CONSTRAINED PRODUCTION OPTIMIZATION WITH AN EMPHASIS ON DERIVATIVE-FREE METHODS

A THESIS SUBMITTED TO THE DEPARTMENT OF
ENERGY RESOURCES ENGINEERING
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF SCIENCE

By
Obiajulu Joseph Isebor
June 2009

Abstract

Production optimization involves the determination of optimum well controls to maximize an objective function such as cumulative oil production or net present value. In practice, this problem additionally requires the satisfaction of physical and economic constraints. Thus the overall problem represents a challenging nonlinearly constrained optimization. This work entails a comparative study of several popular optimization methods applied to the solution of the fully constrained production optimization problem. The methods considered include gradient-based methods, a genetic algorithm, general pattern search and Hooke-Jeeves direct search.

In the application of these methods to bound-constrained problems, it is shown that derivative-free methods tend to be about an order of magnitude slower than gradient-based methods that compute gradients using an adjoint procedure. The efficiency of the derivative-free methods is significantly improved through the use of surrogate-based optimization and distributed computing. A hybrid implementation combining the most desirable parts of some of the different methods is presented and shown to perform better than the individual methods.

In order to address the solution of the fully constrained production optimization problem, different constraint handling techniques are investigated, including the sequential quadratic programming approach, penalty function approach, filter method, and a hybrid methodology. The results of the application of these methods indicate that the gradient-based sequential quadratic programming, general pattern search with filter and a hybrid method combining the genetic algorithm with a robust penalty function treatment and an efficient local search method are the most promising.

Acknowledgments

First and foremost, I would like to thank God with whom all things are possible. It is by His grace that we are able to thrive and succeed and all glory and honor go to Him.

I would like to express my most sincere gratitude to my advisor, Prof. Lou Durlofsky. In addition to providing insightful comments and direction on my work, he has become a role model to me in terms of having a great work ethic and displaying professionalism in academics and research. I feel very privileged to be one of his students, and I eagerly look forward to continuing to work with him in the near future. Special thanks go to Dr. David Echeverría and Jerome Onwunalu with whom I have had very productive discussions regarding the use of optimization algorithms and who have provided invaluable assistance with the codes used in this work. I also look forward to maintaining a working relationship with both of them in the years to come.

Many thanks to my friends and family for their support throughout my academic career. I would specially like to thank my mother, Mrs. Esther Isebor, for her constant encouragement without which I certainly would not be where I am today. I consider myself lucky to be surrounded by great friends and colleagues who have supported me in ways that they might not even know. Of special mention are Richard Mills Osuamkpe, Garfield Williams, Shola Olorunnipa, Michael Krause, Ethan Chabora, Matthieu Rousset and Amar Alsheri, among many others. It has really been a blessing having all these people consider me as a friend, and I would like to acknowledge them all as having played a major role in my studies and my life and helping me grow into the person I am today.

I am indebted to all the students and staff (teaching, research and administrative) of the Department of Energy Resources Engineering for providing the most conducive and challenging learning environment I have ever been a part of. Also, I am very grateful to the industrial affiliates of the SUPRI-B (reservoir simulation), SUPRI-HW (advanced wells) and Smart Fields Research Consortia at Stanford University for their financial support during my studies.

Contents

Abstract **v**

Acknowledgments **vii**

Table of Contents **ix**

List of Tables **xi**

List of Figures **xiii**

1 Introduction **1**

1.1 Problem Statement and Objectives 2

1.2 Literature Survey 3

1.2.1 Gradient-based Optimization Methods 3

1.2.2 Gradient-Free Optimization Methods 5

1.3 Thesis Outline 6

2 Optimization Methods Considered **9**

2.1 Gradient-based Methods 9

2.1.1 Gradient Computation from Adjoint Model 10

2.1.2 Gradient Computation from Finite Differences 12

2.2 Derivative-Free Methods 13

2.2.1 Genetic Algorithms 13

2.2.2 General Pattern Search 16

2.2.3 Hooke-Jeeves Direct Search 19

2.3 Hybrid Methodology 21

2.4 Computational Enhancement Strategies 22
2.4.1 Surrogate-based Optimization 22
2.4.2 Distributed Computing 25

3 Optimization with Nonlinear Constraints 27
3.1 Sequential Quadratic Programming 27
3.2 Penalty Functions 29
3.2.1 Basic Description 29
3.2.2 Parameterless Penalty Function for GAs 31
3.3 The Filter Method 33
3.4 Hybrid Methodologies 35

4 Example Applications 37
4.1 Optimization Cases With Bound Constraints 38
4.1.1 Case 1: Section of Stanford VI Reservoir Model 38
4.1.2 Case 2: Two-dimensional Heterogeneous Model 44
4.2 Optimization Cases Including Nonlinear Constraints 52
4.2.1 Case 3: Two-dimensional Heterogeneous Model 52
4.2.2 Case 4: Section of SPE 10 Model 60

5 Conclusions and Future Work 67
5.1 Conclusions 67
5.2 Future Work 68

Nomenclature 69

Bibliography 71

List of Tables

4.1 Case 1 simulation and optimization parameters 39
4.2 Summary of performance of different optimizers for Case 1 40
4.3 Case 2 simulation and optimization parameters 45
4.4 Summary of performance of different optimizers for Case 2 46
4.5 Summary of performance of GA with ANN surrogate (Case 2) 48
4.6 Summary of performance of different optimizers for Case 2, with distributed computing 51
4.7 Case 3 simulation and optimization parameters 52
4.8 Case 3 results summary 58
4.9 Case 4 simulation and optimization parameters 61

List of Figures

2.1 Illustration of gradient-based optimization procedure 10

2.2 Illustration of main GA operators 14

2.3 Workflow of the GA optimization process 16

2.4 Examples of positive spanning frames for $\mathbb{R}^n$ with $n + 1$ and $2n$ poll points (from [28]) 17

2.5 Illustration of a sequence of polls in $\mathbb{R}^n$ with $2n$ poll frame (from [28]) 18

2.6 Illustration of exploratory and pattern moves in HJDS 19

2.7 Illustration of the workflow for training an ANN (from [18]) 23

2.8 Kriging response surface for 2D problem with data points shown in black (from [33]) 24

3.1 Illustration of how parameterless penalty function modifies original problem (from [17]) 32

3.2 Illustration of a filter at iteration k (from [6]) 34

4.1 Section of Stanford VI reservoir model, showing permeability field and wells 38

4.2 Relative permeability curves for the oil and water phases (Case 1) . . 39

4.3 Comparison of the performance of gradient-based and derivative-free algorithms (Case 1) 40

4.4 Comparison of the resulting BHP profiles for two injectors and two producers from SQP+adjoint and HJDS optimization algorithms . . . 42

4.5 Comparison of Case 1 cumulative production and injection profiles for the base case, SQP+adjoint and HJDS solutions 42

4.6 Comparison of the five SQP+adjoint runs using initial guesses from the final population of the GA 43
4.7 Comparison of the five hybrid solutions for one of the production wells 44
4.8 Two-dimensional reservoir model (x-permeability field shown) with four production (red) and four injection (blue) wells arranged in a line drive pattern 45
4.9 Comparison of the performance of the gradient-based and derivative-free algorithms (Case 2) 46
4.10 Comparison of the resulting BHP profiles for two injectors and two producers for all optimization methods considered 47
4.11 Comparison of Case 2 cumulative production and injection profiles of the base case and SQP+adjoint solution 48
4.12 Results using GA with ANN surrogate (Case 2) 49
4.13 Comparison of GPS with DACE kriging surrogate with other algorithms (Case 2) 50
4.14 Comparison of the Case 2 results with distributed computing 51
4.15 Two-dimensional reservoir model with two production (red) and two injection (blue) wells (x-direction permeability displayed) 53
4.16 SQP+adjoint results excluding nonlinear constraint enforcement, showing constraint violation 54
4.17 Results of tuning penalty parameter (Case 3) 54
4.18 Comparison of results for GA with parameterless and traditional penalty functions (Case 3) 56
4.19 Comparison of results for GPS with filter and GPS with traditional penalty function (Case 3) 56
4.20 Comparison of results for parameterless penalty GA hybridized with HJDS with traditional penalty function and HJDS with traditional penalty function alone 57
4.21 SQP+FD results showing constraint satisfaction (Case 3) 58
4.22 Comparison of Case 3 cumulative production and injection profiles of the base case, bound-constrained solution for SQP+adjoint, and the best nonlinearly constrained solution from GPS+tuned penalty. . . . 59

4.23 Section of SPE 10 reservoir model, showing porosity of top layer. Model contains 16 injection (blue) and 9 production (red) wells 60
4.24 Relative permeability curves for the oil and water phases (Case 4) . . 61
4.25 SQP+adjoint result for total liquid production rate, excluding nonlinear constraints (Case 4) 62
4.26 Plot of feasible fraction of GA showing that the first phase of hybrid procedure successfully identifies the feasible region (Case 4) 63
4.27 Nonlinearly constrained hybrid results compared with SQP+adjoint bound-constrained results (Case 4) 64
4.28 Plot of total liquid production rate of hybrid solution showing constraint satisfaction (Case 4) 64
4.29 Comparison of Case 4 cumulative production and injection profiles of the base case, bound-constrained solution for SQP+adjoint, and the hybrid nonlinearly constrained solution 65

Chapter 1

Introduction

As a result of economic and population growth, the world total energy demand in 2030 is expected to be approximately 35% higher than it was in 2005, despite gains in energy efficiency [19]. Oil and natural gas together provided almost 60% of global energy in 2005, and forecasts indicate that they will continue to be significant contributors to global energy for decades to come. However, despite increasing demand, oil production in many fields around the world is reaching a plateau and the number of significant discoveries is decreasing every year. As a result, in the oil and gas industry, significant effort is being made to develop effective reservoir management technologies to optimize oil and gas production. This has led to a great deal of interest in the idea of efficient closed-loop reservoir management [24, 25], of which production optimization is a crucial part.

In the context of efficient reservoir management, production optimization implies maximizing or minimizing a particular objective function, such as cumulative oil or water production or net present value (NPV) over a specified period of time by finding the optimal set of control variables, such as well rates or bottom-hole pressures (BHPs). Since the relationship between the reservoir dynamics and the control variables is in general nonlinear, finding the optimal set of controls is a very challenging task. In addition, the problem usually needs to be solved subject to operational constraints, such as maximum and minimum BHPs, maximum field water injection rates, surface facility handling capacities, etc. Thus, the problem is most properly viewed as a constrained optimization problem.

1.1 Problem Statement and Objectives

The production optimization problem can be formally stated as:

$$\begin{aligned} &\max_{\mathbf{u}\in\mathbb{R}^n} \; J(\mathbf{u}), \\ &\text{subject to:} \\ &c_i(\mathbf{u}) \leq 0 \qquad i = 1, \cdots, m, \\ &\mathbf{l}_B \leq A\mathbf{u} \leq \mathbf{u}_B, \end{aligned} \tag{1.1}$$

where $J(\mathbf{u})$ is the objective function (e.g., NPV or cumulative oil produced) to be optimized, $\mathbf{u}$ is the vector of control variables (e.g., well BHPs), the c_i's represent the m nonlinear inequality constraints in the problem (e.g., maximum allowable water cut in any well) and $\mathbf{l}_B \leq A\mathbf{u} \leq \mathbf{u}_B$ refers to bound and linear constraints (such as maximum and minimum well BHPs). The objective function is usually computed using the output from a reservoir simulator, and this makes function evaluations expensive.

This production optimization problem can be addressed using various gradient-based optimization approaches. The derivative of the objective function with respect to the controls, which is required for these gradient-based algorithms, can be computed using numerical finite-differences, though this requires a high number of function evaluations to compute all of the components of the gradient. An efficient adjoint-based technique for computing the required gradients greatly reduces the computational effort [9, 24, 25, 43, 45, 46]. However, adjoint-based techniques require extraction of information from the reservoir simulator during the course of the computations, and therefore are only feasible with full access to and detailed knowledge of the simulator source code. Even when such access exists, the effort associated with the development of the adjoint code is significant. Finally, it is important to note that gradient-based production optimization techniques, though computationally efficient, converge to local rather than global optima.

The goals of this study are to implement and assess several optimization methods, with emphasis on procedures that do not require gradient information. These techniques are referred to as derivative-free optimization algorithms [15]. As will be shown in later chapters, these methods are straightforward to implement and most of them parallelize naturally because the function evaluations do not have to be computed sequentially.

Consistent with our expectations, the derivative-free methods assessed in this study typically require about an order of magnitude more function evaluations than adjoint-based optimization techniques. Therefore, computational cost reduction strategies, such as surrogate-based optimization and distributed computing, which act to improve the computational efficiency of the derivative-free methods, are also evaluated. In addition, hybrid methodologies, which combine the positive features of different algorithms, are investigated.

Many existing derivative-free implementations can readily handle problems with only bound and linear constraints on the control variables. As stated earlier, however, the production optimization problem is a generally constrained problem, including nonlinear inequality constraints. Therefore, we investigate various techniques for handling these nonlinear constraints with the overall goal of implementing a set of methods appropriate for fully constrained production optimization problems.

1.2 Literature Survey

A wide variety of optimization algorithms have been used in industrial applications, including production optimization. The optimization methods that have been applied to solve the production optimization problem, as reported in the literature, can be broadly classified as gradient-based and gradient-free methods.

1.2.1 Gradient-based Optimization Methods

The gradient-based optimization methods applied to the production optimization problem include the steepest ascent/descent method, conjugate gradient method, and sequential quadratic programming (SQP) [40]. These gradient-based optimization methods are known to be efficient and show fast convergence when the gradient of the objective function with respect to the controls is available. Different methods for computing the gradient have been used and they can be classified into three categories: numerical gradients computed using finite-difference (FD) techniques [2, 31, 51, 53], ensemble-based gradients computed using the ensemble Kalman filter (EnKF) [14], and adjoint-based techniques from optimal control theory [9, 10, 24, 43, 45, 46]. The advantage of using the numerical and ensemble-based gradients is that the simulator can be treated as a "black box" and the gradients computed from function evaluations. The adjoint method is far more efficient, but requires access to and detailed knowledge of the reservoir simulator source code, as discussed above.

The available literature on the use of gradient-based optimization methods applied to problems in the oil and gas industry include the following. Aitokhuehi [2] used the conjugate gradient and Levenberg-Marquardt algorithms with numerical gradients for the real-time optimization of smart wells with model updating. Carroll [12] demonstrated the use of Newton's method, also with numerical gradients, for the optimization of production systems. Kumar [29] applied the conjugate gradient method, with numerically computed gradients, to optimize well settings so as to maximize residually trapped CO_2 in geologic carbon sequestration. Wang et al. [50, 51] used SQP with numerical gradients to solve the production optimization problem, and Litvak et al. [31] applied the method to the Prudhoe Bay E-field. Yeten et al. [53] also used SQP for the optimization of smart well controls. Wang et al. [49] compared the steepest ascent method with FD gradients, simultaneous perturbation stochastic approximation (SPSA) gradients, and ensemble-based gradients for a production optimization problem. They showed that the steepest ascent method with FD gradients was the most efficient of the three methods, if a small perturbation size was used. Chaudhri et al. [14] demonstrated the use of an improved ensemble-based gradient computation approach for production optimization using the conjugate gradient method. The use of numerical gradients decreases the efficiency of these gradient-based optimization methods because the number of function evaluations (simulations) typically scales with the dimension of the parameter space, i.e., with the number of optimization parameters.

The use of adjoint procedures, which derive from optimal control theory, alleviates this problem by efficiently providing the gradient of the objective function with respect to the controls. These gradients are obtained by solving an adjoint model using Jacobian and other information extracted during the forward simulation. Sarma et al. [45, 46] and Jansen and Brouwer [9, 10, 25] have demonstrated the effectiveness of adjoint procedures for production optimization and closed-loop reservoir management. Earlier investigators who applied optimal control theory for enhanced oil recovery include Ramirez and co-workers [32, 38, 43], Asheim [4] and Virnovsky [48].

The preceding authors did not include consideration of nonlinear constraints. Sarma et al. [45, 47] described a feasible direction algorithm for handling nonlinear path constraints. The nonlinear inequality constraints in production optimization are path constraints because they need to be satisfied at every time step during the reservoir simulation. Sarma distinguished between two types of algorithms for dealing with nonlinear path constraints: algorithms that solve the path constraints implicitly together with the dynamic system (presupposes full access to the forward model or simulator), and algorithms that calculate the path constraints explicitly after the dynamic system has been solved. The algorithm in [45] and [47] is a constraint handling method of the first type. It entails a feasible direction algorithm that uses lumped constraints and a feasible line search. Some iteration to achieve feasible controls at each time step of the forward simulation is also required. Sarma et al. presented promising results for certain types of nonlinearly constrained problems, but the method cannot yet handle all types of relevant nonlinear constraints. The nonlinear constraint handling techniques presented in this work are of the second type, which treat the simulator as a black box. This enables us to handle any type of nonlinear constraint, though the computational demands will significantly exceed those of Sarma et al.

1.2.2 Gradient-Free Optimization Methods

The gradient-free methods do not require the explicit calculation of objective function gradients and use just the objective function values obtained from function evaluations (reservoir simulation plus economic model evaluation). These gradient-free methods can be further classified into deterministic (e.g., polytope or Nelder-Mead simplex method [30, 39]) and stochastic methods, including genetic algorithms (GAs) and tabu search. Carroll [12] used the polytope method for production systems optimization and showed that it is an effective alternative to gradient-based methods. Of the stochastic algorithms, GAs appear to be the most commonly used and have been applied for the solution of the well placement problem, where the objective is to optimize well type, location and trajectory [21, 41, 44]. Cullick et al. [16] used a global optimizer based on tabu search to optimize production strategies while accounting for risk. Harding et al. [22] used a GA with problem-specific crossover operators to select optimal well rates for the maximization of NPV. Almeida et al. [3] used the GA for production optimization of smart wells, where they also accounted for technical uncertainties, such as the risk of valve failure. It should be noted that none of this work addressed nonlinear constraint handling.

There are many other derivative-free methods that have been applied in other contexts. Of particular note are the so-called direct search methods, which are stencil or frame-based optimization techniques. Kolda et al. [28] and Conn et al. [15] presented excellent reviews of these methods, including their history and convergence theory. Direct search algorithms have been in existence for a long time. Hooke and Jeeves [23] introduced their direct search method in 1961. Audet and Dennis [5] introduced a more general version of pattern search methods and Marsden et al. applied these methods to optimization problems in aeroacoustics [35] and medicine [34].

1.3 Thesis Outline

This project entails a comparative study of optimization methods, with an emphasis on derivative-free methods, to solve generally constrained production optimization problems. In Chapter 2, brief descriptions of the different optimization methods considered in this study are presented. First, a gradient-based method is presented, with gradients obtained through numerical finite difference and using adjoint models. Next, the derivative-free methods are discussed. These include the GA and two direct search methods, general pattern search (GPS) and Hooke-Jeeves direct search (HJDS), which can be seen as a subset of GPS methods. The advantages and disadvantages of these methods are discussed, and a hybrid methodology that takes advantage of the positive features of different methods is presented. Strategies to reduce the computational cost of the derivative-free methods, including surrogate-based optimization and distributed computing, are also discussed. The basic optimization methods presented in Chapter 2 can only handle bound and linearly constrained problems.

