RESERVOIR FLOODING OPTIMIZATION BY CONTROL POLYNOMIAL APPROXIMATIONS

A Dissertation

by

NADAV SOREK

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

August 2017

Major Subject: Petroleum Engineering

ABSTRACT

In this dissertation, we provide novel parametrization procedures for water-flooding production optimization problems, using polynomial approximation techniques. The methods project the original infinite dimensional controls space into a polynomial subspace. Our contribution includes new parameterization formulations using natural polynomials, orthogonal Chebyshev polynomials and Cubic spline interpolation.

We show that the proposed methods are well suited for black-box approach with stochastic global-search method as they tend to produce smooth control trajectories, while reducing the solution space size. We demonstrate their efficiency on synthetic two-dimensional problems and on a realistic 3-dimensional problem.

By contributing with a new adjoint method formulation for polynomial approximation, we implemented the methods also with gradient-based algorithms.

In addition to fine-scale simulation, we also performed reduced order modeling, where we demonstrated a synergistic effect when combining polynomial approximation with model order reduction, that leads to faster optimization with higher gains in terms of Net Present Value.

Finally, we performed gradient-based optimization under uncertainty. We proposed a new multi-objective function with three components, one that maximizes the expected value of all realizations, and two that maximize the averages of distribution tails from both sides. The new objective provides decision makers with the flexibility to choose the amount of risk they are willing to take, while deciding on production strategy or performing reserves estimation ($P10, P50, P90$).

DEDICATION

To my brother, Alon, that I lost and always find him in my dreams.

To my wife, Ayala, that I was lost till I found her to live my dreams.

To my kids, Yonatan, Mia Bella and my dog, Tsuni.

To my brothers, may they all live long, Tamir, Amit, Noam and Asi and their beloved families.

Last but certainly not least, to my strong and smart parents, Dan and Elisheva.

ACKNOWLEDGMENTS

I would like to thank my committee chair, Dr. Gildin, for being much more than an academic advisor for me, and for always believing in me and teaching me the powerful tool of positive thinking.

I would like to thank my committee co-chair Dr. Ehlig-Economides for the endless knowledge and support she provided me, and for being the reason for me to come all the way from Israel to Texas to pursue my PhD in Petroleum Engineering.

I would like to thank my committee members, Dr. Datta-Gupta and Dr. Butenko, for their guidance and support throughout the course of this research.

A big thank is devoted to the late Dr. Floudas, who was a committee member before his unfortunate sudden death. Dr. Floudas provided the initial ignition to this research and his contribution was invaluable to me. His death is a great loss.

I would like also to thank Dr. King and Dr. Nasarbadi, for attending my preliminary and final examinations and providing added value.

I would like to thank Foundation CMG and the Israeli Ministry of National Infrastructures, Energy and Water Resources for the financial support for this research.

Thanks also to my friends and colleagues and the department faculty and staff for making my time at Texas A&M University a great experience.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Professors Gildin (advisor) and Datta-Gupta of the Petroleum Engineering Department at A&M, Professor Ehlig-Economides (co-advisor) of the Petroleum Engineering Department at University of Houston, and Professor Butenko of the Department of Industrial and Systems Engineering at A&M.

This work was supported also by the late Professor Floudas of the Chemical Engineering Department and Texas A&M Energy Institute, who was part of the dissertation committee before his sudden unfortunate death.

The results and theory presented in Sections 3.4 and 4.3, obtained through a collaboration with Professor Gildin and Hardikkumar Zalavadia of the Petroleum Engineering Department and were published in 2017 in an article listed in the Reservoir Simulation Conference proceeding.

The results and theory presented in Sections 3.1, 3.2, 4.1 and 4.2.2 obtained through a collaboration with Professor Gildin of the Petroleum Engineering Department, with Burcu Beykal of the Chemical Engineering Department and Professor Fani Boukouvala of the School of Chemical & Biomolecular Engineering at Georgia Tech, and were published in 2016 in an article listed in Computational Geoscience.

All other work conducted for the dissertation was completed by the student independently.

Funding Sources

First year of graduate study and research was supported by a fellowship from Economides funds. The second, third and fourth years of graduate study were supported by a fellowship from Foundation CMG.

All four years of graduate study and research were also supported by the Israeli Ministry of National Infrastructures, Energy and Water Resources under the scholarship program to PhD and post doctoral positions in leading universities abroad in the areas of science and engineering of natural gas and oil.

TABLE OF CONTENTS

Page

ABSTRACT ii

DEDICATION iii

ACKNOWLEDGMENTS iv

CONTRIBUTORS AND FUNDING SOURCES v

TABLE OF CONTENTS vii

LIST OF FIGURES x

LIST OF TABLES xiv

1. INTRODUCTION 1

1.1 Waterflooding Optimal Control 2
1.1.1 Reservoir Simulation PDE 2
1.1.2 The Objective Function 6
1.1.3 The Infinite Dimensional Problem 7
1.1.4 The Finite Dimensional Problem 7
1.2 Motivation for Polynomial Approximation 10
1.3 Scope and Objectives 14
1.4 Dissertation Organization 16

2. POLYNOMIAL APPROXIMATION AND OPTIMIZATION METHODS . . . 17

2.1 Introduction to Polynomial Approximation Theory 17
2.1.1 Polynomial Approximation: Continuous Functions 19
2.1.2 Polynomial Approximation:Non-Continuous Functions 24
2.2 Piecewise Zero Order Polynomial Approximation 26
2.3 Natural Polynomial Approximation 28
2.4 Orthogonal Chebyshev Polynomial Approximation 29
2.5 Piecewise Polynomial Approximation by Spline Interpolation 33
2.6 Time and Control Scaling 35

3. OPTIMIZATION METHODS 37

3.1 Global versus Local Search in Production Optimization 37
3.1.1 Optimization Dimensionality Reduction 39
3.2 Gradient-Free Optimization Using Particle Swarm Optimization 45
3.3 Adjoint Method for Polynomial Approximation 48
3.4 Model Order Reduction 53
3.4.1 Model Order Reduction with On-line Training 54
3.5 Optimization Under Uncertainty 56
3.5.1 Multi-Objective Optimization 58
3.5.2 Expected Value 58
3.5.3 Conditional Value-at-Risk 59
3.5.4 Conditional Value-at-Success 61
3.5.5 Total Objective Function 62

