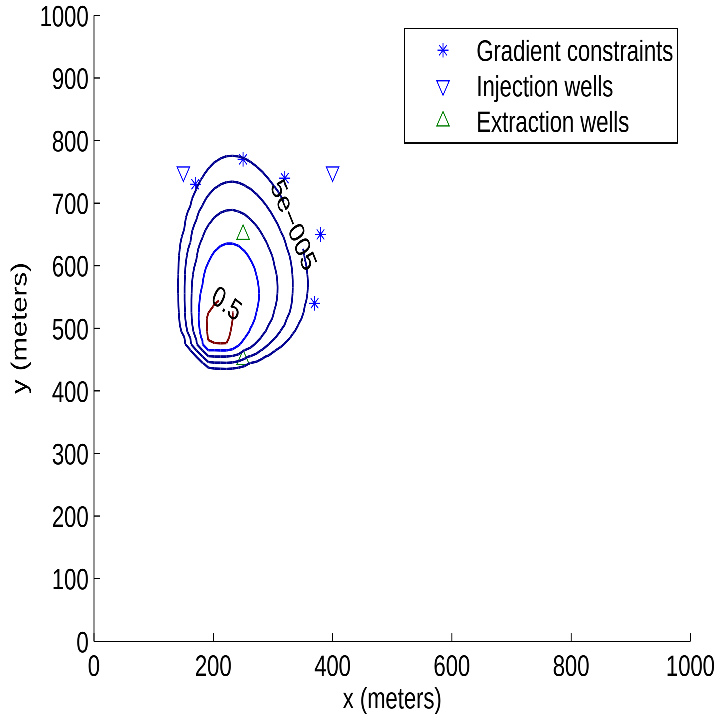


Fig. 3. Initial plume, constraint locations, and iterate.

(x, y) coordinates and pumping rates for each well and the resultant design returned by each method with the locations rounded to the nearest grid block. Figure 4 shows the value of the objective function as a function of the number of function evaluations for each of the methods considered. We used the initial iterate for APPS and IFFCO, which required such a point. We also used the initial iterate for the two versions of NOMAD and the two versions of the GA, which was not strictly required. The two versions of the GA correspond to formulations with and without integer variables for disabling wells. The mixed-integer formulation was implemented as described for the water supply CP. DIRECT-L and DE did not rely upon the initial iterate.

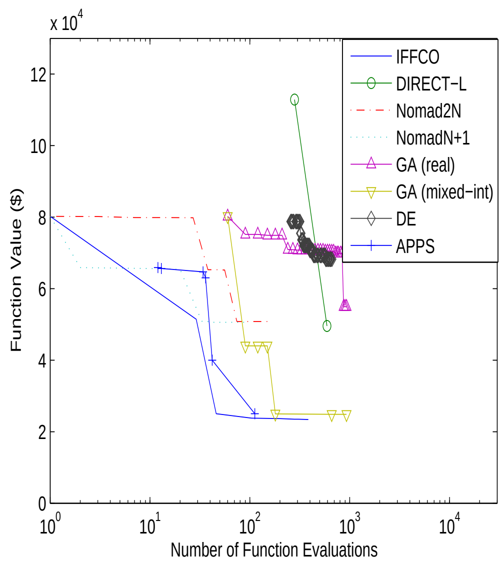
4.3.2 Hydraulic Capture Discussion

Results summarized in Table 6 show that the lowest cost designs were found by IFFCO, the mixed-integer GA, and APPS, respectively with all three designs significantly better than the baseline initial iterate because they were able to reduce the design to a single pumping well. The IFFCO design was the lowest cost because it reduced the pumping rate significantly below the value originally specified, where the GA and APPS did not accomplish this design aspect. The location of the single well was, however, similar for all three of these methods. For those three designs, the well locations differ by at most two MODFLOW grid points in either the x or y direction. The solutions with

Table 6
Baseline solutions for the hydraulic capture CP

Method	Minimum f	Number of Function Evaluations
Initial Condition	\$ 80,211	—
IFFCO	\$ 23,421	385
DIRECT-L	\$ 49,549	592
NOMAD(2N)	\$ 50,797	168
NOMAD(N+1)	\$ 50,574	94
GA (real)	\$ 54,973	930
GA (mixed-int)	\$ 24,870	930
DE	\$ 68,238	665
APPS	\$ 25,018	111

Fig. 4. Solution profiles for hydraulic capture CP. Note that the GA is seeded.



one extraction well are comparable to those found in the literature for this problem [32, 39, 43, 57]. DIRECT-L, NOMAD, the real GA, and DE all had solutions with much larger objective function costs due to the inclusion of at least two wells in the final design. Figure 4 shows that IFFCO was the most efficient solution method followed by APPS and the GA. All solution profiles show the discrete nature associated with the reduction in capital cost associated with eliminating a well from the design.