Chapter 3 presents techniques for nonlinear constraint handling. Such procedures enable the methods presented in Chapter 2 to handle the fully constrained production optimization problem, with bound, linear and nonlinear inequality constraints. The use of sequential quadratic programming (SQP) to handle nonlinear constraints in gradient-based optimization is described. This method is a very popular nonlinear programming method because of its efficiency in searching for a constrained optimal solution. The traditional penalty function methods are presented and it is noted that these methods can be used with any base optimization algorithm because they only modify the objective function. However, they may display a lack of robustness, as shown by some of the results in Chapter 4. A variant of the traditional penalty function for use with the GA is presented and is shown to improve the robustness of the traditional method. The filter method for nonlinear constraint handling is presented next and an implementation for use with the GPS algorithm is described. A hybrid methodology, which combines the GA with robust penalty function and an efficient local method, is also presented.

Chapter 4 presents results using the methods described in Chapters 2 and 3. First the base methods presented in Chapter 2 are used to solve two production optimization cases (subject only to bound constraints) with heterogeneous reservoir models. The results highlight the characteristics of the different methods. From the results, the efficiency of the adjoint-based method is apparent. The significant improvement achieved when the derivative-free methods are used with surrogates, and the speedup obtained using distributed computing, are demonstrated. Next, nonlinear constraints are introduced in the production optimization problem and the different constraint handling techniques are evaluated using two different heterogeneous reservoir models. From the results of these cases, it is seen that the SQP constraint handling, the filter method for GPS, and a hybrid of the robust penalty function GA and a local optimization method with traditional penalty, are the most promising methods for the solution of the fully constrained production optimization problem.

The thesis concludes in Chapter 5, where the methods and findings are summarized and recommendations for future work are provided.

Chapter 2

Optimization Methods Considered

The derivative-free methods considered in this study are the genetic algorithm (GA), general pattern search (GPS) and Hooke-Jeeves direct search (HJDS). In this chapter, brief descriptions of these methods, together with the gradient-based method used for comparison, are provided. A hybrid methodology involving the robust exploratory GA and the more efficient local gradient-based procedure is described. Strategies for improving the computational efficiency of the derivative-free methods are also discussed.

2.1 Gradient-based Methods

Optimization methods are basically prescribed procedures for searching the solution space with the goal of finding the optimum point. Gradient-based optimization methods utilize the gradient of the objective function with respect to the control variables to guide the search. A good example is the steepest ascent/descent (maximize/minimize) method which searches using

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \alpha_k \left(\nabla J\right)^k, \tag{2.1}$$

where α_k is the step size for the kth iteration (found using a line search [40]) and $\left(\nabla J\right)^k$ is the gradient of the objective function with respect to the control vector $\mathbf{u}$ evaluated at iteration k. A typical gradient based optimization process is illustrated in Figure 2.1 for a problem with two control variables. Here $\mathbf{u}^0$ designates the initial guess, $\mathbf{u}^*$ is the optimum point, and the closed curves are contours of the objective function. Equation 2.1 and Figure 2.1 show that an iteration in a gradient-based optimization method usually entails the computation of a gradient, the use of the gradient to determine a search direction, and a line search in this direction to improve the objective function.

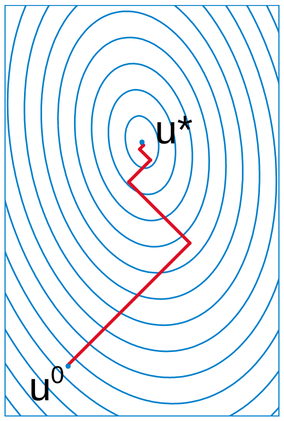


Figure 2.1: Illustration of gradient-based optimization procedure

The actual gradient-based optimization algorithm used in this study is the sequential quadratic programming approach (SQP) [40]. This procedure will be discussed in more detail in Chapter 3 where we present gradient-based optimization for the fully constrained problem. The objective function gradients used in gradient-based methods can be obtained using either an adjoint procedure or numerical finite differences. We now briefly describe these two approaches.

2.1.1 Gradient Computation from Adjoint Model

This description of the adjoint model for gradient computation follows that of Sarma et al. [46]. In the adjoint formulation, the objective function to be optimized is given by:

$$J(\mathbf{u}) = \sum_{n=0}^{N-1} L^n \left(\mathbf{x}^{n+1}, \mathbf{u}^n\right), \tag{2.2}$$

where $\mathbf{x}$ refers to the dynamic states of the system, L is a kernel known as the Lagrangian, n designates time step and N is the total number of time steps. In addition to the linear and nonlinear constraints presented in Equation 1.1, this objective function is optimized subject to two additional sets of constraints given by:

$$\begin{aligned} \mathbf{g}^n\left(\mathbf{x}^{n+1}, \mathbf{x}^n, \mathbf{u}^n\right) = 0 \quad \forall n \in (0, \cdots, N-1), \\ \mathbf{x}^0 = \mathbf{x}_0 \quad \text{(Initial Condition)}, \end{aligned} \tag{2.3}$$

where the set of equations $\mathbf{g}^n$, together with the initial condition, define the dynamic system (reservoir simulation equations for each grid block at each time step). In order to obtain the adjoint model, an augmented objective function, J_A, consisting of a combination of the original objective function and the constraints in Equation 2.3 is constructed. The form of this augmented objective function is:

$$J_A = \sum_{n=0}^{N-1} L^n\left(\mathbf{x}^{n+1}, \mathbf{u}^n\right) + \boldsymbol{\lambda}^{T0}\left(\mathbf{x}^0 - \mathbf{x}_0\right) + \sum_{n=0}^{N-1} \boldsymbol{\lambda}^{T(n+1)} \mathbf{g}^n\left(\mathbf{x}^{n+1}, \mathbf{x}^n, \mathbf{u}^n\right), \tag{2.4}$$

where the vectors $\boldsymbol{\lambda}^n$ are known as Lagrange multipliers and superscript T designates transpose. For optimality of the original problem, as well as the augmented problem, the first variation of J_A must equal zero. The first variation of J_A is given by:

$$\begin{aligned} \delta J_A &= \left[\frac{\partial L^{N-1}}{\partial \mathbf{x}^N} + \boldsymbol{\lambda}^{TN} \frac{\partial \mathbf{g}^{N-1}}{\partial \mathbf{x}^N}\right] \delta \mathbf{x}^N + \sum_{n=1}^{N-1} \left[\frac{\partial L^{n-1}}{\partial \mathbf{x}^n} + \boldsymbol{\lambda}^{T(n+1)} \frac{\partial \mathbf{g}^n}{\partial \mathbf{x}^n} + \boldsymbol{\lambda}^{Tn} \frac{\partial \mathbf{g}^{n-1}}{\partial \mathbf{x}^n}\right] \delta \mathbf{x}^n \\ &+ \sum_{n=0}^{N-1} \left[\frac{\partial L^n}{\partial \mathbf{u}^n} + \boldsymbol{\lambda}^{T(n+1)} \frac{\partial \mathbf{g}^n}{\partial \mathbf{u}^n}\right] \delta \mathbf{u}^n. \end{aligned} \tag{2.5}$$

The terms involving $\delta \mathbf{x}^n$ can be set to zero by choosing $\boldsymbol{\lambda}^n$ such that:

$$\begin{aligned} \boldsymbol{\lambda}^{Tn} &= -\left[\frac{\partial L^{n-1}}{\partial \mathbf{x}^n} + \boldsymbol{\lambda}^{T(n+1)} \frac{\partial \mathbf{g}^n}{\partial \mathbf{x}^n}\right] \left[\frac{\partial \mathbf{g}^{n-1}}{\partial \mathbf{x}^n}\right]^{-1} \quad \forall n = 1, \ldots, N-1 \\ \boldsymbol{\lambda}^{TN} &= -\left[\frac{\partial L^{N-1}}{\partial \mathbf{x}^N}\right] \left[\frac{\partial \mathbf{g}^{N-1}}{\partial \mathbf{x}^N}\right]^{-1} \quad \text{(Final Condition)}. \end{aligned} \tag{2.6}$$

This equation constitutes the adjoint model. Note that $\boldsymbol{\lambda}^n$ depends on $\boldsymbol{\lambda}^{n+1}$, therefore the adjoint problem is solved backwards in time. With the $\boldsymbol{\lambda}^n$'s calculated from solving the adjoint model, Equation 2.5 reduces to:

$$\delta J_A = \sum_{n=0}^{N-1} \left[\frac{\partial L^n}{\partial \mathbf{u}^n} + \boldsymbol{\lambda}^{T(n+1)} \frac{\partial \mathbf{g}^n}{\partial \mathbf{u}^n} \right] \delta \mathbf{u}^n. \tag{2.7}$$

Therefore, the required gradients of the objective function with respect to the controls are given as:

$$\frac{dJ}{d\mathbf{u}^n} = \frac{dJ_A}{d\mathbf{u}^n} = \left[\frac{\partial L^n}{\partial \mathbf{u}^n} + \lambda^{T(n+1)} \frac{\partial \mathbf{g}^n}{\partial \mathbf{u}^n} \right] \quad \forall n \in (0, \ldots, N-1)\,. \tag{2.8}$$

The $\partial \mathbf{g}^{n-1}/\partial \mathbf{x}^n$ term corresponds to the Jacobian matrix of the forward problem. Other terms in Equations 2.6 and 2.8 are also computed during the flow simulations. This is why access to the source code of the reservoir simulator is needed.

The gradients from Equation 2.8 are used to guide the search using any suitable gradient-based optimizer. The advantage of the adjoint method is that it is very efficient since each gradient computation consists of one forward reservoir simulation and one (backward) solution of Equation 2.6. In addition to requiring access to the simulator source code, another complication of using adjoints is that in order to handle nonlinear constraints, gradients of all the constraints with respect to the controls are, in general, required. Sarma et al. [45, 47] showed, however, that a constraint lumping procedure was quite effective. For the full description of the adjoint technique and its implementation in GPRS (Stanford's general purpose research simulator [11, 26]), see Sarma [45].

2.1.2 Gradient Computation from Finite Differences

Finite difference methods can also be used to compute gradients, and this was done as part of this study. It was found that using the second-order central difference stencil provided better results than the first-order forward difference. The central difference formula to approximate the components of the gradient is:

$$\frac{\partial J}{\partial u_i} = \frac{J\left(u_i + \delta u_i\right) - J\left(u_i - \delta u_i\right)}{2\delta u_i}, \quad i = 1, \ldots, N_u, \tag{2.9}$$

where N_u is the total number of control variables and δu_i is the perturbation size. The advantage of this method is that it is easy to couple to any simulator because only function evaluations are needed in order to compute the gradient. The disadvantage is that for each gradient computation, $2N_u$ simulation runs are required. This implies that the method will not be practical for a large number of controls and expensive simulation runs, which is often the case in production optimization because simulation runs can require minutes or hours of computation. An additional disadvantage of this method is that the perturbation size, δu_i, must be selected carefully for the method to yield reasonable results.

We next consider derivative-free methods, which do not need gradient information to search the solution space for an optimal solution.

2.2 Derivative-Free Methods

2.2.1 Genetic Algorithms

Genetic algorithms (GAs) are stochastic search techniques that are based on the theory of natural selection. GAs find solutions to optimization problems by generating a set of possible solutions called a population, evaluating the fitness (i.e., objective function value) of all the "individuals" in the population, ranking these individuals, and then applying certain GA operators to generate a new set of solutions (a new population). An individual refers to a potential solution to the optimization problem (a vector of control variables in the production optimization problem).

In the GA optimization process, the population of individuals evolves from generation (iteration level) to generation mainly using the GA operators of selection, crossover and mutation. The way these operators work is illustrated in Figure 2.2.

In a particular generation, the selection operator chooses the fittest individuals (individuals with the best objective function values) in the population to be parents,

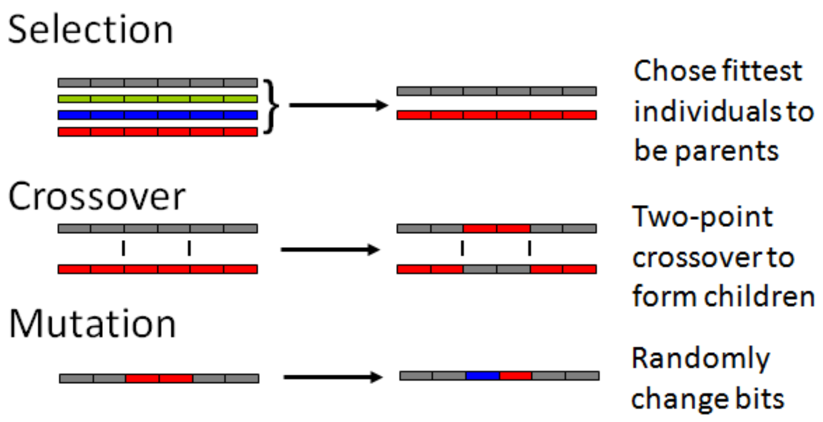

Selection
Chose fittest individuals to be parents
Crossover
Two-point crossover to form children
Mutation
Randomly change bits

Figure 2.2: Illustration of main GA operators

which will produce the next generation of individuals. The selection operator mimics the survival of the fittest evolution strategy in nature. It is the operator that ensures that the population moves towards a better region of the solution space during the optimization process. There are different kinds of selection including stochastic uniform, roulette wheel and tournament, among others. Refer to [42] for a more detailed description of these different selection techniques. In the GA implementation used for this study (GA available in the Matlab Genetic Algorithm and Direct Search Toolbox [36]), the user can specify the type of selection used. For example, in Chapter 3, it will be shown that using the tournament selection technique enables the use of a robust penalty function method with the GA for solving the nonlinearly constrained problem.

After selecting the best individuals as parents, the crossover operator combines these parents to produce children (population of individuals for the next generation). Figure 2.2 illustrates the two-point crossover technique where the elements of two parents are crossed over at two points to produce two children. Other kinds of crossover include the single-point and arithmetic for vectors with real (not binary) bits. More details of the different kinds of crossover methods can be found in [36, 42]. The crossover operator is responsible for probabilistically combining fit individuals with the possibility of producing better children. Thus the objective function tends to improve as the optimization progresses.

Another major GA operator is the mutation operator. In mutation, a specific bit or element of a parent or control vector is probabilistically changed to a new value (see Figure 2.2). The mutation is governed by the mutation rate and it gives the GA its exploratory nature because it is possible to mutate an individual into any region of the search space. Specifying the selection method, crossover method and rate, as well as the mutation type and rate, can make the GA applicable for many types of optimization problems. For this reason, GAs are regarded as robust optimizers.

Another GA feature that can be useful is elitism, where one or more of the best individuals in a population always proceed to the next generation. This option assures that the best individuals are not lost to crossover and mutation. By using the elitism option, the user ensures monotonicity in the evolution of the best individual throughout the optimization process.

Since the GA is a population-based optimization algorithm, one of the most important parameters for the GA is the population size. If the population size is too small, the GA may exhibit premature convergence because the population loses diversity quickly (the individuals become too much alike). Diversity is here defined as the average Euclidean distance between individuals in the population. On the other hand, if the population size is too large, the GA may waste computational resources, which implies that many simulations must be performed for improvement. See [42] for further assessment of this issue. In this work, a constant GA population size is used. The workflow for the GA applied here is presented in Figure 2.3.

In Figure 2.3, the box highlighted in red shows how the simulator is coupled to the GA code. Here the simulator represents the fitness function evaluation module. It is in this module that further enhancements, presented in Section 2.4, for improvement of the GA's performance will be implemented. The stopping/convergence criteria used in the GA workflow include the user defined maximum number of generations (which is typically constrained by the amount of time and computational resources available) and the number of generations without an improvement in the objective function. More details about the GA implementation can be found in [36].

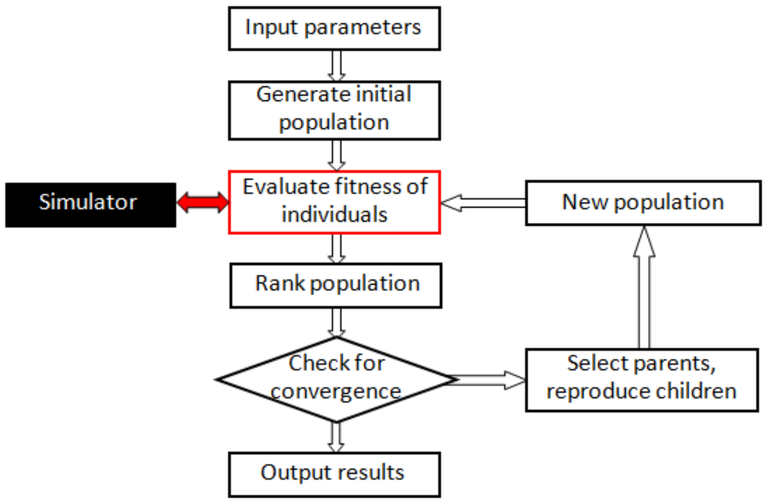

Input parameters
Generate initial population
Simulator
Evaluate fitness of individuals
New population
Rank population
Check for convergence
Select parents, reproduce children
Output results

Figure 2.3: Workflow of the GA optimization process

GAs are particularly useful for exploring complex nonsmooth search spaces with multiple local optima and are able to identify promising regions in the search space. However, the GA often shows slow convergence to an actual optimal value. In other words, the GA can get close to an optimal solution, but may be slow to achieve the exact optimum. For this reason, a hybrid implementation of the exploratory GA with a more efficient local search was implemented. This hybrid methodology is presented in Section 2.3.

2.2.2 General Pattern Search

The general pattern search (GPS) optimization method is a subset of direct search methods. These techniques have become increasingly popular in various applications where objective function gradients are not readily available or the objective function is noisy and gradients (especially approximate gradients) can be misleading. The popularity of these methods is driven in part by the fact that there is recently developed theory that proves convergence of the direct search methods to optimal solutions. The direct search methods are frame or stencil-based optimization procedures that work only with objective function evaluations (no gradient information required).

The GPS algorithm at each iteration can be divided into two steps: an optional search step and a required poll step. The search step applies user-defined search strategies and can lead to a great increase in the efficiency of the algorithm. This gives the user quite a bit of flexibility, because any other suitable optimization method can be used to search the solution space for a better point in any iteration. The algorithm goes into the poll step only when the search step is unsuccessful. It will be shown later that using a surrogate-based search step can improve the efficiency of the GPS algorithm. The poll step is required to provide convergence to an optimal solution [28]. In any particular iteration k, polling is performed by defining a set of poll points, obtained by building a frame consisting of a set of poll directions, D_k, and step-size Δ_k, and evaluating the objective function at each of these poll points. D_k is a matrix whose columns form a positive spanning basis set of $\mathbb{R}^n$, where n is the number of optimization variables and the dimension of the solution space. Figure 2.4 presents two types of frames that positively span $\mathbb{R}^2$.

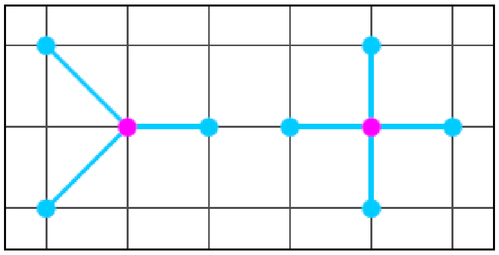


Figure 2.4: Examples of positive spanning frames for $\mathbb{R}^n$ with $n+1$ and $2n$ poll points (from [28])

The use of positive spanning directions is important in the convergence theory because it ensures that at least one of the poll directions is an ascent/descent direction. This implies that as long as the current iterate is not an optimum point, one of the poll directions in the frame will lead to an improvement in the objective function value. Figure 2.5 illustrates a sequence of polls for a simple version of the GPS algorithm with $2n$ poll directions in $\mathbb{R}^2$. Typical production optimization problems will of course entail many more than two control parameters, so the dimensionality of the search space will be much higher (e.g., 100 for the example case considered in Section 4.2.2).