4. NUMERICAL RESULTS 64

4.1 Optimization Dimensionality Reduction 65
4.1.1 Example 1: Two-Dimensional Model 65
4.1.2 Example 2: Three-Dimensional UNISIM-I-D Model 71
4.1.3 Dimensionality Reduction: Results Conclusions 75
4.2 Gradient-Based versus Gradient-Free 76
4.2.1 Example 1: Two-Dimensional Model 77
4.2.2 Example 2: Three-Dimensional UNISIM-I-D Model 81
4.2.3 Gradient-Based versus Gradient-Free: Results Conclusions 89
4.3 Model Order Reduction with Polynomial Approximation 89
4.3.1 Problem Description 89
4.3.2 On-line Training 91
4.3.3 POD-DEIM Performance for a Representative Case 91
4.3.4 Comparison: PSO Optimal Control Solutions 94
4.3.5 Global versus Local Search with POD-DEIM 96
4.3.6 Model Order Reduction: Results Conclusions 99
4.4 Optimization Under Uncertainty 101
4.4.1 Test Case Description 102
4.4.2 Uncertainty Reduction by Control Refinement 102
4.4.3 Testing the Multi-Objective Function 105
4.4.4 Optimization Under Uncertainty: Results Conclusions 108

5. CONCLUSIONS AND FUTURE SCOPE FOR STUDY 113

5.1 Suggested Future Work 114

REFERENCES 116

APPENDIX A. MODEL ORDER REDUCTION: POD-DEIM 128

A.1 Proper Orthogonal Decomposition (POD) 128
A.2 Discrete Empirical Interpolation (DEIM) 130

APPENDIX B. INCOMPRESSIBLE SEQUENTIAL SOLVER 133

LIST OF FIGURES

FIGURE Page

1.1 Example of piece-wise zero order polynomial approximation often used in the literature. 10

1.2 (a) Permeability field and wells for the motivating example. (b) Permeability and porosity ranges for each facies. 11

1.3 Best solution found by Left: Local (gradient-based) search and Right: Global (stochastic) search. 12

1.4 Polynomials functions (red curves) fitted to adjoint gradient-based solution (blue dots). 13

2.1 Illustration of a continuous control function with infinite different infinitesimal segments, ∂x. Therefore, the original problem lies within an infinite dimensional space. 19

2.2 Control function with two-non-differentiable points, when it meets the upper constraints. 21

2.3 The function $f(x) = |x|$ and its fitting to varying degree polynomials . . . 23

2.4 Fitting error as a function of a polynomial degree 23

2.5 Control function from Figure 2.1 and its fitting to varying degree polynomials. 25

2.6 Control function from Figure 2.2 and its fitting to varying degree polynomials. 25

2.7 Comparison of Monomial basis functions and Chebyshev basis functions. 30

2.8 Demonstration of the control and time scaled box with three polynomials resulted from a linear combination of Chebyshev polynomials 32

3.1 Pseudo-code for PSO maximization with a dynamic neighborhood (adopted from Matlab documantaion). 46

3.2 Implementation test for second order Chebyshev polynomial approximation (three coefficients per well): Adjoint gradients are compared against numerical finite-difference gradients. Each curve represents a different well. Horizontal axis: the three coefficients. Vertical axis: gradients magnitude. The test was performed on the 5-spot case presented in section 1.2. 53

3.3 Work-flow for optimization-online training of POD-DEIM. 55

3.4 Illustration of VaR and CVaR for a given distribution. Realizations with cumulative distribution (shaded area) that falls below α constitute the distribution lower tail. VaR is the highest NPV amongst these realizations, while CVaR is their weighted average. 60

3.5 Illustration of VaR, CVaR, Expected Value (EV), VaS and CVaS for a given distribution. 63

4.1 Example 1 - FCM and Free controls - results of the best objective function found among all 10 runs. 69

4.2 Example 1 - FCM and Free controls - First 100 iterations of the best, worst and closest to mean runs out of the 10 runs. 69

4.3 Example 1 - ICM and Free controls - results of the best objective function found among all 10 runs. 70

4.4 Example 1 - ICM and Free controls - First 100 iterations of the best, worst and closest to mean runs out of the 10 runs. 70

4.5 UNISIM Benchmark: a) Permeability distribution. b) Completed wells intervals of the four vertical and 10 horizontal producing wells (in red) and 11 horizontal injection wells (in blue) used for production optimization (following the work in [1]). 71

4.6 (a) Gradient free (PSO) and (b) gradient based (IPOPT) results for different numbers of basis functions, normalized to the best solution found ($17.73 million). 79

4.7 (a) Gradient free (PSO) and (b) gradient based (IPOPT) number of iterations. 80

4.8 Optimal controls of level 10 obtained by each parametrization method. Top: Gradient-free (PSO). Bottom: Gradient-based (IPOPT). 82

4.9 Example 2 - a) Gradient-free methods best runs comparison. b) Gradient-based methods comparison with final solutions. c) Zoomed-in to the first 200 iterations of the ICMs, PCMs and the IP-LBFGS methods. 83

4.10 Example 2: Percentage difference between the best solutions of FCMs and ICMs to IP-LBFGS. Left: Three variables per well. Right: Four variables per well. Markers and lines are differences before and after convergence, respectively. Final differences after IP-LBFGS convergence are shown. . 84

4.11 Top view: control results obtained by the ICMs, FCMs , Free PSO and the gradient-based methods. 86

4.12 Side view: control results obtained by the ICMs, FCMs , Free PSO and the gradient-based methods. 87

4.13 Best run of 4 points interpolation for Example 2 - water cut, producers oil and water rates, injector water rates and economics measures 87

4.14 Reservoir permeability for testing POD-DEIM with polynomial approximation (after [2]) 90

4.15 a) Pressure and (b) Saturation basis selection for POD and (c) cells selected for DEIM (in strong colors). 92

4.16 Three training samples from the initial eight-particles swarm of Chebyshev profiles. 92

4.17 Top) Saturation profiles from POD-DEIM Chebyshev optimal control using fine scale simulation and (Below) using the reduced model at different times and their relative errors. 93

4.18 Oil production rate from the high fidelity simulation and reduced model simulation for 4 production wells using the Chebyshev optimal control solution from POD-DEIM. 94

4.19 Water production rate from the high fidelity simulation and reduced model simulation for 4 production wells using the Chebyshev optimal control solution from POD-DEIM. 95

4.20 Control trajectories relative error for a PSO algorithm. 95

4.21 PSO optimal control solutions from POD-DEIM and fine-scale using all three parameterization methods. 97

4.22 Relative NPV errors between the fine scale and reduced scale optimization using PSO and IPOPT for each parameterization methodology. 98

4.23 Ensemble of 30 permeability field realizations of channelized reservoirs . 102

4.24 Histogram of permeability values for all realizations. 103

4.25 Maximizing expected NPV for all realizations: Sensitivity of number variables per well to expected (mean) and standard deviation values. 104

4.26 Maximizing expected NPV for all realizations: optimization efficiency. . 105

4.27 Maximizing total objective J_{tot}: results for different combinations of weights η_i. Results are given for the averages of the all ensemble of realizations, and of the bottom and top 20% (AVG, CVaR and CVaS, respectively). Best (MAX) and worst (MIN) results are also plotted. 106