The hydraulic capture problem has the additional difficulty of enforcing the head gradient constraint to contain the plume while simultaneously removing wells to decrease the installation cost. Only IFFCO, APPS, and the mixed-integer GA were able to find the optimal design. The GA using a strictly real-variable formulation, that is using (17) to handle the number of wells instead of using integer variables, was unable to find the optimal design for

Table 7
Baseline solution details for hydraulic capture CP

Method	x (m)	y (m)	Q (m^3/s)
Initial Iterate	150	750	0.0064
	400	750	0.0064
	250	650	-0.0064
	250	450	-0.0064
IFFCO	—	—	0.0
	—	—	0.0
	260	640	-0.0053
	—	—	0.0
DIRECT-L	170	170	-0.0043
	—	—	0.0
	170	500	-0.0043
	—	—	0.0
NOMAD (2N)	—	—	0.0
	—	—	0.0
	380	730	-0.0064
	290	310	-0.0064
NOMAD (N+1)	—	—	0.0
	—	—	0.0
	330	650	-0.0064
	650	270	-0.0064
GA (real)	—	—	0.0
	350	910	0.0057
	980	740	0.0020
	250	610	-0.0060
GA (mixed-int)	—	—	0.0
	—	—	0.0
	—	—	0.0
	240	620	-0.0063
DE	230	670	-0.0048
	300	770	0.0024
	570	450	0.0020
	660	370	0.0048
APPS	—	—	0.0
	—	—	0.0
	250	650	-0.0064
	—	—	0.0

the hydraulic capture problem even with a seeded initial iterate. In terms of reliablity, the sampling methods will return the same result for the same initial iterate, which is not the case for a GA since randomness is inherent in the algorithm. The sensitivity of the GA to the random seed is also a problem.

Given the same initial iterate, APPS and IFFCO were able to choose the appropriate number of wells and terminated based on a scaling budget, using fewer function calls than the GA. NOMAD, however, turned off two wells, and did not turn off the third well. The reason for this is that NOMAD terminated prematurely based on a small stencil size. A larger initial stencil or one centered at a point where all wells are pumping at their maximum rates might correct this problem.

DIRECT terminated with a suboptimal solution when its budget of function evaluations had been exhausted. Since DIRECT will sample densely in design space if given an infinite budget, DIRECT would have found a one-well solution if given a sufficient budget. The GA would likely have done so as well. DIRECT and DE do not permit seeding the optimization with good points, which put them at a disadvantage.

For the hydraulic capture problem, Design Explorer was unable to build a surrogate model within the given function evaluation budget that captured the features of the objective function. This particular difficulty was due to the narrow region of decrease defined by the inactive-well threshold. However, surrogate model approaches are gaining popularity in this field, and should not be discarded as possibilities. See [43, 68] for example.

4.3.3 Robustness of Design

To evaluate the robustness of the design for the hydraulic capture CP, we evaluated the sensitivity of the results to the initial iterate and other factors. We evaluated the sensitivity of the solution achieved by APPS and IFFCO to the initial iterate. We excluded the GA from this analysis because the mixed-integer formulation achieved an optimal design with or without an initial iterate for this case. Similarly as described in § 4.2.3, for this application we generated a set of 65 initial iterates using the DIRECT algorithm and a cluster analysis. Neither of these methods were able to make significant improvement on this set of starting points for the hydraulic capture application and converged to local minima with all four wells operating at relatively low pumping rates.