From Figure 2.5 it can be seen that the basic idea for the GPS method, and most direct search methods in general, is to traverse the solution space from an initial point to

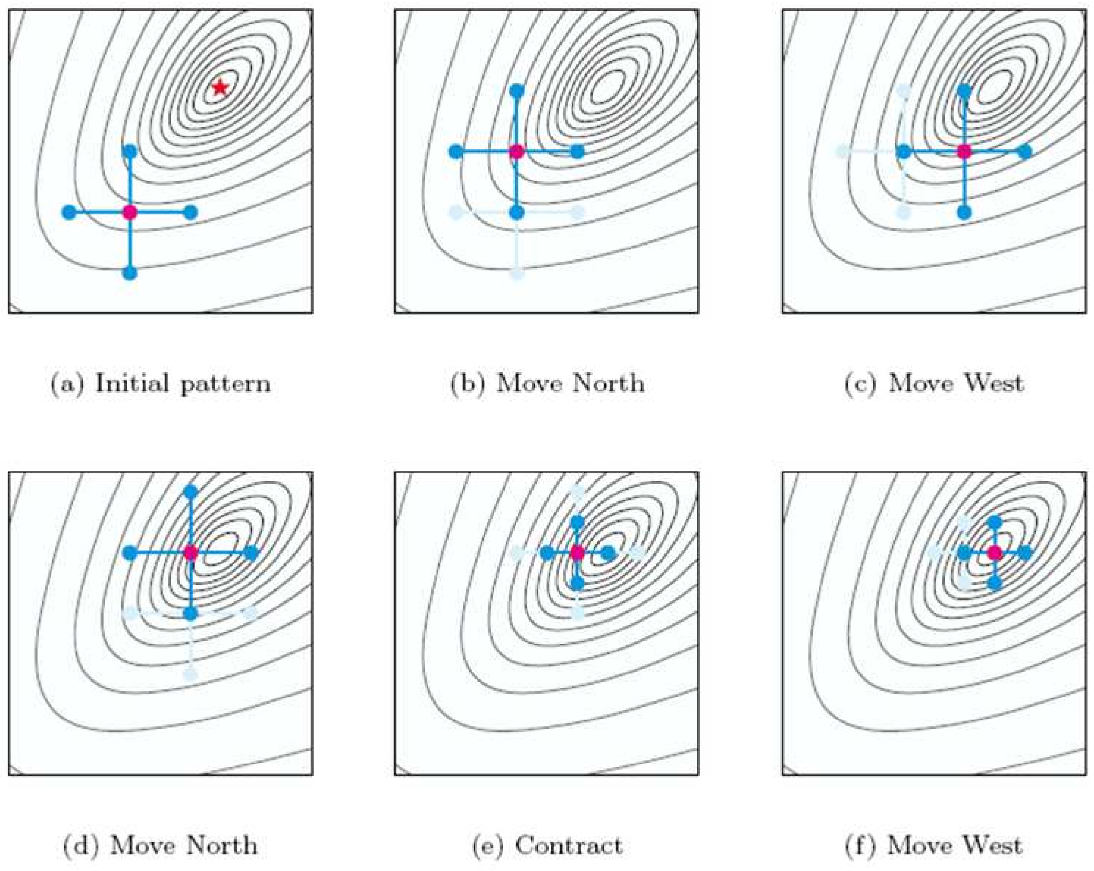


Figure 2.5: Illustration of a sequence of polls in $\mathbb{R}^n$ with $2n$ poll frame (from [28])

the optimal point using a sequence of poll steps based on frame/stencil directions and a step-size defined on an underlying mesh. A successful iteration occurs when a poll point with a better function value than the current iterate is found. An unsuccessful iteration occurs when no improvement is obtained with the current step size. When this occurs, the poll step size is reduced, unless a stopping criterion, i.e., a minimum step size, prevents this. In Figure 2.5, as the optimization progresses, the polling stencil remains the same, but this does not have to be the case.

There is also a generalized version of the GPS algorithm called the Mesh Adaptive Direct Search (MADS) algorithm. MADS uses variable direction vectors to build the polling stencil [7]. The performance of both GPS and MADS were found to be similar for the set of problems considered in this study, hence only the GPS results will be presented in Chapter 4. Refer to [36] for more details of the implementation of GPS and MADS and [28] for more algorithmic details and convergence theory for the GPS method.

The advantages of using the GPS method include the obvious fact that gradients of the objective function are not required in the optimization process. Also, the method parallelizes naturally since the objective function evaluations of the poll points in a particular iteration can be accomplished in a distributed fashion over multiple processors. The disadvantage is that in the absence of multiple processors, the method can be very slow since it requires $O(n)$ function evaluations (reservoir simulations in our case) at each iteration. In Section 2.4, a surrogate-based approach for improving the efficiency of this method will be introduced.

2.2.3 Hooke-Jeeves Direct Search

The Hooke-Jeeves Direct Search (HJDS) algorithm is also a stencil-based method, but it traverses the solution space in a different fashion than the GPS method. The HJDS optimization strategy is based on two types of moves: exploratory moves and pattern moves. These moves are illustrated in Figure 2.6 for an optimization sequence in $\mathbb{R}^2$.

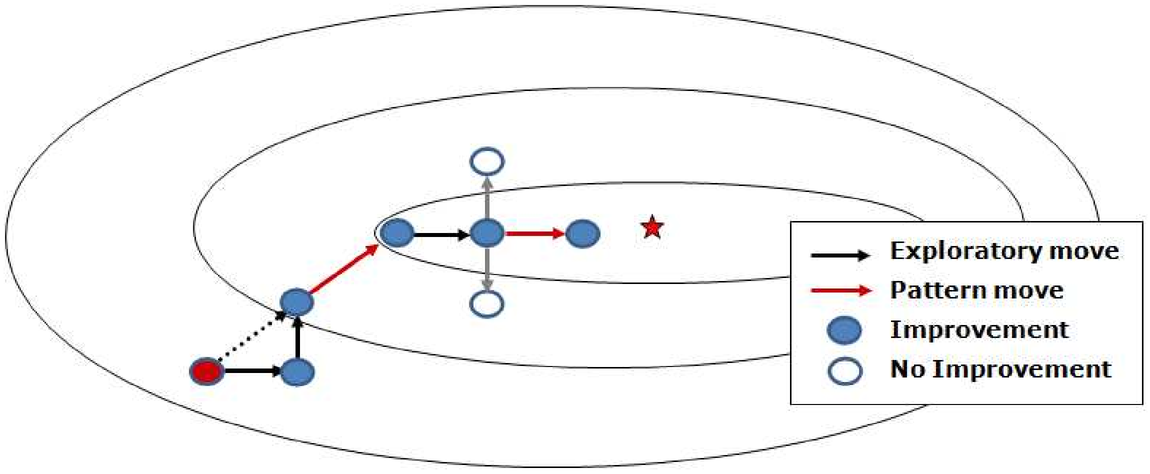


Figure 2.6: Illustration of exploratory and pattern moves in HJDS

The algorithm begins with a base point $\mathbf{u}$ and step size Δ. During the exploratory move, the objective function is evaluated at successive perturbations of the base point in the search directions $\{\mathbf{d}_j\}$, where $\mathbf{d}_j$ is the jth column of the direction matrix D (which is equal to the identity matrix I for the implementation used in this study). The best value $J_b = J(\mathbf{u}_b)$ and the best point $\mathbf{u}_b$ are recorded, and $\mathbf{u}_b$ is then set to $\mathbf{u}$. In the exploratory sampling, each coordinate direction is searched by first evaluating $J(\mathbf{u}_b + \mathbf{d}_j)$ and only evaluating $J(\mathbf{u}_b - \mathbf{d}_j)$ if $J(\mathbf{u}_b + \mathbf{d}_j)$ is less than J_b (for a maximization problem). The exploratory phase will either produce a new base point or it will fail, meaning that no improving point was found, in which case the step size will be reduced.

This HJDS algorithm is inherently serial in nature since the perturbations of element u_i depend on the results of the perturbations of element u_{i-1}. If the exploratory phase succeeds in finding an improving point, $\mathbf{u}_b$, then the underlying successful search direction $\mathbf{d}^b$ (dotted arrow in Figure 2.6) is given by $\mathbf{d}^b = \mathbf{u}_b - \mathbf{u}$. At this point, rather than center the next exploration at $\mathbf{u}_b$, the algorithm performs the pattern move, which is an aggressive step that tries to move further in the underlying successful direction. The algorithm centers the next exploratory move at $\mathbf{u}_c = \mathbf{u} + 2\mathbf{d}^b = \mathbf{u}_b + \mathbf{d}^b$. If the second exploratory move centered at $\mathbf{u}_c$ fails to improve upon J_b, then the exploratory move centered at $\mathbf{u}_b$ is tried. If that fails, then Δ is reduced, $\mathbf{u}$ is set to $\mathbf{u}_b$ and the process is repeated.

In the implementation used in this study, the optimization is stopped either after a predefined number of iterations or if a certain number of iterations occurs without any improvement in the objective function, or if convergence is achieved. Convergence in this case is defined as Δ decreasing below a user-defined tolerance. For more details about the HJDS method, see [23, 27].

The advantage of the HJDS method is that it uses a very efficient search strategy in traversing the solution space. This search strategy is only based on function evaluations and does not require gradient computation, even though it moves along gradient-like search directions. The main drawback of the HJDS method is that it is inherently serial because of its search strategy and thus cannot be improved by straightforward distributed computing.

The HJDS, GPS and gradient-based methods are considered to be local optimizers that are dependent on the initial guess needed to begin the optimization process. However, in situations where the objective function surface contains multiple local optima, both the HJDS and GPS algorithms have the advantage over the gradient-based methods that, with a large enough initial step size, they can avoid many of the local optima that would trap gradient-based methods, enabling them to move into better regions of the solution space. This does not mean that these direct search methods will find the global optimum, but that they can provide better solutions than gradient-based optimizers in some cases.

2.3 Hybrid Methodology

Each of the methods presented has advantages and disadvantages. It is possible to combine some of the methods so as to mitigate some of the disadvantages. The hybrid methodology consists of two phases. The first phase is the more exploratory phase of the algorithm, while the second phase represents a more efficient local search. The hybrid implementation used in this study consists of the first phase optimization with the GA and the second phase optimization with the SQP method with adjoint-based gradient computation, using the result from the first phase as the initial guess.

The GA was chosen for the first phase because of its robustness, ability to broadly search the solution space (exploration) and the fact that it does not require a user-specified initial guess since the initial population is generated randomly. Also, the GA is population-based and generates a set of possible solutions that can be used as different initial guesses for the second phase. For the second phase, the SQP method with adjoint-based gradient computation was chosen for its efficiency and fast convergence to a local optimum. The idea is that the GA will generate a point in the vicinity of the optimal solution, and the SQP method will quickly converge to this solution.

2.4 Computational Enhancement Strategies

One of the main drawbacks that all the derivative-free methods have is that the computational cost (number of function evaluations) scales with the number of optimization variables. So, for production optimization problems with a large number of variables (≥ 100), these methods become impractical in their basic form. Several researchers have addressed this issue and have devised various strategies to reduce the computational cost of derivative-free methods [15, 27]. We now discuss some of these strategies.

2.4.1 Surrogate-based Optimization

By far the most expensive part of the production optimization process is the evaluation of the objective function because this requires computationally expensive reservoir simulations to be performed. One way to reduce this high computational cost is by using surrogates or proxies for the reservoir simulator. Two such approaches are the use of computationally efficient reduced numerics or reduced physics models, such as reduced order models and streamline simulators, and the use of functional surrogates or response surfaces obtained from some kind of interpolation of the function values found in previous iterations.

The use of functional surrogates such as artificial neural networks (ANN), kriging response surfaces and statistical proxies with GAs has been investigated by several researchers within the context of the well placement problem [21, 41, 52]. In this study, the use of ANN with the GA and the use of the kriging surrogate with GPS will be investigated for the production optimization problem.

Genetic Algorithm with Artificial Neural Networks

Artificial neural networks (ANNs) are composed of simple elements inspired by biological nervous systems. A neural network can be trained to perform a particular function by adjusting the values of the connections (weights) between elements. Typically, neural networks are adjusted, or trained, so that a particular input leads to a specific target output. This training situation is illustrated in Figure 2.7. See [18] for a more detailed description of ANNs and their implementation.

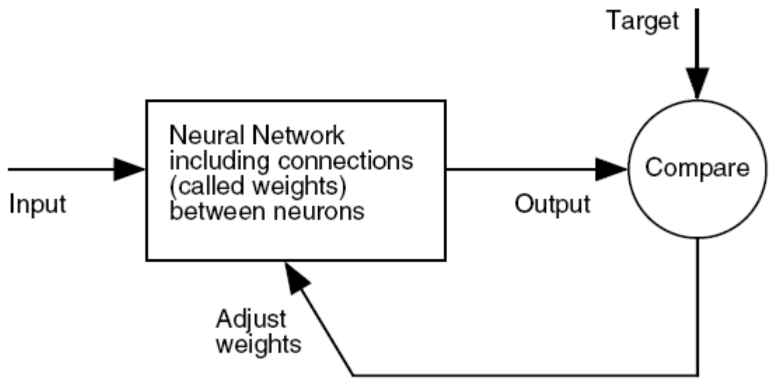

Target
Neural Network including connections (called weights) between neurons
Input
Output
Compare
Adjust weights

Figure 2.7: Illustration of the workflow for training an ANN (from [18])

In this study, we apply ANNs to improve the efficiency of the GA used to solve the production optimization problem. Recall from Figure 2.3 that the simulator is coupled to the GA at the function evaluation module (pictured in red). This is the module where we can include the ANN. The basic idea is to run and save simulated individuals with their corresponding objective function values for a few generations of the GA, and to then use these points to train an ANN. At subsequent generations, instead of evaluating the fitness of all of the individuals in the population using the reservoir simulator, only a user-defined fraction of the population will have their fitness evaluated by the simulator. For the remainder of the population, fitness will be evaluated using the trained ANN.

With the ANN surrogate, expensive function evaluations will not be wasted on portions of the GA population that are not expected to be used as parents for the next generation. By only simulating a small portion of the population, as opposed to the full population, significant computational savings can be achieved. The disadvantage of using this method is that in order to construct sufficiently accurate ANN approximations to the objective function, a large set of inputs and corresponding fitnesses are required. This means we may need to simulate many generations in the beginning of the optimization process to adequately train the ANN.

General Pattern Search with DACE Surrogate

Recall from Section 2.2.2 that the GPS method is divided into search and poll steps. Booker et al. [8] used this two-step characteristic of the GPS method to develop the Surrogate Management Framework (SMF) where a surrogate is created and calibrated in the beginning of the optimization process and then used in the GPS search steps. For a surrogate-based search step, an approximation $S(\mathbf{u})$ to the objective function $J(\mathbf{u})$ is optimized and the resulting point evaluated with the simulator. If the solution from the surrogate model gives an improvement in the objective function, the search is declared successful, the surrogate model is updated, and the algorithm moves to the next iteration without going into the poll step. In the implementation used here, a response surface obtained by the Matlab kriging toolbox, DACE, is used as the surrogate. See [33] for a detailed description of the theory and implementation of the DACE kriging toolbox. Figure 2.8 presents a kriging response surface generated by this Matlab toolbox for a 2D problem.

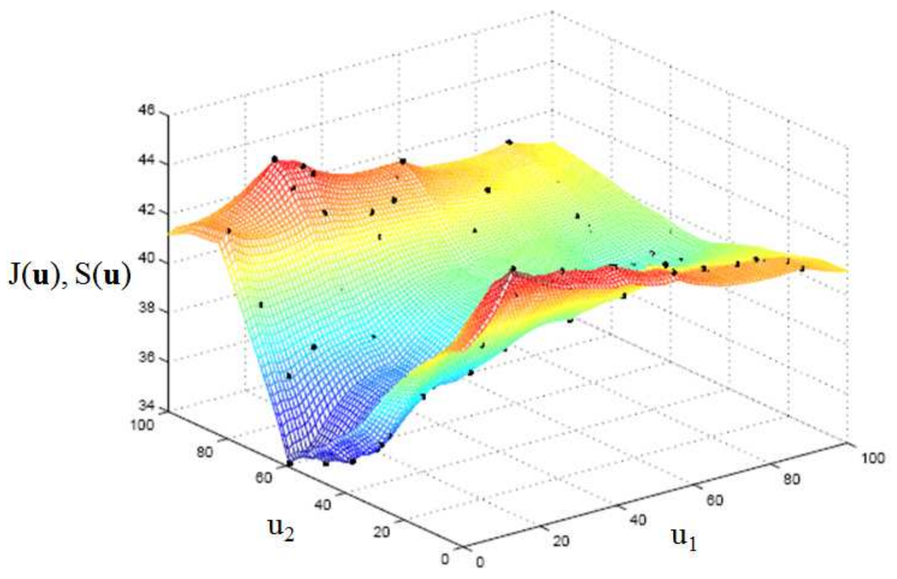


Figure 2.8: Kriging response surface for 2D problem with data points shown in black (from [33])

The inclusion of the surrogate-based search step in the GPS algorithm leads to a significant reduction in computational expense. As will be shown later in the results section, the GPS algorithm with a surrogate-based search step shows very large improvement within the first few iterations/function evaluations because efficient searching of the solution space is first accomplished using the computationally inexpensive surrogate model.

2.4.2 Distributed Computing

The computational efficiency of some of the optimization techniques presented above can be significantly enhanced by distributing the objective function evaluations (reservoir simulations) over several computer processors; i.e., by running each simulation on a different processor. If a parallel reservoir simulator is available, a given simulation run could be distributed over many processors, though this degree of parallelization is not considered here. The speedup obtained by the straightforward distribution of the objective function evaluations will depend on the number of processors available, but we have found that even with as few as 10 processors, significant speedup can be achieved.

Of the optimization methods presented, the gradient-based methods with finite difference gradients (using Equation 2.9), the GA, and the GPS algorithms can be run in a straightforward distributed fashion. This is because for all these techniques, a number of independent simulations can be performed simultaneously. Another time-saving strategy implemented in the algorithms considered in this work is caching. This means that some number of function evaluations are saved so that none is repeated. This plays a significant role in the direct search methods where points may be re-evaluated due to the regular nature of the polling stencils.

This concludes the basic description of the optimization methods considered in this study, together with some strategies for improving efficiency. We note that, as described here, these methods handle unconstrained optimization problems. The extension to bound and linearly constrained problems is straightforward. For the GA, crossover and mutation operations are performed such that the generated individuals stay within the boundaries. For the direct search methods, the poll directions are chosen such that for bound constraints, the poll step is modified to keep poll points either on or within the bounds. For linear constraints, the poll directions are chosen to be parallel to the active linear constraints at the current iterate. Refer to the references presented throughout this chapter for more detailed descriptions of how the respective algorithms perform linear constraint handling.

The actual implementations of all the algorithms described in this chapter handle bound and linearly constrained optimization problems. The results presented in Section 4.1 are for bound-constrained production optimization problems. The following chapter presents nonlinear constraint handling techniques that enable the methods to handle fully constrained production optimization problems.