4.28 Maximizing total objective J_{tot}: Ranking of each η_i combination according to each objective function: (top) J_E, (middle) J_{CVaR}, (bottom) J_{CVaS}. Ranking score range from 1 (left) to 8 (right) 108

4.29 Accumulated ranking according to Fig. 4.28. 109

4.30 Ranking of each η_i combination according to their P percentile: (top) $P20$, (middle) $P50$, (bottom) $P80$. Ranking score range from 1 (left) to 8 (right) 110

4.31 Accumulated ranking according to Fig. 4.30. 111

4.32 Empirical cumulative distribution function (CDF) for selected η_is combinations. 111

4.33 probability density function for selected η_is combinations. 112

LIST OF TABLES

TABLE Page

3.1 Parameter values used in both examples for the Particle Swarm Optimization algorithm 48

4.1 Control set cardinality and space dimensionality reduction for Example 1 66

4.2 Variables hard bounds in both example 1 and example 2. 67

4.3 Example 1: Results of 10 optimization runs, (Function tolerance: 10^{-6} for 20 consecutive stall iterations) 68

4.4 Control set cardinality and solution space dimensionality reduction for Example 2. 73

4.5 Example 2: Results of 10 optimization runs, (Function tolerance: 10^{-4} for 12 consecutive stall iterations) 74

4.6 Average simulation duration for each method. 88

4.7 NPV and number of iterations for fine scale and reduced scale optimization using the 3 methods with IPOPT. NPV values are in million USD 96

4.8 NPV and number of iterations for fine scale and reduced scale optimization using the 3 methods with PSO. NPV values are in million USD 98

1. INTRODUCTION

Reservoir management requires making many decisions throughout the life of a hydrocarbon field. Within this decision-making processes, incorporating reservoir simulation has become increasingly common in the petroleum industry. The quest for optimal decisions with the aid of reservoir simulation can be classified as PDE-constrained optimization, as reservoir simulation solves numerically partial differential equations (PDE) describing the underground flow.

The broad class of PDE-constrained optimization problems includes problems such as optimal design, inverse problem and optimal control [3].

With reservoir simulation, we can simulate different processes of hydrocarbons recovery. Primary recovery is the process of utilizing the natural energy stored in a reservoir, combined with artificial lift techniques (such as pumps), to withdraw oil and gas. Secondary recovery is the injection of fluids into the ground to increase reservoir pressure and to enhance production. This process is often called reservoir flooding, and when water is the injected fluid, the process called waterflooding, which is the most used secondary recovery method worldwide [4].

In waterflooding, once the wells are drilled, the main reservoir management task is to determine how much water to inject from injection wells and how much fluid to withdraw from production wells and at what pressures. The development of smart down-hole valves, as part of an intelligent completion of a wellbore, has encouraged researchers to address this reservoir management optimization problem with the theory of optimal control [5]. Then, the common goal is to find a continuous scheme to control the opening level of each valve at any point in time, which maximize some financial measure. Although the formulation of the problem is somewhat easy to state, it has brought up issues with the size of the large-scale optimization and the computational efficiency.

1.1 Waterflooding Optimal Control

Optimal control is a PDE-constrained optimization problem, where the PDE describes some dynamic system that evolves through time. In waterflooding optimal control, reservoir simulation is often used to solve a dynamic system of flow in a porous media and wells. Then, the goal is to seek inputs to the simulator that optimize some objective function.

In section 1.1.1, we briefly describe the governing equations and numerical procedures within a reservoir simulation. Then, We briefly mention several methods to perform reduced order modeling to ease the computational burden of a reservoir simulation. We then show how the simulation equations can be presented in a typical optimal control state space formulation.

In section 1.1.2, we describe the objective function used in this work, namely, the Net Present Value (NPV) integral function. Then in section 1.1.3, we integrate previous sections to present the infinite dimensional optimal control problem. Thereafter, we describe how to transform the infinite dimensional problem into a solvable finite dimensional problem.

1.1.1 Reservoir Simulation PDE

We consider two-phase immiscible system containing oil (o) and water (w). The mass conservation equations for the two phases, with Darcy law and gravity, in its differential form can be written as follow:

$$\frac{\partial}{\partial t}\left(\Phi\rho_l s_l\right) = \nabla\cdot\left(\frac{\rho_l k_{rl} K}{\mu_l}\left(\nabla p_l - \rho g \nabla z\right)\right) - \tilde{m}_l, \tag{1.1}$$

where, l = liquid (oil or water), ϕ = porosity of the porous medium, ρ = density of the fluid per unit volume, and $\tilde{m}$ denotes the external sources (injection, negative) and sinks (production, positive) as mass per unit volume per unit time. μ, k_{rl} and K are the viscosity, relative permeability and absolute permeability, respectively. p_l is the pressure of each phase, and s_l is the saturation of each phase. g is the gravity acceleration constant and z accounts for vertical coordinates.

Eq. 1.1, written for $i = o, w$, along with the saturation equation $(s_o + s_w = 1)$ and capillary equation $(p_{cow} = p_o - p_w)$ defines the flow problem. The equations are then discretized and two equations with two variables are assigned for each grid cell: p_o and s_w.

Here, we take a simplified form of the conservation equations in porous media. Many books and papers have been published in this area, and the keen reader can visit classical references as [6, 7, 8, 9, 10].

Using an implicit formulation, the two coupled equations, one for each phase, can be rearranged in a matrix form:

$$T^{n+1}\mathbf{x}^{n+1} + D^{n+1}(\mathbf{x}^{n+1} - \mathbf{x}^{n}) + G^{n+1} + Q^{n+1} = R^{n+1}, \tag{1.2}$$

where, $\mathbf{x} = [\mathbf{p_o}, \mathbf{s_w}]^T$, D is the Accumulation matrix, T is the Transmissibility matrix, G is the Gravity vector and Q contains the (volumetric) q_l sources and sinks terms, and R is the residual vector. We satisfy $R^{n+1} = 0$ upon convergence to the solution for each time step (see [8] for detailed derivation). This nonlinear system of equations is solved using Newton's method:

$$J^{n+1}\delta^{n+1} = -R^{n+1} \tag{1.3}$$

$$\mathbf{x}^{n+1} = \mathbf{x}^{n} + \delta^{n+1}, \tag{1.4}$$

where, the Jacobian matrix is $J^{n+1} = \partial R^{n+1}/\partial \mathbf{x}^{n+1}$. The numerical solution size usually ranges from hundreds to millions of cells. For example, for a reservoir with the order of 100,000 cells - 80%-90% of the total simulation time is spent on solving Equation 1.3 with a linear solver [7]. Considering the computational complexity of performing single reservoir simulation, one can realize the need for both efficient optimization methods and simulation. As we will see in the next chapters, reduced order reservoir modeling and control parameterization are some of the ways to mitigate such problems.