Following reasoning similar to that used for the six-well water supply CP design, we generated two additional initial iterates. Table 8 gives the initial locations and well rates. Initial Iterate A had one injection well outside the plume, operating at $Q = 0.0064$ m/s^3, two extraction wells in the interior of the plume operating at $Q = -0.0064$ m/s^3, and one extraction well behind the plume where the head values are higher operating at $Q = -0.0032$ m/s^3. Initial Iterate B has a similar configuration as the one in Figure 3 (same pumping rates) but the wells are spaced further apart. APPS was able to generate a near-optimal one-well design for both of these cases, as was IFFCO with the caveat that the IFFCO result required two restarts of the algorithm for initial iterate A and a single restart of the algorithm for the initial iterate B. The IFFCO restarts led to the lowest cost designs but required many more function evaluations than APPS. For both APPS and IFFCO, the number of function evaluations was still roughly half the number needed by the mixed-integer GA for the baseline case.

Table 8
Additional initial iterates for hydraulic capture CP

Parameter	Iterate A	Iterate B
$x(1)$ (m)	250	400
$y(1)$ (m)	800	750
$Q(1)$ (m^3/s)	0.0064	0.0064
$x(2)$ (m)	250	650
$y(2)$ (m)	300	650
$Q(2)$ (m^3/s)	-0.0064	0.0064
$x(3)$ (m)	250	650
$y(3)$ (m)	650	250
$Q(3)$ (m^3/s)	-0.0064	-0.0064
$x(4)$ (m)	250	250
$y(4)$ (m)	450	200
$Q(4)$ (m^3/s)	-0.0064	-0.0064

To investigate the behavior of the GA in more detail, we examined the sensitivity of the solution to a random seed and found that the results were sensitive to this value. The GA failed to return the optimal design for nine out of 10 different random variables chosen. This sensitivity was not necessarily surprising given that the population size and number of generations (both 30) were at the low end of the range typically suggested for GA's. Increasing both the population size and the number of generations yielded near optimal designs for 10 different random seeds with no initial iterate. Of course, running the full number of generations with this population size required 10^4 function evaluations, even though one-well designs were obtained after 30 generations (3000 function evaluations) for each seed.

5 Conclusions

Deterministic sampling methods have not been widely used in the water resources community to compute optimal solutions. Several methods from this class were introduced and compared. Part of the motivation for this work is the fact that gradient based methods are not applicable to the problems presented here. For the five well problem (the easiest of the test suite here), we examined the performance of one gradient-based code, the FDNIPS solver from the OPT++ v2.0 [65] framework. This code is a nonlinear interior point code based on the work in [5, 6, 28]. The code uses finite difference gradients, either trust region or line search globalization, and a choice of three merit functions. We tried several combinations of the options. In every case the optimization failed after 1000 calls to the function or failed because the line search had reduced the step length 40 times without a sufficient decrease in the merit function.

However, we showed that this suite of derivative-free optimizers can be applied in an off-the-shelf manner using default parameters and, in most cases, obtain a significant decrease in cost. Moreover, the promising results presented here do not reflect the benefits that could arise from algorithm tuning. For example, numerical experiments showed that simple restarts with IFFCO made a significant difference on the hydraulic capture problem for a subset of the initial iterates.

An appropriate initial iterate was found to be an important part of the problem specification for certain methods. Guidance is provided to generate iterates that performed well for the experiments that were performed in this study. APPS, NOMAD, and IFFCO were found to provide good designs and efficient solutions when supplied with an appropriate initial iterate. The mixed-integer GA method was shown to be robust, but generally less computationally efficient than the best sampling methods. The methods that did not make use of feasible initial iterates did not succeed on the hydraulic capture problem. These findings indicate that a hybrid approach that combines the fast local, search of a stencil based method with the global search of a heuristic method is warranted. Such hybrid approaches are already appearing in the groundwater literature [70, 71].

The inclusion of the fixed costs posed a significant challenge. For example, the surrogate approach followed by DE was unable to model the inactive well-threshhold. However, both [69] and [64] show that realistic realistic cost functions lead to better solutions, especially when remediation horizons are short. The advantage of the mixed-integer GA over the methods relying on the inactive-well threshhold implies that alternate methods for determing the appropriate number of wells is warranted. Methods that choose the appropriate number of wells while simultaneously taking advantage of the fast convergence of the stencil based methods or the compuational efficiency of surrogate models are attractive. We should note that although NOMAD is capable of handling categorical variables, for comparison sake, that problem formulation was not considered. Methods for handling fixed costs directly and through the enhancement of the algorithms compared here is a topic of future work.