Chapter 3

Optimization with Nonlinear Constraints

The previous chapter presented descriptions of the basic optimization methods considered in this study. Additional treatments are required to handle nonlinear constraints, which commonly arise in practical production optimization problems. For instance there could be a field wide maximum water injection constraint or a minimum field oil rate constraint with wells that are BHP controlled. The following sections describe nonlinear constraint handling techniques for the optimization methods presented in Chapter 2.

3.1 Sequential Quadratic Programming

The sequential quadratic programming (SQP) method represents the state-of-the-art in gradient-based nonlinear programming methods. The development of the SQP method presented here follows that of [40]. The method focuses on the solution of the Karush-Kuhn-Tucker (KKT) equations. Referring to Equation 1.1, the KKT equations, which are necessary conditions for optimality for a constrained optimization problem [40], can be stated as:

$$\begin{aligned} \nabla J(\mathbf{u}^*) + \sum_{i=1}^{m} \lambda_i^* \cdot \nabla c_i(\mathbf{u}^*) &= 0, \\ c_i(\mathbf{u}^*) &\leq 0 \qquad i = 1, \cdots, m, \\ \lambda_i^* &\geq 0 \qquad i = 1, \cdots, m, \end{aligned} \tag{3.1}$$

where m is the total number of inequality constraints in the problem, $\mathbf{u}^*$ represents the constrained optimal point, and the λ_i^*'s are the optimal Lagrange multipliers. In Equation 3.1, we consider only the maximum constraint violation for each type of constraint. When the maximum constraint violation is satisfied, constraints at all time steps will be honored. The first equation describes a canceling of the gradients between the objective function and the active constraints at the solution point, $\mathbf{u}^*$. For the gradients to be canceled, Lagrange multipliers, $\lambda_i, i = 1, \cdots, m$, are necessary to balance the magnitudes of the objective function and constraint gradients. The idea of the SQP method is to closely mimic Newton's method for unconstrained optimization, within the context of constrained optimization. At each iteration, an approximation is made of the Hessian of the Lagrangian function using a quasi-Newton updating method, and this Hessian is used to generate a QP subproblem whose solution is used to form a search direction for a line search procedure.

Given the general fully constrained production optimization formulation, Equation 1.1, SQP involves the formulation of a QP subproblem based on a quadratic approximation of the following Lagrangian function:

$$L(\mathbf{u}, \boldsymbol{\lambda}) = J(\mathbf{u}) + \sum_{i=1}^{m} \lambda_i \cdot \nabla c_i(\mathbf{u}). \tag{3.2}$$

The resulting QP subproblem to be solved at each iteration of the SQP method is:

$$\begin{gathered} \min_{\mathbf{d} \in \mathbb{R}^{\mathbf{n}}} \frac{1}{2} \mathbf{d}^T H_k \mathbf{d} + \nabla J(\mathbf{u}_k)^T \mathbf{d}, \\ \text{subject to:} \\ \nabla c_i(\mathbf{u}_k)^T \mathbf{d} + c_i(\mathbf{u}_k) \leq 0 \quad i = 1, \cdots, m, \end{gathered} \tag{3.3}$$

where H_k is the approximation of the Hessian of the Lagrange function at iteration k and $\mathbf{d}$ is the desired search direction. This subproblem can be solved using any QP algorithm (see [40] for a description of some QP algorithms). The solution is then used to form a new iterate:

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \alpha_k \mathbf{d}_k. \tag{3.4}$$

The step length parameter, α_k, is determined by an appropriate line search procedure so that a sufficient decrease in a merit function is obtained. Refer to [37, 40] for more detailed descriptions of the SQP method, including the updating of the Hessian matrix, solution of the QP subproblem, and the line search using a merit function. The SQP implementation used in this work is available in the Matlab optimization toolbox. In the current implementation, the gradients required in Equations 3.2 and 3.3 are provided by adjoint models or finite differences for the objective function gradients and by finite differences (using an equation similar to Equation 2.9) for each of the constraint gradients.

3.2 Penalty Functions

The penalty function method for constraint handling is a generic method that can be used with most optimization algorithms. Its performance, however, is not satisfactory in some cases. This section introduces penalty functions, discusses their advantages and disadvantages, and describes a more robust formulation for use with the GA.

3.2.1 Basic Description

The penalty function method for handling inequality constraints in optimization problems involves modifying the objective function by including a penalty term which depends on the constraint violation. By modifying the objective function in this way, the production optimization problem (Equation 1.1) can be expressed as:

$$\begin{gathered}\max_{\mathbf{u}\in\mathbb{R}^n} F(\mathbf{u}) = J(\mathbf{u}) - \rho h(\mathbf{u}), \\ \text{subject to:} \\ \mathbf{l}_B \leq A\mathbf{u} \leq \mathbf{u}_B, \\ \text{where} \quad h(\mathbf{u}) = \sum_{i=1}^{m} \max(0, c_i(\mathbf{u}))^2. \end{gathered} \tag{3.5}$$

In this equation, $h(\mathbf{u})$ is the lumped constraint violation function, which is zero when all the constraints are satisfied and has a positive value when any of the constraints are violated, F is the modified objective function, and $\rho > 0$ is called the penalty parameter. From Equation 3.5 it can be seen that the penalty function approach changes the original problem with nonlinear inequality constraints to a problem with only bound and linear constraints, which can be handled by any of the optimization methods presented in Chapter 2. In the above equation, a single penalty parameter is used because the constraints are normalized. The penalty parameter acts to ensure that the constraint violation is about the same order of magnitude as the objective function value.

The optimal solution of $F(\mathbf{u})$ depends on the choice of the penalty parameter ρ. If ρ is chosen correctly, the solution of the optimization problem can be efficient and accurate. On the other hand, a poor choice of ρ could lead to poor performance and inaccurate solutions. If ρ is too small, the distortion of the objective function is small, but the optimum of $F(\mathbf{u})$ may not be near the true constrained optimum. If ρ is too large, the distortion of the objective function might be so severe that $F(\mathbf{u})$ may have artificial locally optimal solutions not present in the original problem [17]. In addition, if ρ is too large, then any monotonic method would be forced to follow the nonlinear constraint manifold very closely, resulting in slow convergence.

In order to avoid a poor choice of ρ, iteration is usually performed. This involves starting with an initial guess for ρ, solving a subproblem, and then updating ρ at every iteration. This method may work well but has limitations in practice because it requires a lot of function evaluations. Another way to obtain a reasonable choice for

ρ is by tuning, where different intuitive choices of ρ are used to solve the optimization problem and the one with the best result is selected. Again, this method could require many function evaluations. Since in solving the production optimization problem the function evaluations involve running a reservoir simulator, these evaluations can be very expensive. There have been different adaptations of the basic penalty function method for specific use with particular optimization algorithms. The following subsection presents a more robust penalty function formulation, proposed by [17], for use with the GA.

3.2.2 Parameterless Penalty Function for GAs

In this method, a penalty term is included in the objective function in order to penalize infeasible solutions, but this term differs from that appearing in Equation 3.5. Our description here follows that presented by Deb in [17].

The parameterless penalty method uses a tournament GA selection operator, where individuals in a randomly selected portion of the current population are compared. During the tournaments, the parameterless penalty function formulation enforces the following criteria:

1. Any feasible solution is preferred over any infeasible solution.
2. When comparing a set of feasible solutions, the one with the better objective function value, $J(\mathbf{u})$, is preferred.
3. When comparing a set of infeasible solutions, the one with the smaller constraint violation, $h(\mathbf{u})$, is preferred.

Recall from the previous section that the penalty parameter is needed to make the constraint violation and objective function about the same magnitude. In this parameterless penalty formulation, the penalty parameter is not needed because individuals are never compared in terms of both objective function and constraint violation in the three tournament scenarios presented above. The modified objective function based on the parameterless penalty function for use in production optimization using the GA with tournament selection is:

$$F(\mathbf{u}) = \begin{cases} J(\mathbf{u}) & \text{if } c_i(\mathbf{u}) \leq 0 \;\; \forall i = 1, 2, \cdots, m, \\ J_{min} - \displaystyle\sum_{i=1}^{m} max\left(0, c_i(\mathbf{u})\right)^2 & \text{otherwise,} \end{cases} \tag{3.6}$$

where the parameter J_{min} is the objective function value of the feasible solution with the smallest value of J in the population (for a maximization problem). Thus, the fitness of an infeasible solution depends not only on the amount of constraint violation, but also on the current population of solutions. However, the fitness of a feasible solution is simply equal to its objective function value.

This parameterless penalty function constraint handling technique is illustrated in Figure 3.1 using a minimization example presented in [17]. The first plot shows the objective function contours, constraint boundary, and the unconstrained (green star) and constrained (red star) optima. The second plot shows the original objective function contours within the constraint boundary and the modified function contours outside this boundary.

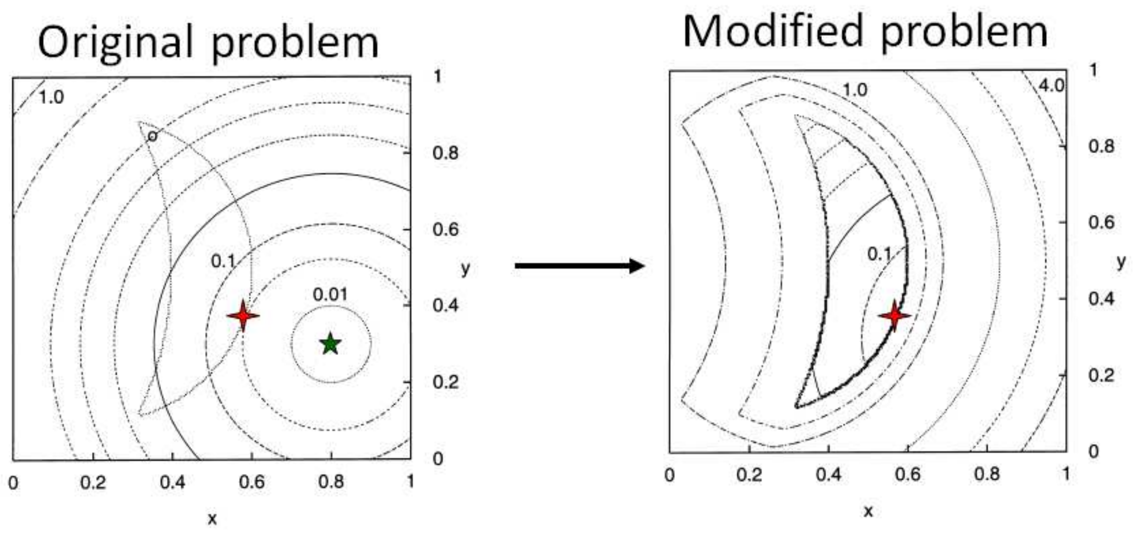


Figure 3.1: Illustration of how parameterless penalty function modifies original problem (from [17])

We see in Figure 3.1 that the parameterless penalty formulation modifies the problem such that the contours are unchanged inside the feasible region, whereas they become parallel to the constraint surface outside the feasible region. Thus, when most solutions in a population are infeasible, the search forces solutions to move towards the feasible region. Once sufficient solutions exist inside the feasible region, the search is directed by the objective function alone. In other words, through the course of the GA optimization, individuals are attracted to feasible regions. Then, the selection operator works with the true objective function value to focus the search in the identified feasible region. This leads to a high degree of robustness for this parameterless penalty function approach since a feasible output is generally obtained.

3.3 The Filter Method

Filter methods are step acceptance mechanisms that avoid the robustness issues present in the traditional penalty function methods. Instead of combining the objective function and constraint violation into a single function, Equation 1.1 is viewed as a bi-objective optimization in which we seek to maximize/minimize $J(\mathbf{u})$ and minimize $h(\mathbf{u})$. The second objective is more important because the solution determined by the optimization algorithm must be feasible. For the case of minimization, a point $\mathbf{u}_a$ is said to *dominate* another point $\mathbf{u}_b$ if and only if $J(\mathbf{u}_a) \leq J(\mathbf{u}_b)$ and $h(\mathbf{u}_a) \leq h(\mathbf{u}_b)$. A filter is defined as a list of pairs $(h(\mathbf{u}_f), J(\mathbf{u}_f))$ such that no pair dominates another pair. An iterate $\mathbf{u}_k$ is acceptable to a filter if $(h(\mathbf{u}_k), J(\mathbf{u}_k))$ is not dominated by any pair in the filter. Refer to [20, 40] for a more detailed introduction to and history of filter methods. The following presents the use of the filter method with GPS (this description follows that given by Audet and Dennis in [6]).

In adapting the filter method for use in GPS, two types of solutions are defined: the best feasible solution and the closest-to-feasible infeasible solution. These are presented in objective function-constraint violation space in Figure 3.2, together with a filter at some iteration k, for a minimization problem. In this figure, f_k^F represents the best feasible solution up to iteration k. The point $\left(h_k^I, f_k^I\right)$ represents the closest-to-feasible infeasible solution, which is the least infeasible filter point. During polling in the GPS algorithm, either one of these solutions can be used as the poll center. Even if there is a best feasible solution, it may still be useful to poll around the least infeasible point, which might have a lower objective function value, in order to explore a different part of the parameter space [20]. In the GPS filter algorithm, an iteration that generates an unfiltered point, i.e., a point below the filter envelope defined in Figure 3.2, is considered a successful iteration.

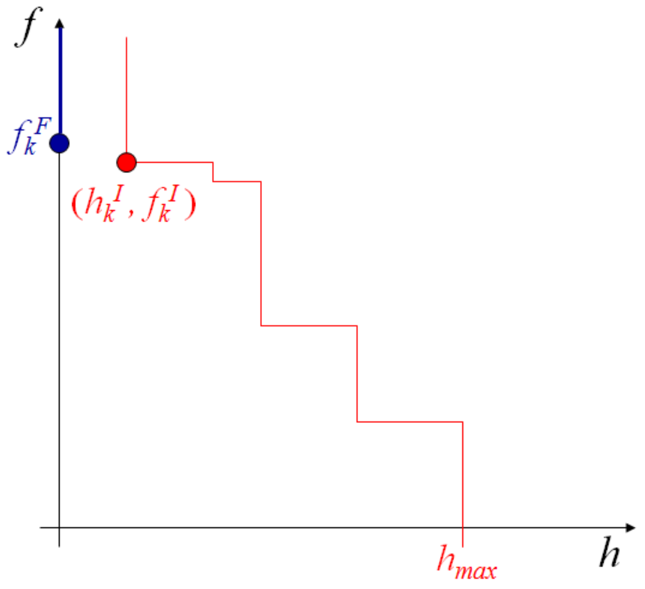


Figure 3.2: Illustration of a filter at iteration k (from [6])

The most useful successful iterations are those that produce an unfiltered feasible iterate, $\mathbf{u}_{k+1}$, which improves on the objective function value of the best feasible point. There are also successful iterations that do not produce a feasible iterate but improve the closest-to-feasible infeasible solution or add extra elements to the filter. When unfiltered points are found at successful iterations, the filter is updated. At unsuccessful iterations, the poll size parameter Δ is decreased, as well as h_{max}, the maximum allowable constraint violation for any particular iteration. It is evident from this description that the GPS filter method accepts points that are infeasible, as long as they are unfiltered. This is done because a better feasible point may be found at later iterations of the search process. This implies that the GPS with filter method produces a non-monotonic search of the constrained solution space. See [1, 6] for more discussion on the GPS with filter algorithm used in this work.

3.4 Hybrid Methodologies

Hybridization of the derivative-free methods with general constraint handling is also considered in this work. The hybrid methodology consists of two phases. First, an exploratory search algorithm with a robust constraint handling strategy is used to obtain an appropriate initial guess for the second phase, in which an efficient local optimizer is applied.

For the first phase of the hybrid method, the GA with the parameterless penalty function was chosen. This algorithm was used because it is a robust global search algorithm that can explore the solution space, identify feasible regions, and output solutions that are not only feasible, but also display high fitness. The advantage of hybridizing the GA with a local optimization method is that the GA then does not require a very large population because convergence to an optimum is not required. All that is required of the GA is to search and identify favorable regions within the feasible area of the solution space, which can be accomplished with a smaller population size.

For the second phase of the method, one of the other derivative-free optimization techniques with traditional penalty function handling can be used. It is recommended to use HJDS if running serially and GPS if running in a distributed computing environment. The penalty parameter for this second phase of the hybrid method is obtained from the result provided by the GA in the first phase. In the implementation used in this study, the penalty parameter is prescribed to be an order of magnitude higher than the objective function value of the solution obtained from the first step. Further tuning of this parameter is possible (though potentially costly) and could lead to improved solutions. As we will see in Chapter 4, this hybridization strategy is very effective in solving nonlinear inequality constrained production optimization problems.

Chapter 4

Example Applications

The methods described in the preceding chapters will now be applied to some synthetic production optimization problems. First, the basic methods without nonlinear constraint handling are applied to cases with only bound constraints. The results for both derivative-free and gradient-based methods are compared. Next, the optimization methods with nonlinear constraint handling techniques are applied to fully constrained production optimization cases. For these cases, the benefits of using the parameterless penalty function, filter and hybrid methods are demonstrated.

In all the production optimization cases presented, the problem involves the maximization of undiscounted net present value (NPV) by adjusting the injector and producer BHPs. Specifically, we seek to maximize the objective function $J(\mathbf{u})$, where

$$J(\mathbf{u}) = r_o Q_o(\mathbf{u}) - c_{wp} Q_{wp}(\mathbf{u}) - c_{wi} Q_{wi}(\mathbf{u}) . \tag{4.1}$$

Here r_o is the price of oil (\$/STB), c_{wp} and c_{wi} are the cost of handling produced water and the cost of water injection (\$/STB), and Q_o, Q_{wp} and Q_{wi} are the cumulative oil production, water production and water injection (STB) obtained from the reservoir simulator. The reservoir simulator used in all these cases was Stanford's general purpose research simulator (GPRS) [11, 26].

4.1 Optimization Cases With Bound Constraints

The two cases presented in this section are bound-constrained problems that are solved using the base optimization algorithms, without any modifications for nonlinear constraint handling. Since the problems involve maximizing the undiscounted NPV by optimizing the well BHPs, the only constraints are bounds on the injector and producer BHPs.

4.1.1 Case 1: Section of Stanford VI Reservoir Model

Optimization Problem Description

The reservoir model used in this case is a $30 \times 40 \times 10$ portion of the Stanford VI reservoir model. See Castro [13] for a detailed description of the Stanford VI reservoir model. Figure 4.1 shows the x-direction permeability field, together with the injector and producer well locations, for the portion of the model used in this study.

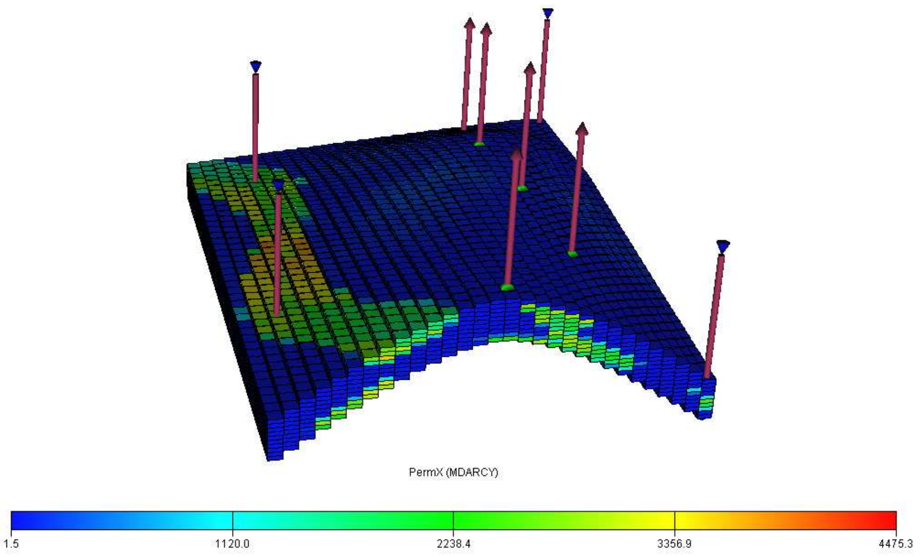


Figure 4.1: Section of Stanford VI reservoir model, showing permeability field and wells

The simulation model involves two-phase oil-water flow. Figure 4.2 presents the oil and water relative permeability curves. Capillary pressure effects were not included in the simulation model. The reservoir simulation and production optimization parameters for this case are summarized in Table 4.1. The reservoir simulation time was 1000 days, with BHPs updated every 100 days (10 control periods). As there are 4 injectors and 5 producers in the reservoir model, the total number of optimization variables in this problem is 90 (9 wells × 10 control periods).