Reduced Order Reservoir Modeling

Reservoir Simulation involves solving a large scale system of nonlinear equations, which might render this problem to be computationally expensive for real-time processing. This might be the case in production optimization and history matching problems which require several hundred to thousands of simulation runs.

Numerous techniques have been applied to reduce the computational costs associated with solving the high fidelity models across different branches of engineering, of which Reduced Order Modeling techniques are often sought-after for many applications such as Design Optimization [11], Control [12], and Computational Fluid Dynamics [13]. Reduced Order Modeling techniques have been investigated in the past decade in the context of reservoir simulation and optimization to mitigate the computational cost associated with the large-scale nature of the reservoir models while maintaining a high order of accuracy. This is achieved by transforming high-dimensional models into lower-dimensional representations that contain dominant characteristics of the corresponding solution space to replicate the input-output behavior of the fine scale model.

Previously, reduced order modeling procedures have been applied for the well-control optimization problem using POD [14], POD-TPWL [15, 16, 17], POD-TPWQ [18] and POD-DEIM [19, 1, 20]. We will devote one section in this work to the last one, POD-

DEIM (Proper Orthogonal Decomposition - Discrete Empirical Interpolation Method).

Reservoir Modeling as a State Space Equation

In order to formulate an optimal control problem, Eq. (2) can be rearranged as follow:

$$\dot{\mathbf{x}} = \tilde{D}^{-1}T\mathbf{x}^{n+1} - \tilde{D}^{-1}G - \tilde{D}^{-1}Q, \tag{1.5}$$

where $\tilde{D} = D\Delta t$.

Note that the control input is assigned through the source-sink q_l terms found in the Q matrix. q_l is the volumetric flow rate defined as:

$$q_l = \frac{\tilde{m}_l \Delta x \Delta y \Delta z}{\rho_l} = WI_l(p_o - p_{wf}), \tag{1.6}$$

where $\Delta x, \Delta y$ and Δz are a given grid cell dimensions and ρ_l is the phase l density. WI_l is the well index associated with each phase, p_o is the pressure of the oil phase in the well block and p_{wf} is the bottom hole flowing pressure. The control input can either be the value assigned as q_l (known as rate control) or as p_{wf} (known as pressure control). Another option is to introduce additional valve control variable as was done in [21].

To this end, as was discussed in [22], the controls enter linearly into the flow model, and we can rewrite the flow equation as:

$$\dot{x}(t) = f_1(x(t)) + f_2(x(t))u(t), \tag{1.7}$$

where $u(t) = [u_1(t), \ldots, u_j(t), \ldots, u_{N_w}(t)]^T$ is a column vector contains continuous control functions of each well j. Here, N_w is the total number of wells. Originally, prior to any discretization, each u_j function maps a continuous time input to a continuous control trajectory.

Equation 1.7 represents the governing partial differential reservoir flow equation, which is now formulated in a well-known optimal control form that describes the reservoir system dynamics in a state space form.

1.1.2 The Objective Function

We consider the Net Present Value (NPV) function as the objective, which accounts for revenue associated with produced oil and for the cost of handling produced and injected water (which is incurred as a result of pumping and separation requirements),

$$J(u) = NPV = \int_0^{T_f} \left(\sum_{j=1}^{N_p} r_o q_o^j(u) - \sum_{j=1}^{N_p} c_{wp} q_{wp}^j(u) - \sum_{j=1}^{N_i} c_{wi} q_{wi}^j(u) \right) \frac{1}{(1+ir)^{t/t_{ref}}} dt, \quad (1.8)$$

where q_o^j, q_{wp}^j and q_{wi}^j are the flow rates of the oil, water produced and water injected for well j, respectively. The revenue from a unit of oil produced, the cost of a unit of water produced and a unit of water injected are represented by r_o, c_{wp} and c_{wi}, respectively. t and T_f are continuous and terminal times, respectively. ir is the interest rate associated with a time reference t_{ref}. N_i and N_p are the total number of injection and production wells, respectively.

Again, as can be observed from Equation 1.8 and as was discussed in [22], the performance measure (the objective function) is linear in the control and can be rewritten in a typical optimal control form:

$$J(u) = \Psi(x(T_f), T_f) + \int_0^{T_f} L(x(t), u(t), t) dt, \quad (1.9)$$

where Ψ is the terminal cost function such as wells plug and abandonment (P&A) costs, and L is the integrand inside the integral in Equation 1.8.

1.1.3 The Infinite Dimensional Problem

Given Eqs. 1.7 and 1.9, we can now introduce the infinite dimension optimal control problem, as follow:

$$\max_{u(t)} \quad \Psi(x(T_f),T_f) + \int_0^{T_f} L(x(t),u(t),t)dt \tag{1.10a}$$

subject to

$$\dot{x}(t) = f_1(x(t)) + f_2(x(t))u(t) \tag{1.10b}$$

$$u(t) = \left[u_1(t),\ldots,u_j(t),\ldots,u_{N_w}(t)\right]^T \tag{1.10c}$$

$$u_j(t) \in \mathbb{U}, \forall t \in [0,T_f] \tag{1.10d}$$

$$\mathbb{U} = \left\{u_j(t) : u_{lb} \leq u_j(t) \leq u_{ub}\right\} \tag{1.10e}$$

$$x(0) = x_0, \tag{1.10f}$$

where the initial conditions are specified as $x(0) = x_0$, and the original bounded infinite dimension control space is defined by $\mathbb{U}$. All the variables in problem 1.10 are continuous (or piecewise continuous), and thus the problem lies in an infinite dimensional space.

The set of equations 1.10 are called the Bolza problem. Two special cases of the Bolza problem are the Lagrange problem when there is no terminal cost, i.e., $\Psi = 0$, and the Mayer problem, in which there is no running cost, i.e., $L = 0$ [23].

1.1.4 The Finite Dimensional Problem

In general, there are two distinct classes of algorithms to solve optimal control problems such as shown in Equation 1.10, namely indirect and direct methods [24]. In the former, one analyzes the optimality conditions before applying numerical discretization. Such indirect methods are dynamic programming which solves the Hamilton-Jacobi-Bellman (HJB) equation (suitable to a few state variables problem) and the Pontryagin Maximum

Principle (PMP) method [25].