Algorithm maturation is expected for all methods and is already underway for some of the methods. These algorithmic changes can reasonably be expected to improve performance on this challenging set of test problems. The work presented herein provides a baseline for the efforts of others to develop and refine optimal design tools for the community problems and other water resources problems.

Finally, we have provided a set of solution approaches for an interpretation of the CPs. The inactive-well threshhold, the rounding of well locations to grid points, the choice of MODFLOW as the simulator, and the spatial discretization are all specific to this work. The CPs offer an opportunity to study multiple aspects of how problem formulation, implementation, simulation, optimzation modelling, and the optimizer effect the solution of underlying management problem. This work is a preliminary attempt at that larger effort.

5.1 Downloads

To facilitate the work of others, a web site was created that includes problem details and links to simulators and optimizers for the two CPs of concern in this work

`http://www4.ncsu.edu/~ctk/community.html`

While many other simulation and optimization approaches are possible, the links provided yield a simple starting point for some of the methods considered in this work.

Acknowledgements

The work at NCSU was partially supported by National Science Foundation grants DMS-0404537, DMS-0070641, DMS-0209695, DMS-0112542, Army Research Office grants DAAD19-02-1-0391, DAAD19-02-1-0111, and W911NF-06-1-0412 and a US Department of Education GAANN fellowship. The work at Clarkson University was partial supported by the NSF-AWM Mentor Travel Grant. The UNC efforts were funded by grant P2 ES05948 from the National Institute of Environmental Health Sciences. Sandia National Lab is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under Contract DE-AC04-94AL85000.

References

[1] A. J. Booker. Well–conditioned kriging models for optimization of computer models. Technical Report M&CT-TECH-002, Boeing Phantom Works, Mathematics and Computing Technology, 2000.

[2] D. P. Ahfeld and A. E. Mulligan. Advective control of groundwater contaminant plumes: Model development and comparison to hydraulic control. *Water Resources Research*, 35:2285–2294, 1999.

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

[4] D. P. Ahfeld, A. Zafirakou, and R. G. Riefler. Solution of the groundwater transport management problem with sequential relaxation. *Advances in Water Resources*, 21:591–604, 1998.

[5] M. Argaez and R.A. Tapia. On the global convergence of a modified augmented Lagrangian linesearch interior-point Newton method for nonlinear programming. *J. Optim. Theory Appl*, 114:1–25, 2001.

[6] M. Argaez, R.A. Tapia, and L. Velazquez. Numerical comparisons of path-following strategies for a primal-dual interior-point method for nonlinear programming. *J. Optim. Theory Appl*, 114:255–272, 2002.

[7] C. Audet and J. E. Dennis, Jr. Analysis of generalized pattern searches. *SIAM Journal on Optimization*, 13(3):889–903, 2003.

[8] C. Audet and J. E. Dennis, Jr. Mesh adaptive direct search algorithms for constrained optimization. *SIAM Journal on optimization*, 17(1):188–217, 2006.

[9] C. Audet, A. J. Booker, Jr. J. E. Dennis, P. D. Frank, and D. Moore. A surrogate-model-based method for constrained optimization, aiaa-2000-4891. In *Proceedings of the Symposium on Multidisciplinary Analysis and Optimization*, 2000.

[10] A. Battermann, J. M. Gablonsky, A. Patrick, C. T. Kelley, K. R. Kavanagh, T. Coffey, and C. T. Miller. Solution of a groundwater flow problem with implicit filtering. *Optimization and Engineering*, 3:189–199, 2002.

[11] M. Björkman and K. H.öm. Global optimization with the DIRECT algorithm in MATLAB. *Advanced Modeling and Optimization*, 2:17–37, 1999.

[12] A. J. Booker. DOE for computer output. Technical Report BCSTECH-94-052, Boeing Computer Services, Seattle, WA, 1994.

[13] A. J. Booker, J. E. Dennis, P. D. Frank, D. B. Serafini, V. Torczon, and M. W. Trosset. A rigorous framework for optimization of expensive function by surrogates. *Structural Optimization*, 17:1–13, 1999.