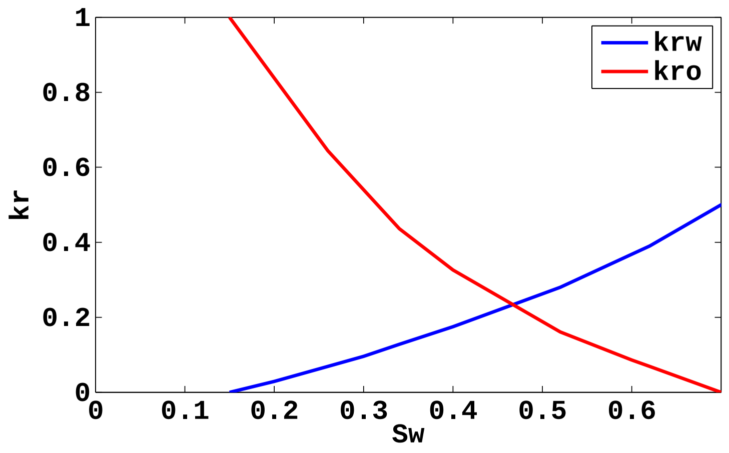


Figure 4.2: Relative permeability curves for the oil and water phases (Case 1)

Table 4.1: Case 1 simulation and optimization parameters

Grid cell dimensions	$50 \times 50 \times 15$ ft^3
Initial pressure, p_i	5080 psi
c_R at ref. pressure	5×10^{-6} psi^{-1}
μ_o at p_i	1.18 cp
μ_w at p_i	0.325 cp
ρ_o	54 lbm/ft^3
ρ_w	58 lbm/ft^3
B_o and B_w at p_i	1.00 RB/STB
r_o	\$90/STB
c_{wp}	\$18/STB
c_{wi}	\$9/STB
Injector BHP range	6000 - 8000 psi
Producer BHP range	2000 - 4000 psi

Production Optimization Results

The optimization algorithms described in Chapter 2 were applied to this production optimization problem. The performance of the algorithms is compared in Figure 4.3, which presents the evolution of the NPV throughout the optimization as a function of the number of simulations. The results presented in Figure 4.3 are summarized in Table 4.2.

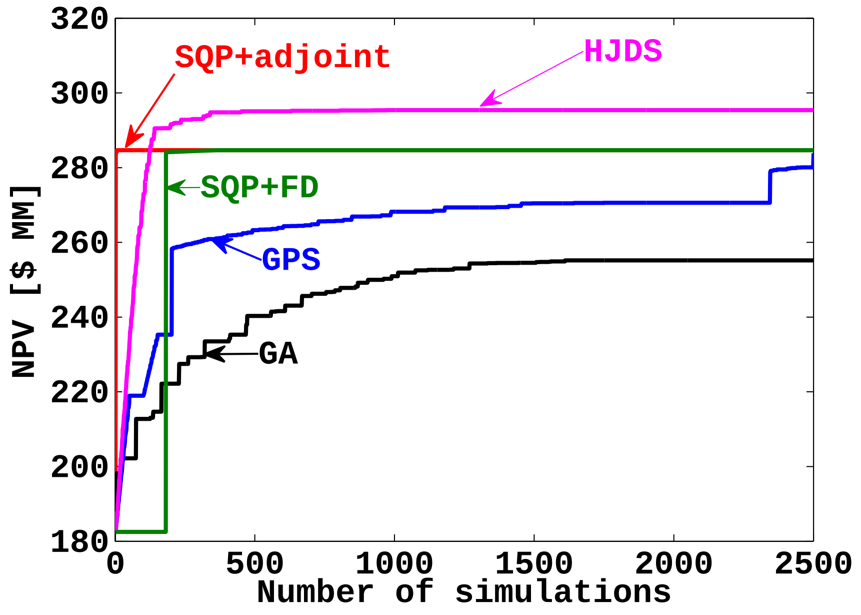


Figure 4.3: Comparison of the performance of gradient-based and derivative-free algorithms (Case 1)

Table 4.2: Summary of performance of different optimizers for Case 1

Optimization algorithm	Number of simulations	Max. NPV [$ MM]
SQP+adjoint	6	284.7
SQP+FD	364	284.7
HJDS	452	295.0
GPS	2500	283.9
GA	2500	255.2

The same initial guess was used for the SQP+adjoint, SQP+FD, GPS and HJDS methods. This initial guess (or base case) was constant BHP control of 7000 psi in all injectors and 3000 psi in all producers. The initial population of the GA was generated randomly. Figure 4.3 highlights the characteristics of the respective optimization methods. These characteristics include the efficiency of the gradient-based techniques and their convergence to a local optimum. The GA was run with a population of 50 individuals, which can be considered to be small given the size of the problem (90 control variables). Recall that population size is limited by the available time and computing resources. The GA exhibits premature convergence because the diversity of the individuals decreases rapidly in the high dimensional space. This is likely due to the small population size used for this high dimensional search.

Even though the GPS and HJDS methods belong to the same family of pattern search optimizers, HJDS displays great efficiency for a derivative-free method because of the gradient-like manner in which it conducts its search of the solution space. Also, even though they are regarded as local optimizers, if one seeds the algorithms with a large enough initial step size, these pattern search methods can avoid some local optima that will trap the gradient-based methods, and thus obtain better solutions. This behavior is observed in this example. The use of a large initial step size in the pattern search methods is akin to performing an initial, though very coarse, exploration of the search space before focusing on a local region and converging within that region.

Figure 4.4 compares some of the resulting optimum BHP profiles from the efficient gradient-based method (SQP+adjoint) and HJDS. From this figure it can be seen that both algorithms give very different “optimal” solutions with the same initial guess. Again this is due to the fact that the HJDS method (and pattern search methods in general) seeded with a large initial step size is able to avoid some local optima.

Figure 4.5 presents plots comparing the cumulative injection and production profiles of the base case and two optimized cases. Here we see that even though the optimized solutions produce about 10% less oil than the base case, the improvement in NPV comes from the more efficient water handling. Both the SQP+adjoint and HJDS algorithms provide BHP controls that reduce the cumulative water injection by about 45%. The SQP+adjoint controls reduce the cumulative water production by 65%

and the HJDS solution provides a 67% reduction in cumulative water produced. This accounts for the higher NPV from the HJDS solution (evident in Figure 4.3 and Table 4.2).

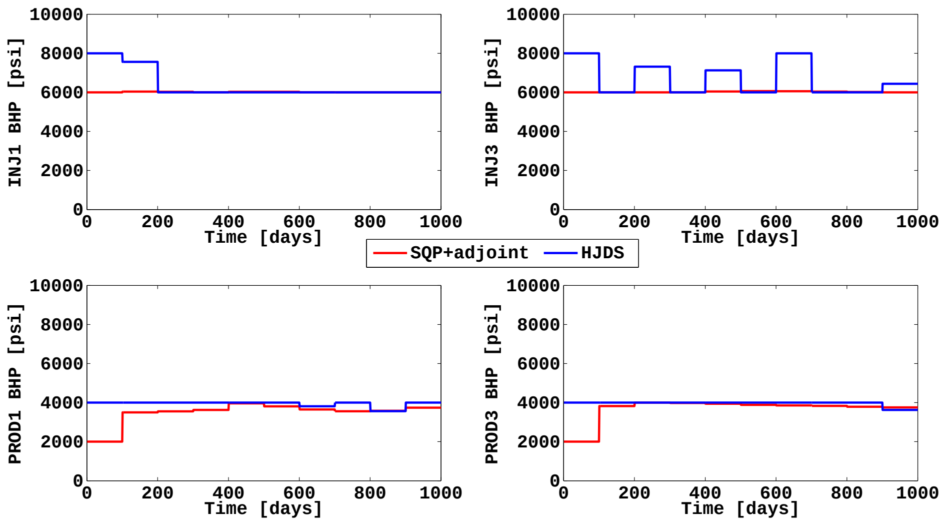
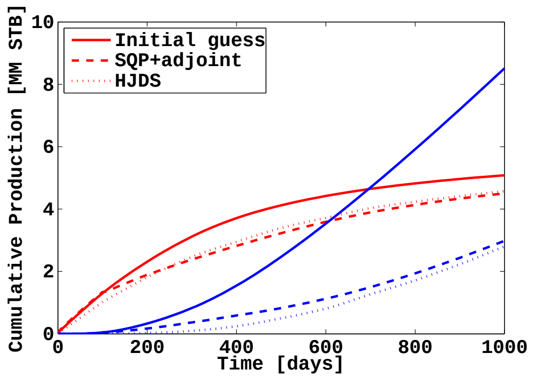
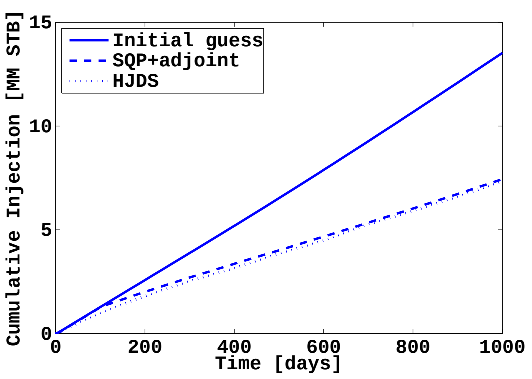


Figure 4.4: Comparison of the resulting BHP profiles for two injectors and two producers from SQP+adjoint and HJDS optimization algorithms



(a) Cumulative oil (red) and water (blue) production



(b) Cumulative water injection

Figure 4.5: Comparison of Case 1 cumulative production and injection profiles for the base case, SQP+adjoint and HJDS solutions

Hybrid Optimization Results

Recall that in Section 2.3 a hybrid method combining the GA and the adjoint-based optimization procedure was presented. Here, the first phase of the method was run with the GA with a population of 50 individuals until convergence (sufficient reduction in population diversity). The individuals in the final GA population were then divided into five clusters. The individuals closest to the cluster centroids were used as five initial guesses for the second phase of the hybrid method, optimization with SQP+adjoint. The results from the second phase of this hybrid procedure are presented in Figure 4.6.

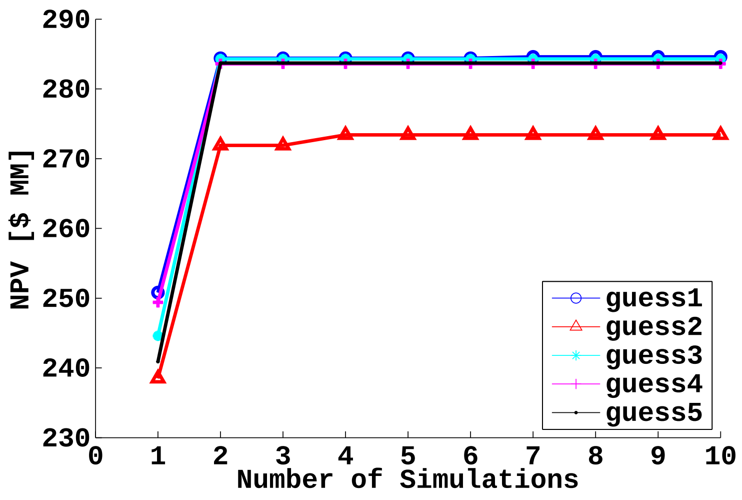


Figure 4.6: Comparison of the five SQP+adjoint runs using initial guesses from the final population of the GA

The efficiency of the SQP+adjoint algorithm is again evident in Figure 4.6, as all of the runs converge within four simulations. Also note that the run with initial guess 2 converges to a local maximum different from those obtained using the other initial guesses. This illustrates that this gradient-based optimizer is dependent on the choice of the initial guess.

Figure 4.7 presents the five hybrid solutions obtained for a particular producer. From Figures 4.6 and 4.7, we observe that even though four of the runs with different initial guesses converge to solutions with essentially identical NPVs, the actual solutions can

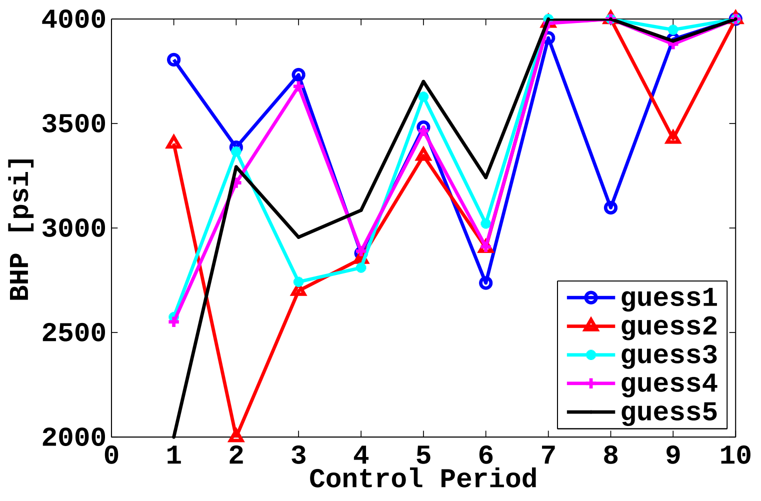


Figure 4.7: Comparison of the five hybrid solutions for one of the production wells

be very different. This indicates that the optimal solutions are not a specific point in the high dimensional solution space, as one might expect. Rather, the solutions represent specific local optima, perhaps arranged along ridges/valleys in the solution space. This is of potential interest as it suggests that, because multiple solutions with similar NPVs exist, the user can select the solution that is easiest to implement or that best satisfies some other criteria.

4.1.2 Case 2: Two-dimensional Heterogeneous Model

Optimization Problem Description

The reservoir model used in this case is a 40×40 2D model with four injectors and four producers. The wells and x-direction permeability are shown in Figure 4.8. This model also involves two-phase oil-water flow, with the same relative permeability curves presented in Figure 4.2. Most of the reservoir simulation parameters are the same as those used in Case 1. The parameters that differ for this case are summarized in Table 4.3. The reservoir simulation time was 3000 days, with the well BHPs updated every 300 days (10 control periods). The total number of optimization variables in this problem is 80 (8 wells $\times$ 10 control periods).

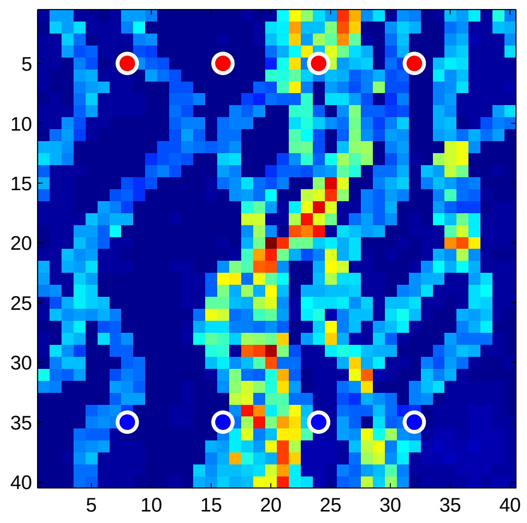


Figure 4.8: Two-dimensional reservoir model (x-permeability field shown) with four production (red) and four injection (blue) wells arranged in a line drive pattern

Table 4.3: Case 2 simulation and optimization parameters

constant ϕ	0.3
k_x/k_y	1.0
r_o	\$80/STB
c_{wp}	\$36/STB
c_{wi}	\$18/STB
Injector BHP range	6000 - 9000 psi
Producer BHP range	2500 - 4500 psi

Production Optimization Results

The performance of the various optimization algorithms is compared in Figure 4.9, which presents the evolution of the NPV as a function of the number of simulations. The results presented in Figure 4.9 are summarized in Table 4.4.

The same initial guess was used for the SQP+adjoint, SQP+FD, GPS and HJDS methods. This initial guess (base case) was constant BHP control of 7500 psi for all injectors and 3000 psi for all producers. Figure 4.9 again illustrates the efficiency of the adjoint-based technique over the other methods considered. It is interesting to note

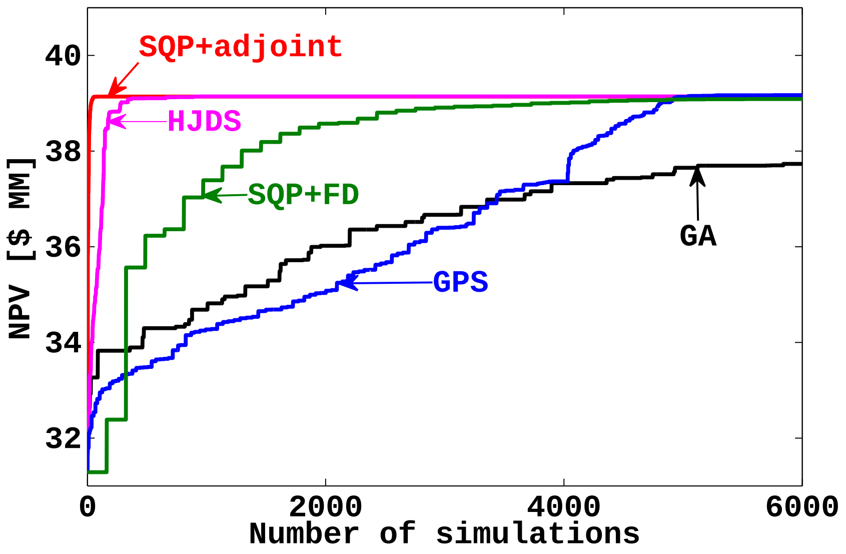


Figure 4.9: Comparison of the performance of the gradient-based and derivative-free algorithms (Case 2)

Table 4.4: Summary of performance of different optimizers for Case 2

Optimization algorithm	Number of simulations	Max. NPV [$ MM]
SQP+adjoint	142	39.14
HJDS	891	39.14
SQP+FD	5841	39.09
GPS	5041	39.14
GA	6000	37.73

that the derivative-free HJDS method is significantly more efficient than the gradient-based SQP with numerical finite difference gradient computation. This suggests that, in the absence of an adjoint-based implementation, HJDS may be preferable over a numerical gradient computation technique since it is not only more efficient, but also may be more robust in avoiding local optima (refer to Case 1 results).

The general performance of the algorithms is consistent with that observed for Case 1. The GPS and SQP+FD algorithms converge to essentially the same NPV as the SQP+adjoint and HJDS algorithms, but they require significantly more simulations. After 6000 simulations, the GA, with a population size of 200, finds a solution with a lower NPV than any of the other methods. As discussed in Section 2.4, the computational efficiency of GA and GPS can be significantly improved with the use of enhancement strategies. These results are presented later.