In the direct method, on the other hand, the optimal control problem is approximated by a finite-dimensional nonlinear programming (NLP) problem. Then, different standard optimization algorithms (stochastic or deterministic, global or local) can solve the approximated problem. Direct methods include single shooting, multiple shooting, and orthogonal collection approaches [26]. The last two methods introduce many decision variables and constraints which might lead to prohibitive computational efforts in reservoir simulation problems. However, implementation of multiple shooting for reservoir flooding problem was recently introduced in [27].

In this work, we use a direct single shooting approach to solving the optimal control problem. This method parameterizes the control trajectories along time, and the parameterization coefficients become the decision variables in the finite-dimensional optimization problem.

To solve the problem formulated in Eq. 1.10, spatial and temporal parameterization of the state variables x takes place within the reservoir simulator, where control variables u temporal parameterization occurs on the optimization level.

In order to cast the solution space into a reduced space, for each well j we perform a linear combination of basis functions, denoted as $\Phi(t_k)$, as follows:

$$\forall t_k \in \{t_0,\dots,t_{N_{sim}}\}, \quad u_j(t_k) \approx \tilde{u}_j(t_k) = \sum_{i=1}^{N_\Phi} w_{i,j}\Phi_{i,j}(t_k), \tag{1.11}$$

where w is a weight value determined by the optimizer (that is, w values become the decision variables). $\Phi(t_k)$ will be evaluated for each simulation time step t_k on the bounded interval $\{t_0,\dots,t_{N_{sim}}\}$.

Let us now denote $\tilde{\mathbf{u}}(t_k) = \left[\tilde{\mathbf{u}}_1(t_k),\dots,\tilde{\mathbf{u}}_j(t_k),\dots,\tilde{\mathbf{u}}_{N_w}(t_k)\right]$ as the vector contains approximations for all wells at each time step. By inserting Eq. 1.11 into Eq. 1.10, we project the infinite dimensional optimization problem into a finite dimension (of a degree $N_\Phi \cdot N_w$) optimization problem. Taking into account the simulation time discretization, we get a new problem formulation, as described by Eq. 1.12.

Note that the integral from Eq. 1.10 was replaced in Eq. 1.12 by a summation over N_{sim} simulation time steps, such that the flow rates are now associated with discrete values with an index k. In this formulation, there is a finite number, n, of decision variables.

$$\max_{w_{i,j}} \quad \Psi(\mathbf{x}(t_{N_{sim}}), t_{N_{sim}}) + \sum_{k=0}^{N_{sim}} L(\mathbf{x}(t_k), \tilde{\mathbf{u}}(t_k), t_k) \tag{1.12a}$$

subject to

$$\dot{\mathbf{x}}(t_k) = f_1(\mathbf{x}(t_k)) + f_2(\mathbf{x}(t_k))\tilde{\mathbf{u}}(t_k) \tag{1.12b}$$

$$\tilde{\mathbf{u}}_j(t_k) = \sum_{i=1}^{N_\Phi} w_{i,j}\Phi_{i,j}(t_k), \forall j \in \{1, \ldots, N_w\} \tag{1.12c}$$

$$w_{lb} \leq w_{i,j} \leq w_{ub} \tag{1.12d}$$

$$x(0) = x_0. \tag{1.12e}$$

A common approach to parameterize a well-control is by the so-called Piece-Wise-Constant (PWC), or piece-wise zero order polynomial approximation. This method specifies a predefined number of intervals along a time horizon, and the goal is to find the optimal constant value for each interval. Figure 1.1 shows an example of piece-wise zero order polynomial approximation. This method is used in almost all published work on well-control production optimization ([5, 28, 29, 30, 31], to list a few examples out of many). There are also several PWC approaches that design an optimization scheme that dynamically adjusts the number of intervals throughout the optimization process [32, 33, 34].

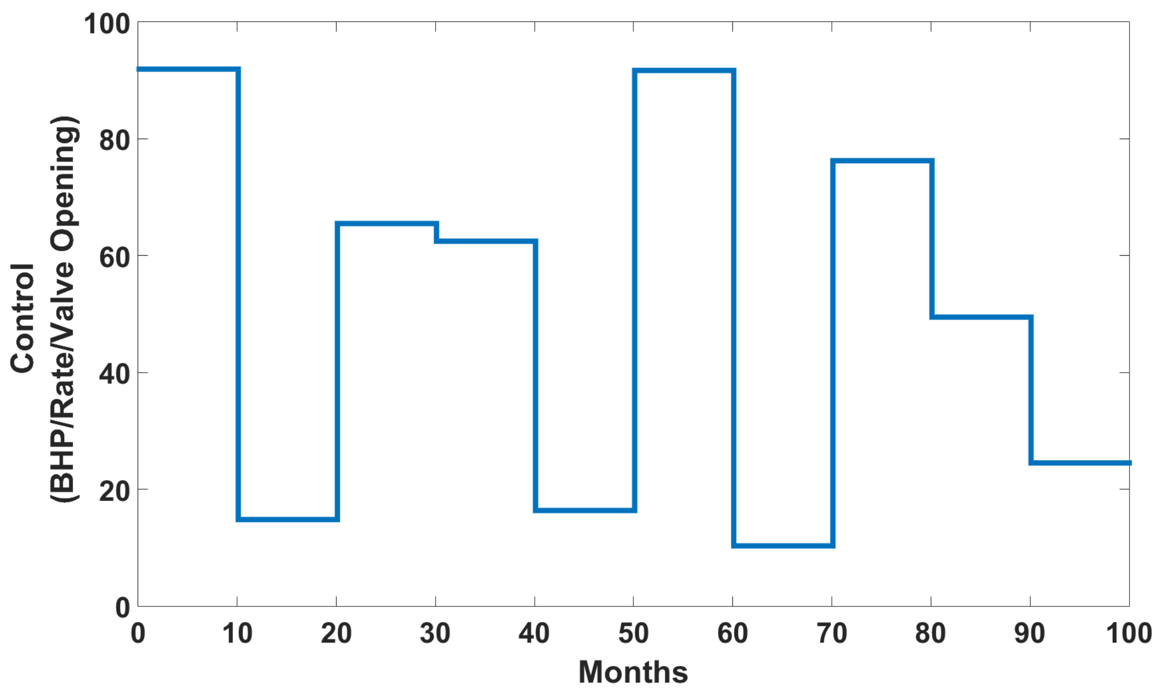
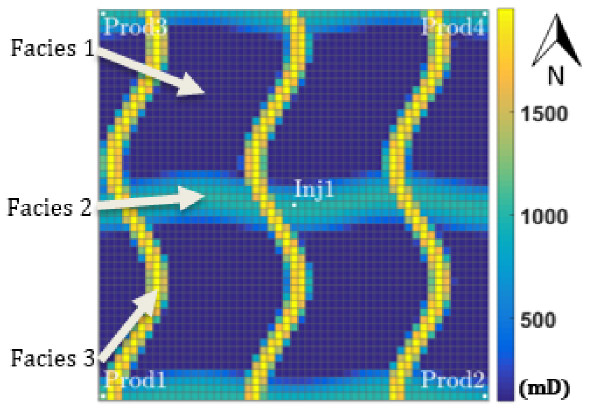


Figure 1.1: Example of piece-wise zero order polynomial approximation often used in the literature.