[14] C. Audet, A. J. Booker, J.E. Dennis, P. D. Frank, and D. W. Moore. A surrogate-model-based method for constrained optimization, aiaa-2000-4891. In *Eighth AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 2000.

[15] R. Carter, J. M. Gablonsky, A. Patrick, C. T. Kelley, and O. J. Eslinger.

Algorithms for noisy problems in gas transmission pipeline optimization. *Optimization and Engineering*, 2:139–157, 2001.

[16] T. D. Choi, O. J. Eslinger, P. Gilmore, A. Patrick, C. T. Kelley, and J. M. Gablonsky. IFFCO: Implicit Filtering for Constrained Optimization, Version 2. Technical Report CRSC-TR99-23, North Carolina State University, Center for Research in Scientific Computation, July 1999.

[17] G. Couture, C. Audet, J. E. Dennis, Jr., and M. A. Abramson. The NOMAD project. http://www.gerad.ca/NOMAD/, 2002.

[18] E.J. Cramer and J.M. Gablonsky. Effective parallel optimization of complex computer simulations. In *Proceedings of the 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, August, 2004.

[19] M. da Conceicao Cunha. On solving aquifer management problems with simulated annealing. *Water Resources Management*, 13:153–170, 1999.

[20] K. Deb. An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*, 186(2–4):311–338, 2000.

[21] K. Deb and R. B. Agrawal. Simulated binary crossover for continuous search space. *Complex Systems*, 9:115–148, 1995.

[22] K. Deb and H. G. Beyer. Self-adaptive genetic algorithms with simulated binary crossover. *Evolutionary Computation Journal*, 9(2):197–221, 2001.

[23] K. Deb and T. Goel. Controlled elitist non-dominated sorting genetic algorithms for better convergence. In E. Zitler, K. Deb, L. Thiele, C.A. Coello-Coello, and D. Corne, editors, *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization EMO 2001*, Lecture Notes on Computer Science, pages 67–81. , 2001.

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

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

[26] J. E. Dennis Jr. and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice–Hall, Inc., Englewood Cliffs, New Jersey, 1983.

[27] D. E. Dougherty and R. A. Marryott. Optimal groundwater management: 1. Simulated annealing. *Water Resources Research*, 27:2493–2508, 1991.

[28] A.S. El-Bakry, R.A. Tapia, T. Tsuchiya, and Y. Zhang. On the formulation and theory of the Newton interior-point method for nonlinear programming. *J. Optim. Theory Appl*, 89:507–541, 1996.

[29] M. Erickson, A. Mayer, and J. Horn. Multi-objective optimal design of groundwater remediation systems: Application of the niched pareto genetic algorithm. *Advances in Water Resources*, 25:51–65, 2002.

[30] D. E. Finkel. *Global Optimization with the DIRECT Algorithm*. PhD thesis, North Carolina State University, Raleigh, North Carolina, 2005.

[31] D. E. Finkel and C. T. Kelley. Convergence analysis of the DIRECT

algorithm. Technical Report CRSC-TR04-28, North Carolina State University, Center for Research in Scientific Computation, July 2004.

[32] K. R. Fowler, C. T. Kelley, C. E. Kees, and C. T. Miller. A hydraulic capture application for optimal remediation design. In C. T. Miller, M. W. Farthing, W. G. Gray, and G. F. Pinder, editors, *Proceedings of the XV International Conference on Computational Methods in Water Resources*, pages 1149–1156. , 2004. June 2004, Chapel Hill, NC.

[33] K. R. Fowler, C. T. Kelley, C. T. Miller, C. E. Kees, Robert W. Darwin, J. P. Reese, M. W. Farthing, and Mark S. C. Reed. Solution of a well-field design problem with implicit filtering. *Optimization and Engineering*, 5: 207–234, 2004.

[34] J. M. Gablonsky. DIRECT Version 2.0 User Guide. Technical Report CRSC-TR01-08, Center for Research in Scientific Computation, North Carolina State University, April 2001.

[35] J. M. Gablonsky. *Modifications of the DIRECT Algorithm*. PhD thesis, North Carolina State University, Raleigh, North Carolina, 2001.

[36] J. M. Gablonsky and C. T. Kelley. A locally-biased form of the DIRECT algorithm. *Journal of Global Optimization*, 21:27–37, 2001.