In this case, the optimal BHP profiles (the optimization results) for all algorithms except the GA are very similar. The resulting optimal BHP profiles for some of the wells are displayed in Figure 4.10. In this figure, in cases where multiple lines are not apparent, results for SQP+adjoint, SQP+FD, HJDS and GPS coincide. The figure shows that the GA results are significantly different from the results of the other methods. Also, even though the results from the other methods are similar for the most part, there are some slight differences, e.g., in INJ2 and PROD3.

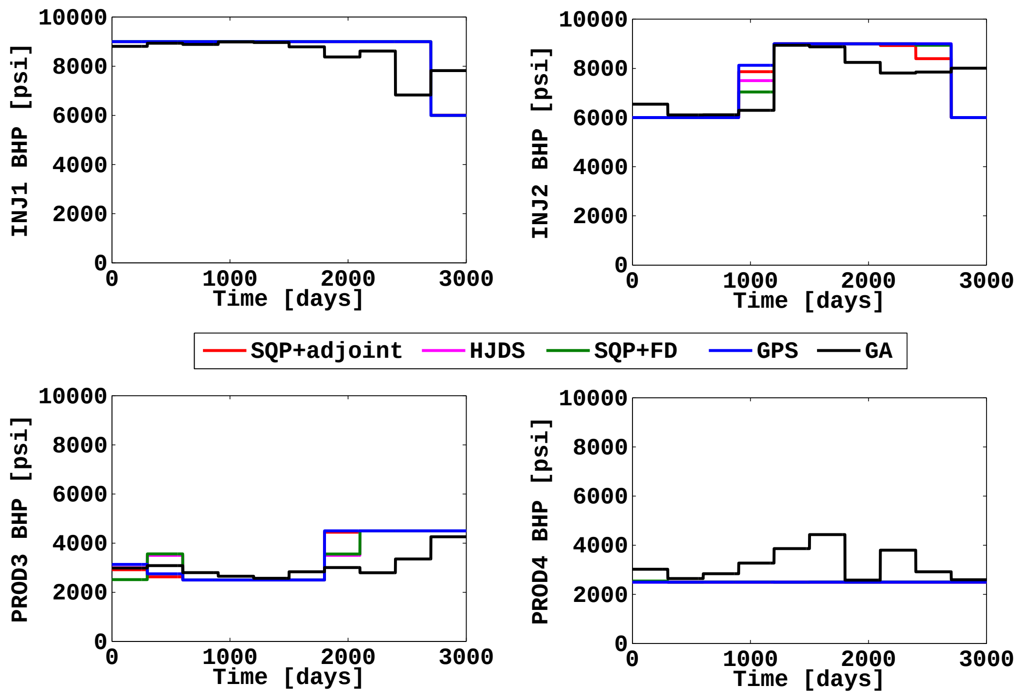
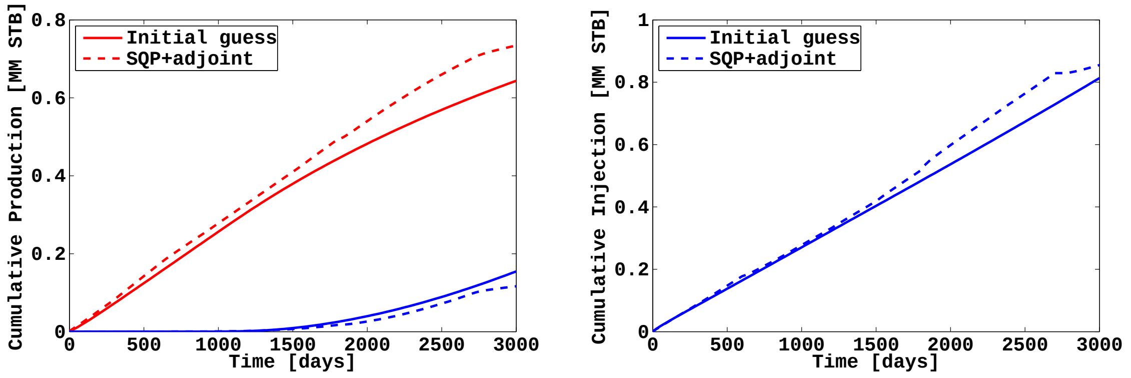


Figure 4.10: Comparison of the resulting BHP profiles for two injectors and two producers for all optimization methods considered

Figure 4.11 presents plots comparing the cumulative injection and production profiles of the base case and the SQP+adjoint optimized case. From this figure, it can be seen that the increase in NPV provided by the optimized solutions over the base case is due mainly to the 14% increase in cumulative oil produced and a 20% decrease in cumulative water produced, even with a corresponding 5% increase in water injected. Thus the "strategy" determined by the optimization algorithm is quite different here than it was for Case 1.



(a) Cumulative oil (red) and water (blue) production

(b) Cumulative water injection

Figure 4.11: Comparison of Case 2 cumulative production and injection profiles of the base case and SQP+adjoint solution

Results with Computational Enhancements for GA and GPS

As discussed in Section 2.4, the computational efficiency of the GA and GPS methods can be improved using computational enhancements such as surrogates (or proxies) and distributed computing. Results for the GA with an artificial neural network (ANN) surrogate are presented in Figure 4.12 and Table 4.5.

Table 4.5: Summary of performance of GA with ANN surrogate (Case 2)

% of population simulated	Number of simulations	Max. NPV [$ MM]
100%	6200	37.53
50%	3500	37.68
20%	1880	37.27
10%	1340	37.56
5%	1070	36.81

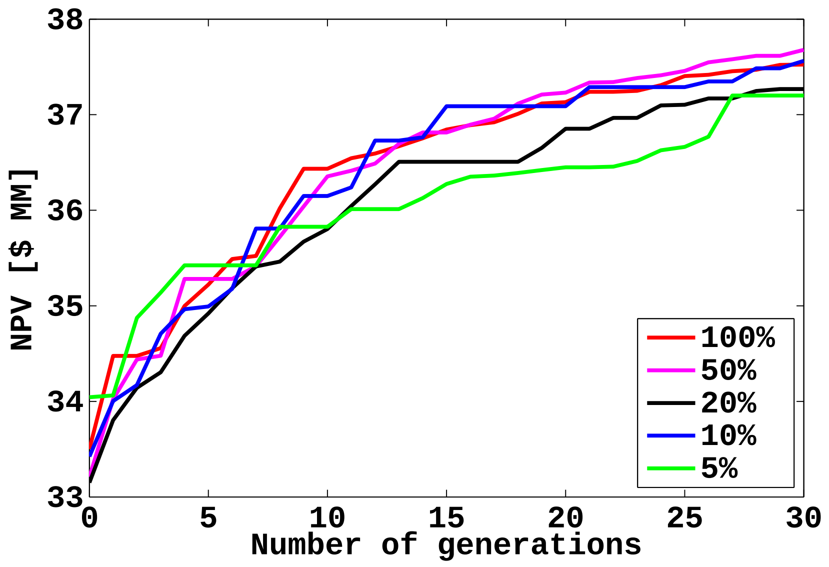


Figure 4.12: Results using GA with ANN surrogate (Case 2)

For these runs, the entire GA population was simulated for the first three generations and all individuals and fitness values were saved. These saved individuals were then used to train an ANN. This ANN was retrained using the individuals simulated at each subsequent generation. The values in the first column of Table 4.5 are the percentage of the GA population whose fitness (objective function value) was obtained using the reservoir simulator, after the first three generations. The fitness of the rest of the population was evaluated by the trained ANN. Figure 4.12 and Table 4.5 show that significant savings in computational effort can be obtained through use of the ANN surrogate, while still providing reasonable solutions to the production optimization problem. We reiterate, however, that the GA underperforms the other optimization techniques for this problem.

Results using GPS with the DACE kriging surrogate are presented in Figure 4.13, together with the basic GPS, HJDS and SQP+adjoint results for comparison purposes. From the figure, the benefit of using the DACE surrogate in the GPS search step is evident. Recall from Section 2.2.2 that the GPS algorithm is divided into search and poll steps. Also recall that in a particular iteration, if the search step is successful in identifying a point that improves the objective function, the poll step is not performed. In Figure 4.13, the rapid initial increase in NPV is due to the success of the surrogate-based search steps in the first few iterations of the optimization. With this enhancement of the GPS algorithm, its performance approaches that of HJDS for this case.

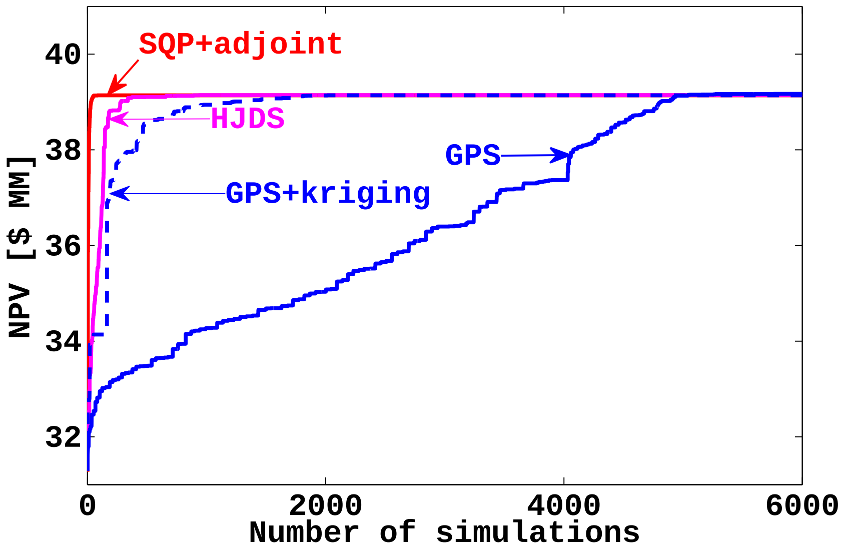


Figure 4.13: Comparison of GPS with DACE kriging surrogate with other algorithms (Case 2)

Results obtained when the function evaluations of the GPS, GA and SQP+FD algorithms are distributed over 40 processors are presented in Figure 4.14 and summarized in Table 4.6. Equivalent simulations here are defined as:

$$\text{Equivalent simulations} = \frac{\text{Number of actual simulations}}{\text{Number of processors used}}. \tag{4.2}$$

In the SQP+FD implementation, this formula was used to calculate the equivalent simulations during the gradient evaluation using finite differences. The simulations performed during the line search were counted as one equivalent simulation because the line search is a serial operation. The results in Figure 4.14 and Table 4.6, compared with those in Figure 4.9 and Table 4.4, clearly display the benefit of distributed computing. The computational savings are obtained by distributing the gradient computation in SQP+FD, the fitness evaluation of the GA population and the evaluation of the poll points in an iteration of GPS. Greater computational savings can be achieved if more processors are available. We reiterate that each simulation runs on only one processor; i.e., we are not running a parallel reservoir simulation code.

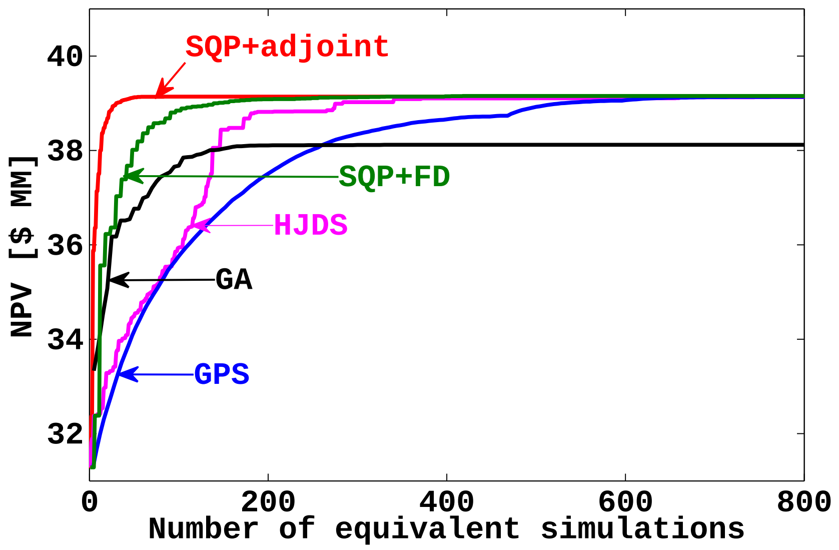


Figure 4.14: Comparison of the Case 2 results with distributed computing

Table 4.6: Summary of performance of different optimizers for Case 2, with distributed computing

Optimization algorithm	Number of equivalent simulations	Max. NPV [$ MM]
SQP+adjoint	142	39.14
SQP+FD	326	39.14
HJDS	891	39.14
GPS	772	39.14
GA	305	38.12

4.2 Optimization Cases Including Nonlinear Constraints

We now consider fully constrained production optimization problems, which include both bound and nonlinear inequality constraints. The problems involve maximizing the undiscounted NPV by optimizing the well BHPs, while honoring the bounds on injector and producer BHPs, as well as field constraints. The latter include a maximum field injection constraint, a minimum field oil production constraint, a maximum field liquid production constraint, and a maximum water cut constraint on all production wells.

4.2.1 Case 3: Two-dimensional Heterogeneous Model

Optimization Problem Description

The reservoir model used in this case is the same 40×40 2D model used in Case 2, except here there are two injectors and two producers, as shown in Figure 4.15, and the cell dimensions are different. The different reservoir simulation and production optimization parameters for this case are summarized in Table 4.7. The reservoir simulation time was 3650 days, which was divided into five control periods of 730 days each. The total number of optimization variables in this problem is 20.

Table 4.7: Case 3 simulation and optimization parameters

Grid cell dimensions	$100 \times 100 \times 20$ ft^3
r_o	\$50/STB
c_{wp}	\$10/STB
c_{wi}	\$5/STB
Injector BHP range	6000 - 10000 psi
Producer BHP range	500 - 4500 psi
Max. field water injection rate	1000 STB/day
Min. field oil production rate	450 STB/day
Max. field liquid production rate	1500 STB/day
Max. water cut in any production well	0.5

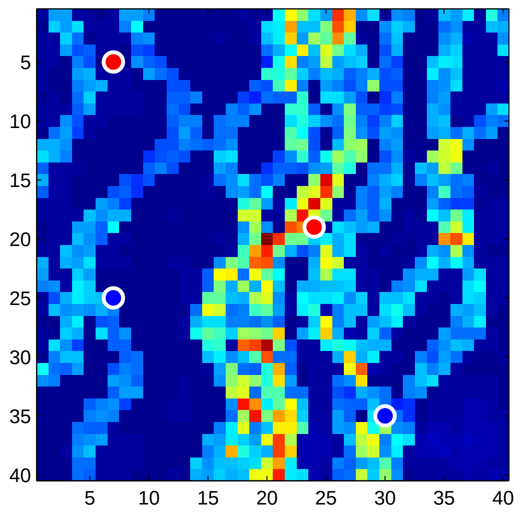


Figure 4.15: Two-dimensional reservoir model with two production (red) and two injection (blue) wells (x-direction permeability displayed)

Production Optimization Results

The problem was first solved with the SQP+adjoint method (most efficient bound constrained method) without considering the nonlinear inequality constraints specified in Table 4.7. The initial guess (base case) was constant BHP of 8000 psi for all injectors and 2500 psi for all producers. The result of this optimization was a bound-constrained maximum NPV of $95.89 million, obtained after 21 simulations. Figure 4.16 presents the field water injection rate and water cut for one of the producers. The maximum constraints for each (indicated by the dotted red lines) were not enforced during the optimization. Since these constraints are violated, it is clear that the nonlinear constraints need to be incorporated into the optimization.

The problem is now solved using the nonlinear constraint handling techniques presented in Chapter 3. All the methods, except the GA and hybrid methods, use the same initial guess as that used for the SQP+adjoint bound-constrained solution. First, the results obtained with the three derivative-free methods using the traditional penalty function constraint handling are presented. Since the correct value for the penalty parameter (ρ) was not known a priori, a tuning process was performed where

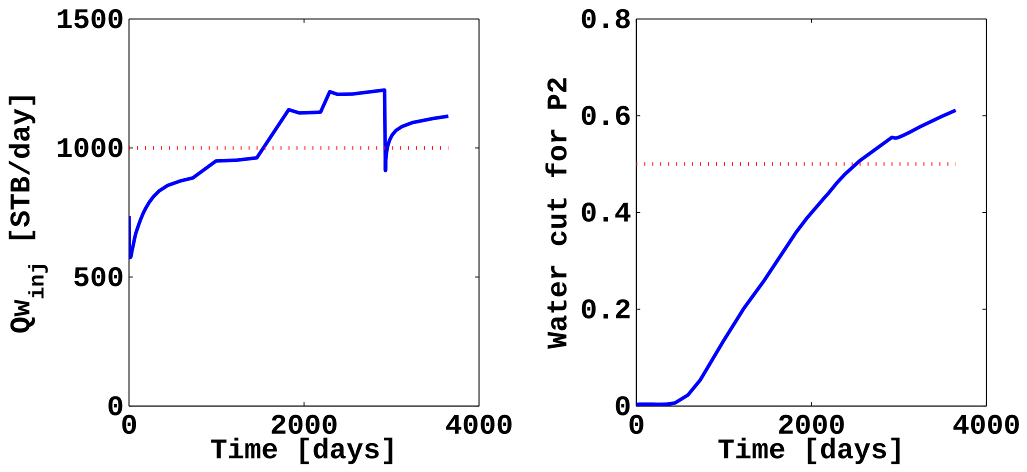


Figure 4.16: SQP+adjoint results excluding nonlinear constraint enforcement, showing constraint violation

the optimization methods were run with different values for ρ, and an "optimal" ρ value was obtained. The results of this tuning process are presented in Figure 4.17.

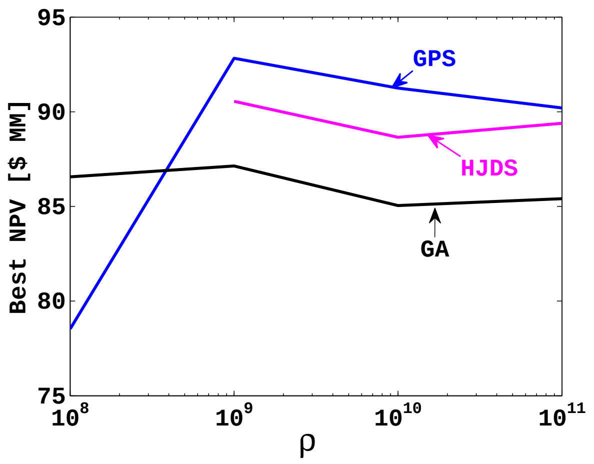


Figure 4.17: Results of tuning penalty parameter (Case 3)

This figure presents the maximum NPV obtained by each optimization algorithm as a function of ρ. Since the GA is stochastic in nature, the results presented in Figure 4.17 are average results from several GA runs that produced solutions. From the tuning results, it can be seen that the final solutions are sensitive to ρ, and the optimal ρ is about 10^9. No results are available for HJDS with a ρ of 10^8 because for this value of ρ, no feasible point was found during the HJDS optimization process, meaning that the HJDS algorithm failed to locate a point in the solution space that satisfied all of the prescribed nonlinear inequality constraints. Similar issues were observed when running the GA, i.e., during some of the GA runs, no feasible solution was found. The results presented in Figure 4.17 are averages of runs that did produce feasible solutions. These observations highlight the robustness issues associated with the traditional penalty function methods, mentioned in Section 3.2.1.