1.2 Motivation for Polynomial Approximation

Let us introduce a synthetic, two-dimensional and channelized reservoir. The model includes 51 by 51 grids consisting of four producing wells and one injector arranged in a five-spot pattern. The grid cell dimensions are 20ft x 20ft x 20ft. Fig. 1.2a shows the permeability distribution as well as the five wells. There are three permeability sub-regions in three distinct geological facies: the original low-permeability geological environment (Facies 1), east-west moderate permeability channels formed by ancient rivers (Facies 2) and newer high-permeability channels whose flow eroded the environment from north to south (Facies 3). Fig. 1.2b provides a table with the minimum, maximum and average permeability values for each sub-region as well as the homogeneous assigned porosity. The initial water saturation is 0.2 (equal to residual water saturation).



(a)

	Min Perm (mD)	Max Perm (mD)	Ave. Perm (mD)	Porosity
Facies 1	100	100	100	
Facies 2	102.3	2000	1366.7	0.2
Facies 3	100.2	1000	679.2	

(b)

Figure 1.2: (a) Permeability field and wells for the motivating example. (b) Permeability and porosity ranges for each facies.

We used the flow transport two-phase incompressible solvers Matlab Reservoir Simulation Toolbox (MRST) as our forward model simulator [10, 35]. Throughout this work, we controlled all wells by adjusting BHP, so that we determine the optimal BHP for each well at all the different time intervals. We simulated a one-year production and adjusted the control every ten days, leading to 185 optimization variables. Throughout this paper, we considered an oil price of \$100 per barrel, produced and injected water treatment costs of \$10 per barrel, and a discount rate of 10%.

We performed the optimization for this example using two approaches: global search with a parallelized dynamic-neighborhood Particle Swarm Optimization (PSO) algorithm, as detailed in section 3.2, and local search with the adjoint aggressive-line-search method MRST implementation [36].

Figure 1.3 shows the best solution found by the two algorithms. We can see that the global optimizer found a solution with a slightly better NPV with more even water distribution throughout the reservoir. Though the convergence mechanism and convergence criteria differ between the two methods, our main goal in this section is to compare the

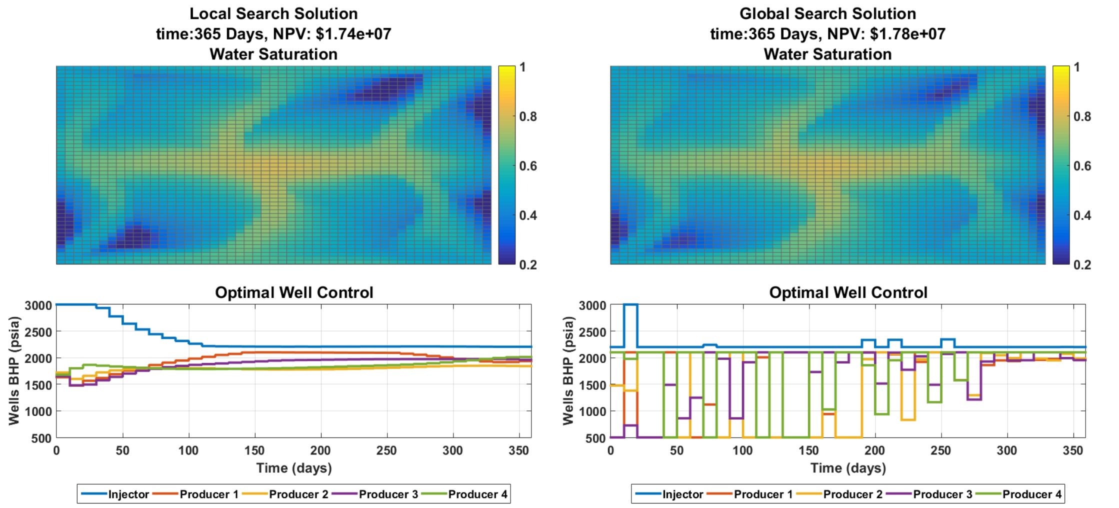


Figure 1.3: Best solution found by Left: Local (gradient-based) search and Right: Global (stochastic) search.

shape of the control trajectory solutions found by each method, rather than the obtained objective function values.

We can see from Figure 1.3 that while the gradient-based method exhibits a smooth solution, the stochastic nature of the global optimizer provides an erratic and impractical control strategy. This control strategy might not be acceptable since such dramatic repeated changes in the pressure will quickly cause a fatigue to production instruments and will damage the equipment. Therefore, in developing a new optimization method, it is better to achieve a smooth and practical solution for the control trajectories, while maintaining the ability to obtain better NPV values.

The adjoint smooth solution, shown in Figure 1.3, inspires considering a new method, other than PWC, that will represent each control trajectory with a polynomial function. Figure1.4 shows the results of fitting this smooth adjoint control trajectories solution to different polynomial functions (with degrees ranging from one to four). Of course, a better