[37] P. Gilmore and C. T. Kelley. An implicit filtering algorithm for optimization of functions with many local minima. *SIAM J. Optim.*, 5:269–285, 1995.

[38] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison Wesley Pub. Company, 1989.

[39] G. A. Gray and K. R. Fowler. Approaching the groundwater remediation problem using multifidelity optimization. In *Proc. of the CMWR XVI - Computational Methods in Water Resources*, 19-22 June 2006.

[40] G. A. Gray and T. G. Kolda. Algorithm 856: APPSPACK 4.0: Asynchronous parallel pattern search for derivative-free optimization. *ACM Transactions on Mathematical Software*, 32(3):485–507, 2006.

[41] J. D. Griffin and T. G. Kolda. Nonlinearly-constrained optimization using asynchronous parallel generating set search. Technical Report SAND2007-3257, Sandia National Laboratories, Albuquerque, NM and Livermore, CA, 2007.

[42] J. D. Griffin, T. G. Kolda, and R. M. Lewis. Asynchronous parallel generating set search for linearly-constrained optimization. Technical Report SAND2006-4621, Sandia National Laboratories, Albuquerque, NM and Livermore, CA, August 2006.

[43] T. Hemker, K. R. Fowler, and 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*, 19-22 June 2006.

[44] T. Hemker, K. R. Fowler, M. W. Farthing, and O. von Stryk. A mixed-integer simulation-based optimization approach with surrogate functions in water resources management. *Submitted to Optimization and Engineering*, 2007.

[45] J. H. Holland. *Adaption in Natural and Artificial Systems*. Univ. of Mich. Press, Ann Arbor, Mich., 1975.

[46] J. H. Holland. Genetic algorithms and the optimal allocation of trials. *SIAM J. Comput.*, 2, 1973.

[47] P. D. Hough, T. G. Kolda, and V. J. Torczon. Asynchronous parallel pattern search for nonlinear optimization. *SIAM J. Sci. Comp.*, 23:134–156, 2001.

[48] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the lipschitz constant. *Journal of Optimization Theory and Application*, 79(1):157–181, October 1993.

[49] L. Kaufman and P. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley & Sons, New York, 1990.

[50] C. T. Kelley. *Iterative Methods for Optimization*. Number 18 in Frontiers in Applied Mathematics. SIAM, Philadelphia, 1999.

[51] S. Kirkpatrick, C. D. Geddat, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[52] T. G. Kolda. Revisiting asynchronous parallel pattern search for nonlinear optimization. *SIAM J. Optimiz.*, 16(2):563–586, December 2005.

[53] T. G. Kolda and V. Torczon. On the convergence of asynchronous parallel pattern search. *SIAM J. Optimiz.*, 14(4):939–964, 2004.

[54] T. G. Kolda and V. J. Torczon. Understanding asynchronous parallel pattern search. In G. DiPillo and A. Murli, editors, *High Performance Algorithms and Software for Nonlinear Optimization*, pages 316–335. Kluwer Academic Publishers B.V., 2003.

[55] T. G. Kolda, R. M. Lewis, and V. Torczon. Stationarity results for generating set search for linearly constrained optimization. *SIAM Journal on Optimization*, 17(4):943–968, 2006.

[56] R. M. Lewis and V. Torczon. Rank ordering and positive bases in pattern search algorithms. Technical Report 96-71, Institute for Computer Applications in Science and Engineering, December 1996.

[57] S. L. Mattot, A. J. Rabideau, and J. R. Craig. Optimization of pump and treat systems using analytic element flow models. *Advances in Water Resources*, 29:760–775, 2006.

[58] A. S. Mayer, C. T. Kelley, and C. T. Miller. Optimal design for problems involving flow and transport phenomena in saturated subsurface systems. *Advances in Water Resources*, 12:1233–1256, 2002.

[59] A. S. Mayer, C. T. Kelley, and C. T. Miller. Electronic supplement to "Optimal design for problems involving flow and transport phenomena in saturated subsurface systems", 2003. "http://www.elsevier.com/gej-ng/10/8/34/58/59/41/63/show/index.htt", 17 pages.

[60] A.S. Mayer, C.T. Kelley, and C.T. Miller. Optimal design for problems involving flow and transport phenonmena in saturated subsurface systems. *Advances in Water Resources*, 12:1233–1256, 2002.