Figure 4.18 presents results using the GA with the more robust parameterless penalty function and the traditional penalty function. These plots present the fraction of the GA population that is feasible as the optimization progresses. Note that the parameterless penalty function GA has a high feasible fraction after about ten generations, implying that it successfully identifies the feasible region and then moves most of the individuals into it to search for the constrained optimum. The traditional penalty function GA only has a few generations where some individuals are feasible. Some of these traditional penalty GA runs fail because during the course of the optimization, none of the individuals are ever moved into the feasible region because of a poor choice for ρ. From Figure 4.18 it can be seen that with tuning, the traditional penalty function GA produces a better solution than the parameterless penalty function GA. On the other hand the parameterless penalty function GA did not require any tuning and, given a large enough population, always generates feasible solutions.

Figure 4.19 presents results for the more robust GPS with filter constraint handling and GPS with the traditional penalty function. Equivalent simulations are used because the runs were distributed over 40 processors. The plots in Figure 4.19 start after a few equivalent simulations because the initial guess of constant BHPs is not feasible, and it takes the algorithm a few iterations to locate a feasible solution. Again, note that even though better results are obtained with the traditional penalty function approach, tuning is required. On the other hand, the GPS with filter avoids tuning and gets to within 3% of the solution found by the tuned penalty approach. This suggests that the GPS with filter may be useful for practical applications.

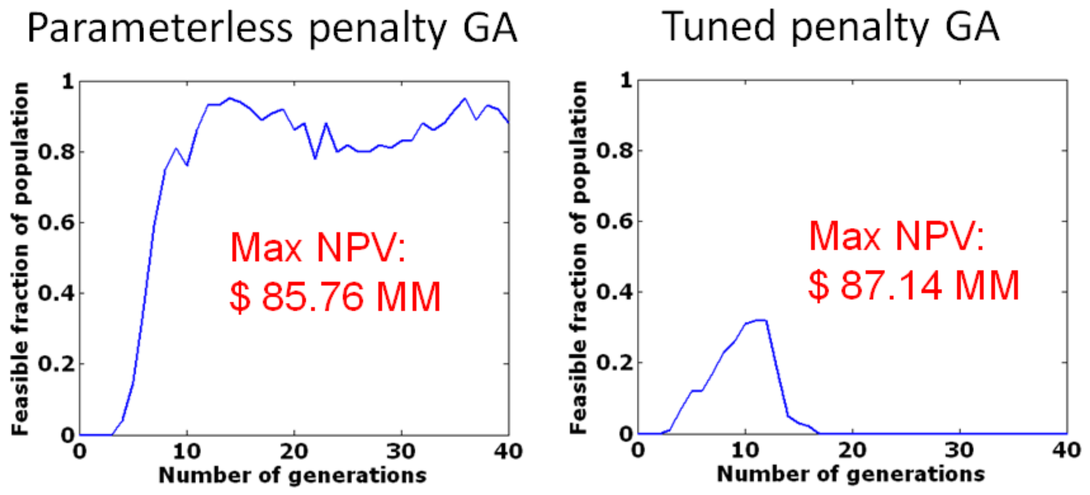


Figure 4.18: Comparison of results for GA with parameterless and traditional penalty functions (Case 3)

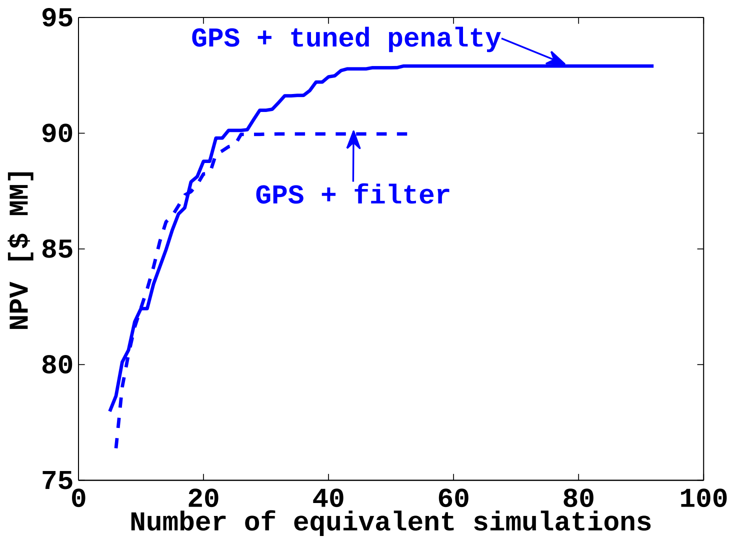


Figure 4.19: Comparison of results for GPS with filter and GPS with traditional penalty function (Case 3)

Results for the hybridization strategy, with parameterless penalty GA and HJDS with traditional penalty function, as discussed in Section 3.4, are presented in Figure 4.20. The penalty parameter used in the traditional penalty function is chosen to be an order of magnitude higher than the objective function value of the GA solution. It is apparent from Figure 4.20 that the hybrid method outperforms the standalone HJDS in terms of both efficiency and the quality of the final solution. This validates the hybridization idea of starting with a robust global search algorithm for initial exploration of the solution space and then using a more efficient local search algorithm to quickly converge to an optimal solution.

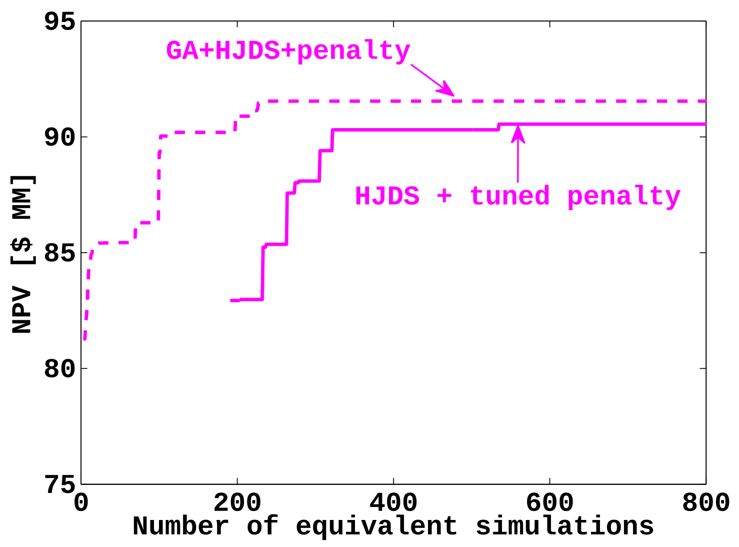


Figure 4.20: Comparison of results for parameterless penalty GA hybridized with HJDS with traditional penalty function and HJDS with traditional penalty function alone

Of all the methods considered, one of the best solutions was obtained using the SQP algorithm with numerical FD used to compute gradients of both the objective function and constraints with respect to controls. The objective function and constraint violation gradients for SQP+FD were computed in a distributed manner over 40 processors. The SQP+FD solution had a maximum NPV of $92.88 million obtained after 538 equivalent simulations. This value is less than that obtained by SQP+adjoint without considering the constraints. This is intuitive because one would expect the problem to be more restrictive with the nonlinear constraints specified. Figure 4.21 presents the field water injection rate and water cut for one of the producers, together with specified maximum constraints for each (indicated by the dotted red lines). It is clear from the figure that these nonlinear constraints are indeed satisfied.

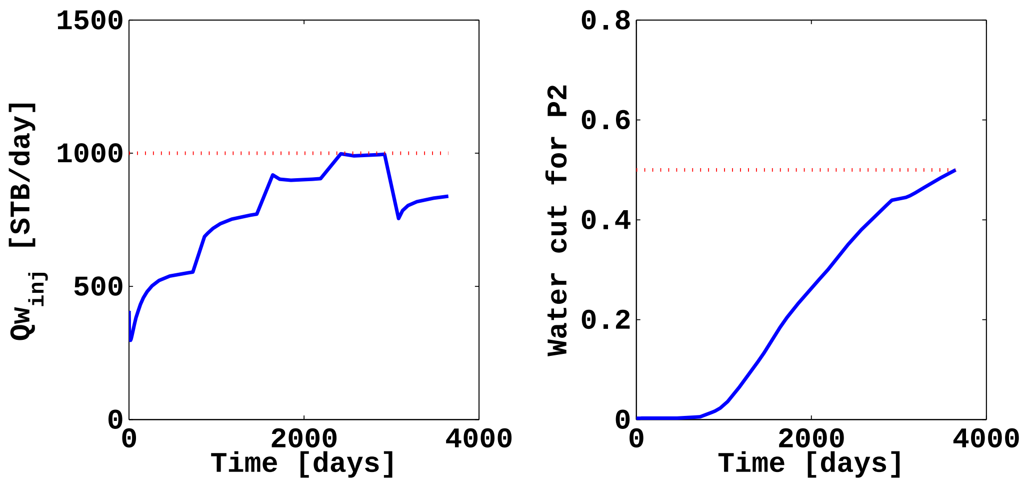


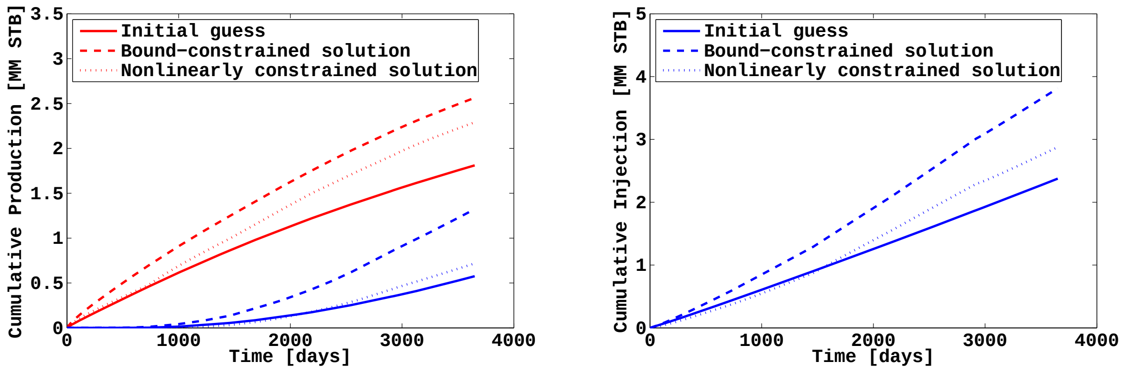
Figure 4.21: SQP+FD results showing constraint satisfaction (Case 3)

The solutions from the other methods with nonlinear constraint handling are qualitatively similar to those in Figure 4.21 as they too satisfy the specified constraints. Table 4.8 presents a summary of the performance of all the methods with nonlinear constraint handling considered. In this table, ΔJ^* designates the percentage difference between the best NPV found for all the methods ($92.90 million, the NPV found by the GPS with tuned penalty) and that found by the particular method.

Table 4.8: Case 3 results summary

Methods considered	ΔJ^*(%)	# of simulations	# of equivalent simulations
GPS+tuned penalty	0.00	1534	52
SQP+FD	0.02	8059	538
Parameterless GA+GPS	0.22	5891	166
Parameterless GA+HJDS	1.39	4334	239
HJDS+tuned penalty	2.46	535	535
GPS+filter	3.06	1236	33
GA+tuned penalty	6.13	1720	43
Parameterless GA	7.97	4200	105

It is evident that, although the best result for this case is achieved with GPS with a tuned penalty parameter, the results from several other methods are promising because they do not require any tuning. This could be especially useful for complex problems that require very expensive simulations. Of note are the SQP, GPS with filter and hybrid methods. The SQP+FD method achieves a solution within 0.02% of the best solution, but requires a large number of simulations (even when run on multiple processors). This demonstrates the applicability of an adjoint formulation for obtaining the objective function and individual (or lumped) constraint gradients. The GPS with filter method could be favored in practice because of its efficiency in obtaining a reasonable solution. The hybrid methods are also recommended because they are efficient and achieve close to optimal results when run either in a serial (parameterless GA+HJDS) or fully parallel (parameterless GA+GPS) fashion.



(a) Cumulative oil (red) and water (blue) production

(b) Cumulative water injection

Figure 4.22: Comparison of Case 3 cumulative production and injection profiles of the base case, bound-constrained solution for SQP+adjoint, and the best nonlinearly constrained solution from GPS+tuned penalty.

Figure 4.22 presents the cumulative production and injection profiles for the base case, bound-constrained solution from SQP+adjoint and the best solution accounting for nonlinear constraints from GPS+tuned penalty. From this figure it can be seen that the improvement in NPV over the base case for the bound-constrained solution comes from injecting significantly more water to produce significantly more oil. The specification of maximum field water injection and maximum production well water cut constraints (i.e., nonlinear constraints) leads to solutions with more efficient use of water and hence less injected and produced water than the bound-constrained solution. The specification of a minimum field oil rate ensures that the produced oil is still sufficient. This, together with the more efficient use of water, leads to an increase in NPV over the base case and an NPV that is quite close to the bound-constrained case. The specific NPVs are \$72.90 million for the base case, \$95.89 million for the bound-constrained case and \$92.90 million for the nonlinearly constrained case.

4.2.2 Case 4: Section of SPE 10 Model

Optimization Problem Description

The reservoir model used in this case is a $60 \times 60 \times 5$ section from the channelized portion of the 3D SPE 10 comparative solution model. Figure 4.23 shows the porosity of the first layer (layer 76 of the full model) for the section of the SPE 10 model used here, as well as the injector and producer well locations. There are a total of 25 wells in this model.

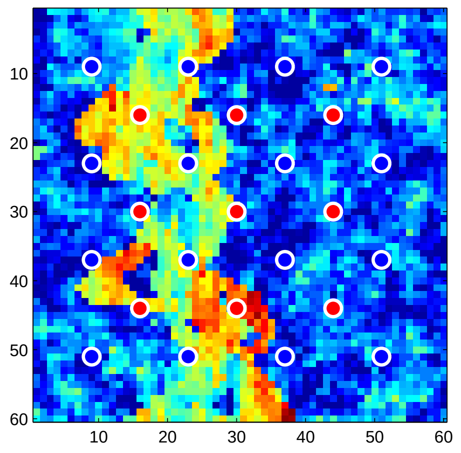


Figure 4.23: Section of SPE 10 reservoir model, showing porosity of top layer. Model contains 16 injection (blue) and 9 production (red) wells

The simulation model involves two-phase oil-water flow. Figure 4.24 presents the relative permeability curves for oil and water used in the simulation model. The remaining reservoir simulation and production optimization parameters for this case are summarized in Table 4.9. The reservoir simulation time was 1460 days, which was divided into four control periods of 365 days each. This production optimization problem therefore has 100 control variables.

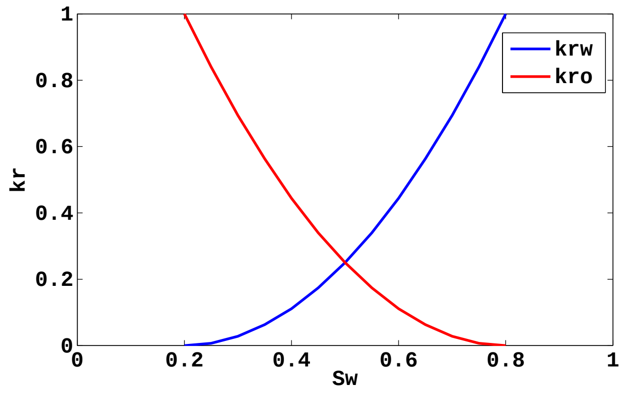


Figure 4.24: Relative permeability curves for the oil and water phases (Case 4)

Table 4.9: Case 4 simulation and optimization parameters

Grid cell dimensions	$500 \times 500 \times 2$ ft^3
Initial pressure, p_i	6000 psi
c_R at ref. pressure	5×10^{-6} psi^{-1}
μ_o at p_i	3.0 cp
μ_w at p_i	0.3 cp
ρ_o	53 lbm/ft^3
ρ_w	64 lbm/ft^3
B_o and B_w at p_i	1.01 RB/STB
r_o	\$50/STB
c_{wp}	\$10/STB
c_{wi}	\$5/STB
Injector BHP range	6500 - 12000 psi
Producer BHP range	500 - 5500 psi
Max. field water injection rate	15000 STB/day
Min. field oil production rate	3000 STB/day
Max. field liquid production rate	10000 STB/day
Max. water cut in any production well	0.7

Production Optimization Results

The problem was first solved with the SQP+adjoint method without considering the nonlinear inequality constraints specified in Table 4.9. The initial guess (base case) was constant BHP of 9250 psi for all the injectors and 3000 psi for all the producers. The SQP+adjoint solution provided a bound-constrained maximum NPV of \$270.0 million obtained after 211 simulations. Figure 4.25 presents one of the constraints, the field liquid production rate, for this bound-constrained solution. Note the clear constraint violation.

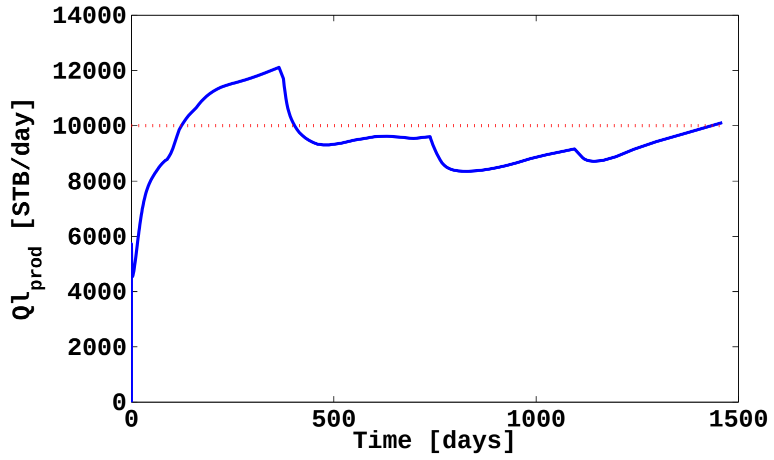


Figure 4.25: SQP+adjoint result for total liquid production rate, excluding nonlinear constraints (Case 4)

Solution of the fully constrained problem was first attempted using the SQP algorithm with numerical FD gradients. The initial guess was the same used to obtain the bound-constrained SQP+adjoint solution presented above. After 11000 simulations, this SQP+FD algorithm could not find a feasible solution and stalled at an infeasible local optimum. This probably occurred because we have a very complex problem in a high dimensional search space.

The problem was then approached using the hybrid method combining the parameterless penalty function GA with the HJDS with traditional penalty. For the GA part of the run, the fraction of the population that is feasible as a function of the number of generations is presented in Figure 4.26. Note from this figure that the robust parameterless penalty GA successfully identifies a feasible region of the solution space. The second phase of the hybrid method is started using the best solution found by the GA and it acts to provide further improvement of the GA solution. This second phase entails using the HJDS algorithm with the (untuned) penalty constraint handling. As in Case 3, the penalty parameter used for this phase is an order of magnitude higher than the objective function value of the GA solution.