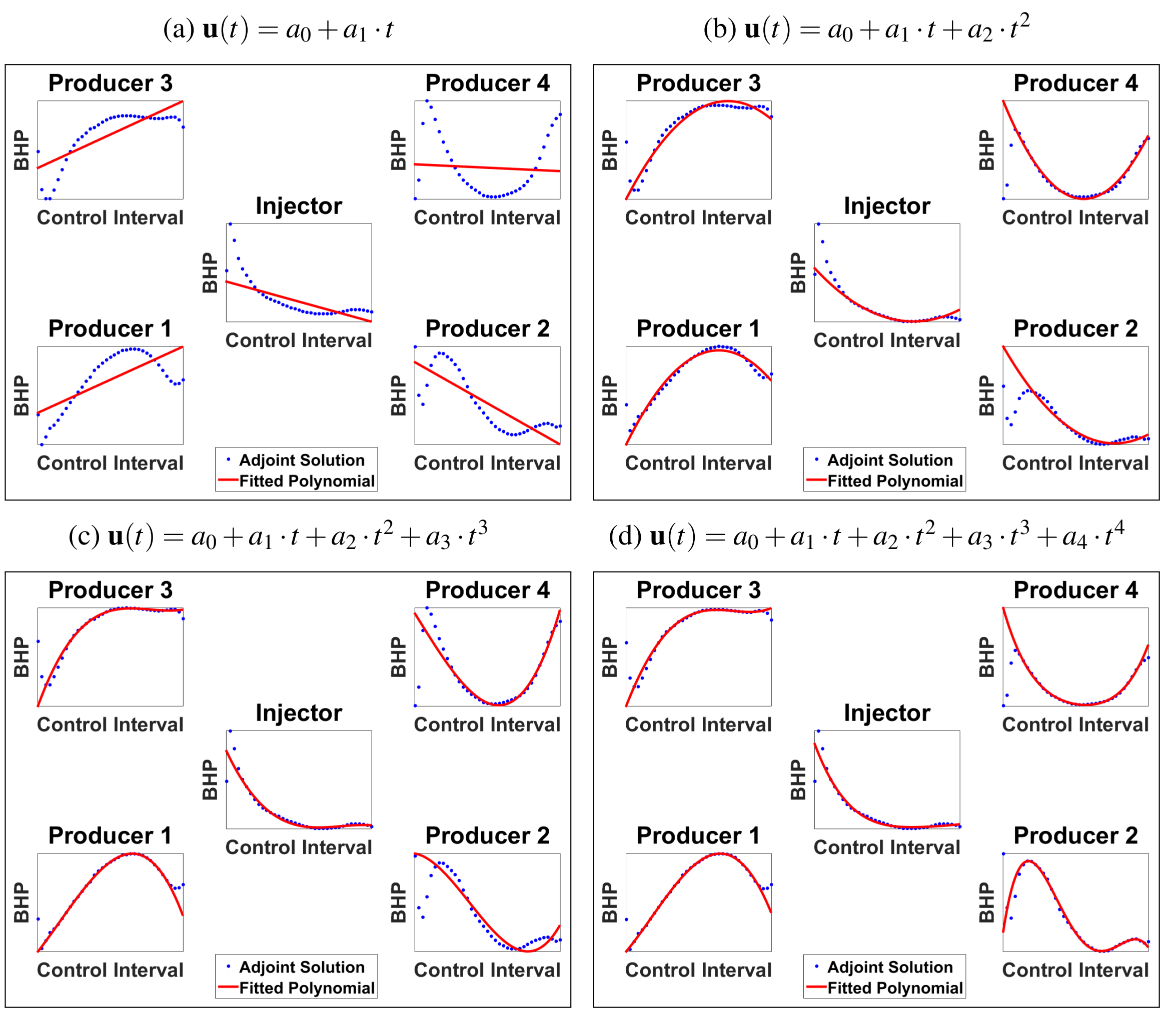


Figure 1.4: Polynomials functions (red curves) fitted to adjoint gradient-based solution (blue dots).

fit is observed for higher degrees, as higher order polynomials can capture multimodal trajectories (for example, see Producer 2 trajectory).

However, it has been repeatedly argued in the literature [37, 38, 31] that the objective function might have a high plateau with an infinite number of control realizations that give similar field life-cycle NPV. Hence, it might be possible to find smaller degree control polynomials amongst the infinite options that leads to the objective function's high plateau.

1.3 Scope and Objectives

Following the motivation example in the previous section, and since clearly there is a lack of published petroleum-related work with higher order polynomial parameterizations, the scope of this work is to purpose novel representation of the control trajectories in waterflooding optimization problems, using polynomial approximation techniques. We will compare the proposed methods with global and local optimization algorithms, and by both fine and reduced order modeling. Eventually, we will consider both deterministic and probabilistic optimization cases.

Specifically, we look into different polynomial approximation methods, and we go a great deal further than the common piece-wise zero order polynomial approximation found in most waterflooding optimization literature. The purposed methods are well suited for the stochastic global-search method as they produce smooth control trajectories while reducing the solution space size. We demonstrate its efficiency on synthetic two-dimensional problems and a realistic 3-dimensional problem.

Waterflooding optimal control problems pose computational difficulties not only in the optimization levels but also at the simulation level. Reduced order modeling techniques can alleviate the hurdle at the simulation level. In this work, we also contribute with a novel workflow that combines polynomial approximation with reduced order modeling.

We address first deterministic cases, that is, we use a single model realization to test the different methods. We perform both stochastic, gradient-free, global-search method and gradient-based local-search method. Researchers and reservoir engineers often use gradient-free methods, where the simulation is treated as a black-box. We show that polynomial approximation serves as an efficient tool for the black-box approach executed in a high-performance computing environment.

Since the true reservoir model is never entirely known, we also consider optimization under uncertainty represented by multiple realizations. For that purpose, gradient-free methods might not be tractable to compute, and one should resort either to an approximate gradient or gradient-based methods. We choose gradient-based method while contributing a new adjoint method formulation for polynomial approximation. We purpose a new multi-objective function with three components, one that maximizes the expected value of all realizations, and two that maximize the average of distribution tails from both sides.

Our objective in this work is to present the following contributions to the field:

1. To present novel parameterization alternatives, other than commonly used PWC approach, using polynomial approximation methods.

2. To demonstrate the added values these methods bring when using a "black-box," gradient-free, optimization approach:

    (a) The purposed methods regulate and smoothen erratic solutions often obtained by stochastic global-search optimization algorithms, and thus amenable for practical deployments in the field.

    (b) Achieve a reduction in the original solution space size and thus enable efficient optimization.

3. To formulate and implement the adjoint method for a polynomial approximation that enables the use of efficient gradient-based methods.

4. To combine control polynomial approximations with reduced order modeling and to demonstrates the computational virtues of the new suggested workflow.

5. To perform optimization under uncertainty with a scenario-based approach (multiple geological realizations), and to contribute the following:

(a) Observation that the addition of more well-control parameters, not only increases the expected objective value but also reduces the associated risk.

(b) To purpose a new objective function that:

   i. Considers also the upper tails of the NPV distribution (and not just the mean and the lower tail as shown in the literature)

   ii. Provides flexibility for reserves estimation ($P10, P50, P90$).

1.4 Dissertation Organization

This dissertation continues as follows: Chapter 2 gives an introduction to polynomial approximation theory, following by a detailed discussion on the different polynomial approximation methods presented in this work. Next, Chapter 3 describes the global optimization method we use, and stress importance of re-parameterization when using the gradient-free method. This chapter also presents a workflow for gradient-based optimization under uncertainty and explains how to acquire the derivative values with the adjoint method for polynomial approximation. Then, Chapter 4 brings all numerical results of this work. We start by demonstrating the effect of dimensionality reduction, followed by performing a comparison between gradient-based methods and a gradient-free method. Then, we compare three different parameterization methods when combining optimization with reduced order modeling, and we conclude this chapter with results for optimization under geological uncertainties. Chapter 5 presents conclusions and provides a list of suggested venues for future scope of study.