[61] M. G. McDonald and A. W. Harbaugh. A modular three dimensional finite difference groundwater flow model. *U.S. Geological Survey Tech-*

niques of Water Resources Investigations, 1988.

[62] M. G. McDonald and A. W. Harbaugh. Programmer's documentation for MODFLOW-96, an update to the U.S. geological survey modular finite difference groundwater flow model. *U.S. Geological Survey Open-File Report 96-486*, 1996.

[63] D. C. McKinney and M. D. Lin. Genetic algorithm solution of groundwater management models. *Water Resources managment*, 30:1897–1906, 1994.

[64] D. C. McKinney and M. D. Lin. Approximate mixed integer nonlinear programming methods for optimal aquifer remdiation design. *Water Resources Research*, 31:731–740, 1995.

[65] J.C. Meza. OPT++ an object-oriented class library for nonlinear optimization. Technical Report SAND94-8225, Sandia National Laboratories, Albuquerque, NM and Livermore, CA, 1994.

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

[67] P. Reed, B. Minsker, and D.E. Goldberg. Designing a competent simple genetic algorithm for search and optimization. *Water Resources Research*, 36(12):3757–3761, 2000.

[68] R. G. Regis and C. A. Shoemaker. Constrained Global Optimization of Expensive Black Box Functions Using Radial Basis Functions. *Journal of Global Optimization*, 31:153 – 171, 2005.

[69] X. Ren and B. Minsker. Which groundwater remediation objective is better, a realistic or a simple one? In *Proceedings of the World Water and Environmental Resources Congress 2003 and Related Symposia*, 2003.

[70] M. Sayeed and G.K. Mahinthakumar. Efficient parallel implementation of hybrid optimzation approaches for solving groundwater inverse problems. *Journal of Computing in Civil Engineering-ASCE*, 19(4):329–340, 2005.

[71] H. Shieh and R.C. Peralta. Optimal in situ bioremediation design by hybrid genetic algorithm-simulated annealing. *Journal of Water Resources Planning and Management-ASCE*, 131(1):67–78, 2005.

[72] D. E. Stoneking, G. L. Bilbro, R. J. Trew, P. Gilmore, and C. T. Kelley. Yield optimization using a GaAs process simulator coupled to a physical device model. *IEEE Transactions on Microwave Theory and Techniques*, 40:1353–1363, 1992.

[73] C. Zheng and P. Wang. *MT3DMS: A Modular Three-Dimensional Multispecies Transport Model for Simulation of Advection, Dispersion, and Chemical Reactions of Contaminants in Groundwater Systems; Documentation and User's Guide*, December 1999.

[74] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation Journal*, 8 (2):173–195, 2000.

A Software

Implicit filtering codes are available in both FORTRAN and MATLAB from
http://www4.ncsu.edu/~ctk/iffco.html.
We used the FORTRAN version in this work. The MATLAB code is under development and when finished will replace the FORTRAN code.

The version of NOMAD used in these tests is C++ optimization software based on the generalized pattern search (GPS) algorithm. Both C++ and MATLAB implementations are available from
http://www.gerad.ca/NOMAD/

Design Explorer is a suite of experimental design, modeling and optimization tools for use with computer simulations. A typical DE modeling session involves defining the problem, the design objectives, and identifying the input and output parameters of the system. Design Explorer is set up to run the simulation automatically. For information on obtaining Design Explorer contact Howard Lohr at
howard.c.lohr@boeing.com.

The FORTRAN implementation of DIRECT that was used to obtain the results presented in this paper can be obtained from
http://www4.ncsu.edu/ ctk/iffco.html.
In addition, a MATLAB implementation can be found at
http://www4.ncsu.edu/ ctk/Finkel_Direct/
and an alternative MATLAB implementation is part of the TOMLAB package [11].

NSGA-II is implemented in C and is available for downloading from
http://www.iitk.ac.in/kangal.
The user is required to implement problem-specific routines for evaluating the objective function and constraints.

APPSPACK is a C++ implementation of APPS that can be used in serial mode or in parallel with MPI. It can be obtained from
http://software.sandia.gov/appspack.
The target platforms for APPSPACK are the loosely-coupled parallel systems now widely available. To find a solution to these problems, we used version 4.0 [40].