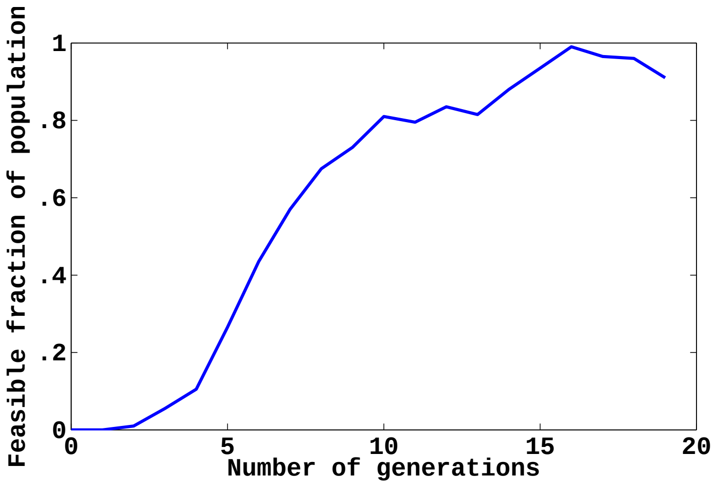


Figure 4.26: Plot of feasible fraction of GA showing that the first phase of hybrid procedure successfully identifies the feasible region (Case 4)

Figure 4.27 compares the hybrid result with that from the SQP+adjoint without nonlinear constraint handling. These results are very interesting because they show that the bound-constrained solution obtained with a local gradient-based optimizer can be worse than the solution obtained for a fully constrained problem optimized using an initial exploration followed by local exploitation. This result clearly demonstrates the potential impact of this two step formulation in tackling complex nonlinear production optimization problems. We reiterate that the solution from the hybrid method satisfies all the specified constraints. Figure 4.28 illustrates satisfaction of the total liquid production constraint.

Figure 4.29 displays cumulative production and injection results for this case. From this figure, the disparity in the cumulative injection and production profiles of the

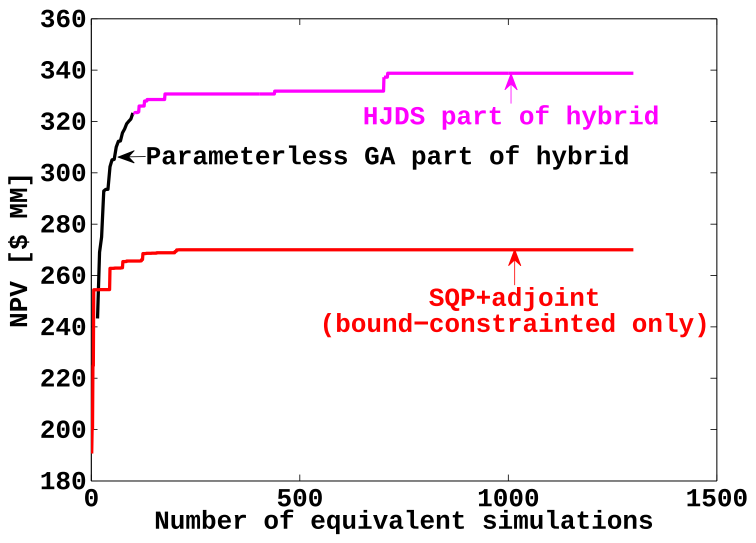


Figure 4.27: Nonlinearly constrained hybrid results compared with SQP+adjoint bound-constrained results (Case 4)

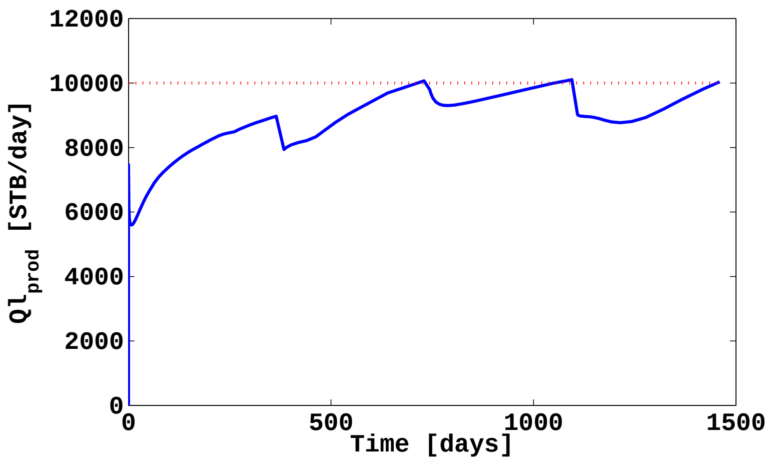
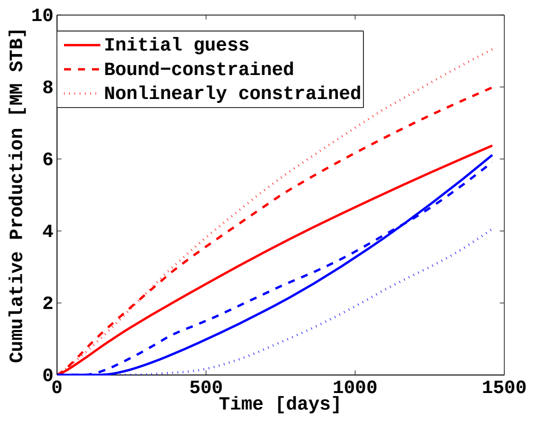
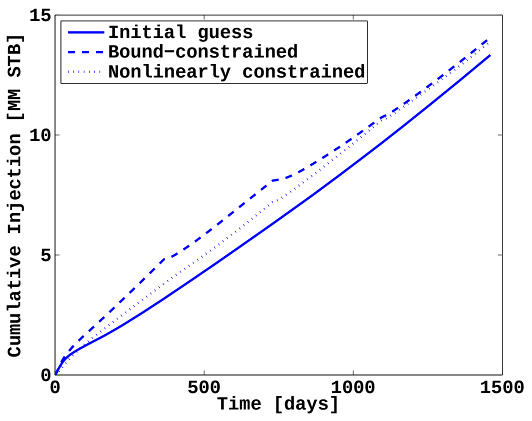


Figure 4.28: Plot of total liquid production rate of hybrid solution showing constraint satisfaction (Case 4)

three cases is apparent. Of special interest is the much better solution obtained by the more robust hybrid method including nonlinear constraints when compared to that obtained by the local gradient-based optimizer without nonlinear constraints. This is due to the fact that the initial guess used in the local optimizer turned out to be quite poor. This can occur in practice as it may be difficult to identify a feasible and high-quality initial guess. For this reason, the use of a global search algorithm such as the GA to obtain one or more reasonable starting points for a local optimizer is a sound idea.



(a) Cumulative oil (red) and water (blue) production



(b) Cumulative water injection

Figure 4.29: Comparison of Case 4 cumulative production and injection profiles of the base case, bound-constrained solution for SQP+adjoint, and the hybrid nonlinearly constrained solution

Chapter 5

Conclusions and Future Work

5.1 Conclusions

This work involved a comparative study of several optimization methods with the goal of identifying the most capable methods for use in solving generally constrained production optimization problems. The methods evaluated include the gradient-based sequential quadratic programming (SQP) method and derivative-free genetic algorithm (GA), general pattern search (GPS) and Hooke-Jeeves direct search (HJDS) procedures. These methods were applied to the solution of production optimization problems with only bound constraints and to the solution of problems involving both bound and nonlinear constraints.

From the optimization results for the bound-constrained problems, it was shown that, as expected, the use of adjoint procedures provided highly efficient gradient-based optimization. The HJDS method was found to be the most efficient of the derivative-free methods when these methods were run serially. The GA and GPS methods displayed slower convergence properties and required many simulations to attain an optimal solution. In order to improve the efficiency of these procedures, several computational enhancement strategies were investigated. It was shown that the use of surrogates and distributed computing significantly increased the efficiency of the methods. A hybrid implementation combining the global search nature of the GA with the efficiency of the SQP with adjoint gradient computation was shown to perform well. This approach is able to provide multiple solutions to the problem, though many were observed to have essentially the same objective function value.

In practice, the solution of production optimization problems is usually subject to physical and economic constraints. These constraints can be nonlinearly related to the optimization variables. Different constraint handling techniques for such problems were investigated, including the SQP approach for constraint handling, penalty function approach, filter method, and a hybrid methodology. Traditional penalty function approaches may require extensive tuning and lack robustness in some cases. From the results for two nonlinearly constrained production optimization cases, it was shown that the SQP, GPS with filter and hybrid methods combining GA with a robust penalty function treatment and an efficient local search method appear to hold the most promise for practical applications.

5.2 Future Work

The following areas are suggested for future investigations:

- Further evaluation of the hybridization strategies to determine clear guidelines for switching from one phase to another.
- Use of reduced order models as surrogates to further improve the computational efficiency of the methods considered in this study. Reduced order models are based on the physics and numerics of the problem so they can be expected to be more robust than artificial neural networks or kriging surrogates.
- Use of the methods considered here in a multi-objective framework to optimize multiple objectives and provide a Pareto front. This could be very useful for practical business decisions.
- Use of some of the procedures presented here to address the coupled well placement and production optimization problems. This would enable very general optimization of oil field problems.

Nomenclature

Abbreviations

ANN	artificial neural network
BHP	well bottom-hole pressure
DACE	design and analysis of computer experiments
FD	finite difference
GA	genetic algorithm
GPS	general pattern search
HJDS	Hooke-Jeeves direct search
KKT	Karush-Kuhn-Tucker
MM	million
NPV	net present value
SQP	sequential quadratic programming

Variables

A	matrix for specification of bound and linear constraints
B	formation volume factor
c	normalized nonlinear inequality constraint or cost
F	modified objective function
h	constraint violation function
J	production optimization objective function
k	permeability
L	Lagrangian
m	number of nonlinear inequality constraints
p	pressure

Q	cumulative injection or production
r	price
S	surrogate approximation to objective function
$\mathbf{d}$	search direction
$\mathbf{g}$	reservoir simulation equations
$\mathbf{u}$	vector of control variables
$\mathbf{x}$	vector of dynamic state variables

Superscripts

$*$	optimal point
0	initial point or condition
F	feasible
I	infeasible
k	iteration index
N	total number of time steps
n	time step or dimension of optimization parameter space
T	transpose

Subscripts

B	bound
i	index
k	iteration index
o	oil
w	water
wi	injected water
wp	produced water

Greek Symbols

α	step size in search direction
Δ	poll step size
λ	Lagrange multiplier
μ	viscosity
ϕ	porosity
ρ	penalty parameter or density

Bibliography

[1] M. A. Abramson. *NOMADm version 4.6 User's Guide*. Department of Mathematics and Statistics, Air Force Institute of Technology, 2007.

[2] I. Aitokhuehi. *Real-Time Optimization of Smart Wells*. Master's thesis, Department of Petroleum Engineering, Stanford University, 2004.

[3] L. F. Almeida, Y. J. Tupac, J. G. Lazo Lazo, M. A. Pacheco, and M. M. B. R. Vellasco. Evolutionary optimization of smart-wells control under technical uncertainties. Paper SPE 107872 presented at the 2007 Latin American & Caribbean Petroleum Engineering Conference, Buenos Aires, Argentina, 15-18 April.

[4] H. Asheim. Maximization of water sweep efficiency by controlling production and injection rates. Paper SPE 18365 presented at the 1988 European Petroleum Conference, London, United Kingdom, 16-19 October.

[5] C. Audet and J. E. Dennis Jr. Analysis of generalized pattern searches. *SIAM Journal on Optimization*, 13(3):889–903, 2002.

[6] C. Audet and J. E. Dennis Jr. A pattern search filter method for nonlinear programming without derivatives. *SIAM Journal on Optimization*, 14(4):980–1010, 2004.

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

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

[9] D. R. Brouwer and J. D. Jansen. Dynamic optimization of waterflooding with smart wells using optimal control theory. *SPE Journal*, 9(4):391–402, 2004.

[10] D. R. Brouwer, G. Naevdal, J. D. Jansen, E. H. Vefring, and C. P. J. W. van Kruijsdiik. Improved reservoir management through optimal control and continuous model updating. Paper SPE 90149 presented at the 2004 SPE Annual Technical Conference and Exhibition, Houston, Texas, 26-29 September.

[11] H. Cao. *Development of Techniques for General Purpose Simulators*. PhD thesis, Department of Petroleum Engineering, Stanford University, 2002.

[12] J. A. Carroll III. *Multivariate Production Systems Optimization*. Master's thesis, Department of Petroleum Engineering, Stanford University, 1990.

[13] S. A. Castro. *A Probabilistic Approach to Jointly Integrate 3D/4D Seismic Production Data and Geological Information for Building Reservoir Models*. PhD thesis, Department of Energy Resources Engineering, Stanford University, 2007.

[14] M. M. Chaudhri, H. A. Phale, N. Liu, and D. S. Oliver. An improved approach for ensemble-based production optimization. Paper SPE 121305 presented at the 2009 SPE Western Regional Meeting, San Jose, California, 24-26 March.

[15] A. R. Conn, K. Scheinberg, and L. N. Vicente. *Introduction to Derivative-Free Optimization*. MPS-SIAM Series on Optimization. MPS-SIAM, 2009.

[16] A. S. Cullick, D. Heath, K. Narayanan, J. April, and J. Kelly. Optimizing multiple-field scheduling and production strategy with reduced risk. Paper SPE 84239 presented at the 2009 SPE Annual Technical Conference and Exhibition, Denver, Colorado, 5-8 October.

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

[18] H. Demuth, M. Beale, and M. Hagan. *Neural Network Toolbox^{TM} 6: User's Guide*. The Mathworks, Inc., 2009.

[19] ExxonMobil. The outlook for energy: A view to 2030. Technical report, ExxonMobil Corporation, 2008.

[20] R. Fletcher, S. Leyffer, and P. Toint. A brief history of filter methods. Technical report, Mathematics and Computer Science Division, Argonne National Laboratory, 2006.

[21] B. Guyaguler. *Optimization of Well Placement and Assessment of Uncertainty.* PhD thesis, Department of Petroleum Engineering, Stanford University, 2002.

[22] T. J. Harding, N. J. Radcliffe, and P. R. King. Optimization of production strategies using stochastic search methods. Paper SPE 35518 presented at the 1996 European 3-D Reservoir Modeling Conference, Stavanger, Norway, 16-17 April.

[23] R. Hooke and T. A. Jeeves. Direct search solution of numerical and statistical problems. *Journal of the ACM*, 8(2):212–229, 1961.

[24] J. D. Jansen, D. R. Brouwer, G. Naevdal, and C. P. J. W. van Kruijsdiik. Closed-loop reservoir management. *First Break*, 23:43–48, 2005.

[25] J. D. Jansen, S. D. Douma, D. R. Brouwer, P. M. J. Van den Hof, O. H. Bosgra, and A. W. Heemink. Closed loop reservoir management. Paper SPE 119098 presented at the 2009 SPE Reservoir Simulation Symposium, The Woodlands, Texas, 2-4 February.

[26] Y. Jiang. *Techniques for Modeling Complex Reservoirs and Advanced Wells.* PhD thesis, Department of Energy Resources Engineering, Stanford University, 2007.

[27] C. T. Kelley. *Iterative Methods for Optimization.* Frontiers in Applied Mathematics. SIAM, 1999.

[28] T. G. Kolda, R. M. Lewis, and V. Torczon. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Review*, 45(3):385–482, 2003.

[29] D. Kumar. *Optimization of Well Settings to Maximize Residually Trapped CO_2 in Geologic Carbon Sequestration.* Master's thesis, Department of Energy Resources Engineering, Stanford University, 2007.

[30] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright. Convergence properties of the Nelder-Mead simplex method in low dimensions. *SIAM Journal on Optimization*, 9(1):112–147, 1998.

[31] M. L. Litvak, L. A. Hutchins, R. C. Skinner, B. L. Darlow, R. C. Wood, and L. J. Kuest. Prudhoe Bay e-field production optimization system based on integrated reservoir and facility simulation. Paper SPE 77643 presented at the 2002 SPE Annual Technical Conference and Exhibition, San Antonio, Texas, 29 September-2 October.

[32] W. Liu and W. F. Ramirez. Optimal control of three-dimensional steam flooding process. *Journal of Petroleum Science and Engineering*, 11(2):137–154, 1989.

[33] S. N. Lophaven, H. B. Nielsen, and J. Sondergaard. DACE: A Matlab kriging toolbox, version 2.0. Technical report, Technical University of Denmark, 2002.

[34] A. L. Marsden, J. A. Feinstein, and C. A. Taylor. A computational framework for derivative-free optimization of cardiovascular geometries. *Computational Methods in Applied Mechanics and Engineering*, 197:1890–1905, 2008.

[35] A. L. Marsden, M. Wang, J. E. Dennis Jr., and P. Moin. Trailing-edge noise reduction using derivative-free optimization and large-eddy simulation. *Journal of Fluid Mechanics*, 572:13–36, 2007.

[36] Mathworks. *Genetic Algorithm and Direct Search Toolbox^{TM} 2: User's Guide*. The Mathworks, Inc., 2009.

[37] Mathworks. *Optimization Toolbox^{TM} 6: User's Guide*. The Mathworks, Inc., 2009.

[38] G. Mehos and W. F. Ramirez. Use of optimal control theory to optimize carbon dioxide miscible flooding enhanced oil recovery. *Journal of Petroleum Science and Engineering*, 2(4):247–260, 1989.

[39] J. A. Nelder and R. Mead. A simplex for function minimization. *The Computer Journal*, 7(4):308–313, 1965.

[40] J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer, second edition, 2006.

[41] J. Onwunalu. *Optimization of Nonconventional Well Placement Using Genetic Algorithms and Statistical Proxy.* Master's thesis, Department of Petroleum Engineering, Stanford University, 2006.

[42] A. Osyczka. *Evolutionary Algorithms for Single and Multicriteria Design Optimization*, volume 79 of *Studies in Fuzziness and Soft Computing.* Physica-Verlag, 2006.

[43] W. F. Ramirez. *Application of Optimal Control Theory to Enhanced Oil Recovery.* Elsevier, 1987.

[44] V. Rigot. *New Well Optimization in Mature Fields.* Master's thesis, Department of Petroleum Engineering, Stanford University, 2003.

[45] P. Sarma. *Efficient Closed-loop Optimal Control of Petroleum Reservoirs Under Uncertainty.* PhD thesis, Department of Petroleum Engineering, Stanford University, 2006.

[46] P. Sarma, K. Aziz, and L. J. Durlofsky. Implementation of adjoint solution for optimal control of smart wells. Paper SPE 92864 presented at the 2005 SPE Reservoir Simulation Symposium, Houston, Texas, 31 January - 2 February.

[47] P. Sarma, W. H. Chen, L. J. Durlofsky, and K. Aziz. Production optimization with adjoint models under nonlinear control-state path inequality constraints. *SPE Reservoir Evaluation & Engineering*, 11(2):326–339, 2008.

[48] G. A. Virnovsky. Waterflooding strategy design using optimal control theory. Paper presented at the 6th European IOR Symposium, Stavanger, Norway, 1991.

[49] C. Wang, G. Li, and A. C. Reynolds. Production optimization in closed-loop reservoir management. Paper SPE 109805 presented at the 2007 SPE Annual Technical Conference and Exhibition, Anaheim, California, 11-14 November.

[50] P. Wang. *Development and Applications of Production Optimization Techniques for Petroleum Fields*. PhD thesis, Department of Petroleum Engineering, Stanford University, 2003.

[51] P. Wang, M. Litvak, and K. Aziz. Optimization of production operations in petroleum fields. Paper SPE 77658 presented at the 2002 SPE Annual Technical Conference and Exhibition, San Antonio, Texas, 29 September-2 October.

[52] B. Yeten. *Optimum Deployment of Nonconventional Wells*. PhD thesis, Department of Petroleum Engineering, Stanford University, 2003.

[53] B. Yeten, L. J. Durlofsky, and K. Aziz. Optimization of smart well control. Paper SPE 79031 presented at the 2002 SPE International Thermal Operations and Heavy Oil Symposium and International Horizontal Well Technology Conference, Calgary, Canada, 4-7 November.