2. POLYNOMIAL APPROXIMATION AND OPTIMIZATION METHODS

In this chapter, we describe the different polynomial approximation methods presented in this dissertation. We start with a general introduction to polynomial approximation theory, and then we go back to the earlier mentioned piece-wise zero order polynomial approximation method, this time with a more rigorous formulation. Then, we continue by bringing two polynomial function approximation approaches, namely, natural polynomials and orthogonal Chebyshev polynomials. We also describe how to obtain a polynomial approximation with interpolation control method, rather than a function control method, using cubic spline interpolation. Finally, we conclude by presenting the scaling procedure we perform for both time and control domains, along with an explanation why this procedure is expected to outperform previous scaling methods presented in the water-flooding optimization literature.

2.1 Introduction to Polynomial Approximation Theory

Approximation theory is a branch of study in mathematics, concerns with the search for best approximations to complicated functions by using simpler functions. We use approximation applications in many aspects of our everyday life. When we take a photo with our cell phone camera, we approximate the reality with an ensemble of pixels. When we ask a computer to compute a complicated function such as $sin(x)$ or $exp(x)$, the computer approximates these by simple basic functions, such as addition and multiplications.

The machines carry these approximations up to a desired resolution or accuracy. Storage capacity and camera quality in our cell phone set a limit on the desired resolution, whereas, in the case of mathematical function approximations, the underlying computer$'$s floating point arithmetic dictates the level of accuracy.

We measure an approximation quality not only by its accuracy but also by its efficiency. In general, the smaller the number of simpler functions to approximate the more complicated function, the better. The efficiency can be quantified both in speed and in (storage) space. The information found in the pixels of a photo are compressed and saved, for example, as a JPEG file, in which several discrete cosine or wavelet functions encode and approximate the relevant information and enable efficient data storage [39]. However, taking too few elements to represent a system would deteriorate the accuracy and the resolution. Hence, we see a trade-off between efficiency and accuracy, which lead to the question: How many basic elements are sufficient to approximate the reality?

This dissertation explores several different approaches to approximate solutions to petroleum engineering optimization problems. Specifically, we will approximate solutions to water-flood optimal control problems which aim at finding the optimal settings of a well along time. Since the original solution is a continuous function in the time domain (or piecewise continuous), we can define the original problem as an infinite dimensional optimization problem.

To illustrate that the problem lives in infinite dimensional space, we add Figure 2.1 which shows a schematic control function of a well valve opening level, as well as a zoom in into an infinitesimal segment. This continuous function can be divided into infinite different such infinitesimal segments. In other words, the optimal control function can take infinite different values along the curve shown in Figure 2.1. Methods derived from the calculus of variation that finds infinite different optimal values are often impractical [40], and thus we cast the problem into a finite dimensional space by parameterization schemes which approximate the original solution. Once we define the optimization in a finite space, we can take advantage of one out of many efficient standard non-linear optimization tools to solve the problem.

In this work, we will investigate several approximation methods, and we will test which leads to better reparameterization of waterflooding optimization problems, both in terms of

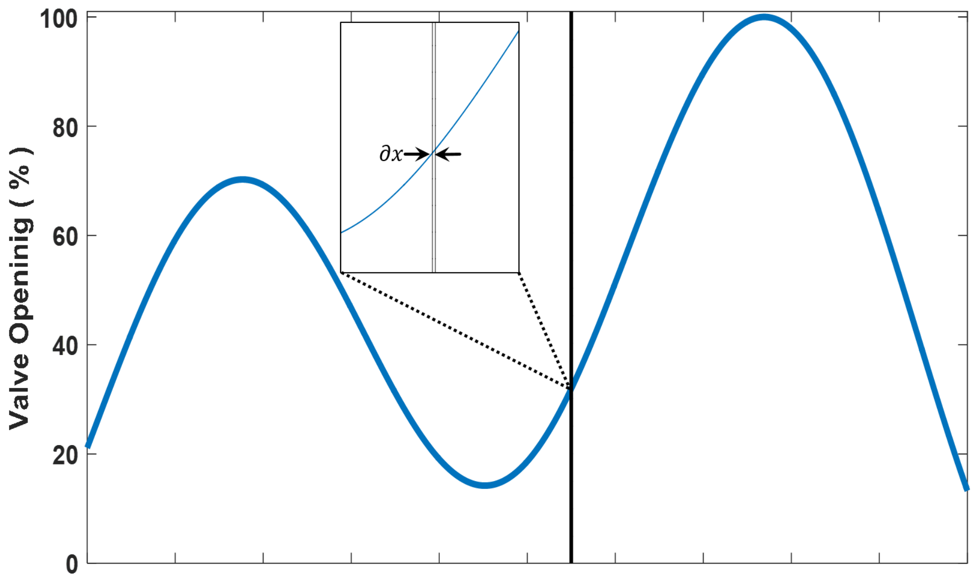


Figure 2.1: Illustration of a continuous control function with infinite different infinitesimal segments, ∂x. Therefore, the original problem lies within an infinite dimensional space.

accuracy (measured by the quality of an objective function) and by its efficiency (measured by the rate of convergence).

Researchers have been developed numerous different approximation methods over the years, including wavelet analysis, compressed sensing and multivariate approximation [41]. In this dissertation, we will investigate the most fundamental family of approximation methods – the polynomial approximation. Specifically, we will test the performances of ordinary polynomials, orthogonal polynomials, especially Chebyshev polynomials, and Cubic spline polynomials.

2.1.1 Polynomial Approximation: Continuous Functions

One of the most important theories in the development of mathematical analysis asserts as follows: we can approximate every continuous function on a bounded interval, up to an arbitrary accuracy, by polynomials. This is the famous Weierstrass approximation theorem, and it is the basis to the theory of approximation of functions of a real variable [42]. Formally, the theorem reads as follow:

Theorem 1. *Weierstrass approximation theorem. Let f be a continuous function on $[-1,1]$, and let $\varepsilon > 0$ be arbitrary. Then there exists a polynomial p such that $|f-p|_\infty < \varepsilon$,*

where, $|\cdot|_\infty$ is the infinity norm (also known as the supremum norm). In words, the maximal error between a polynomial p and the original function f is smaller than a positive arbitrarily small number ε. Karl Weierstrass was 70 years old when he proved that at 1885, and it was Carl Runge who independently discovered it at about the same time [43].

Usually, approximation theory deals with two types of problems: either seeking for a function best fitted to a given data set or alternatively, looking for a simpler representation of a given complicated function. However, is it always the case where we have at hand a data set or a true function to approximate? The answer is no.

Consider the case of optimization. For example, assume we are looking for a function of time to generate optimal profit. In this case, we are not given with data to fit, nor with a known explicit function to approximate. The true shape of the function is not known a priory, and we need to make some assumptions regarding the behavior of this function.

Since a polynomial can approximate any continuous function, Weierstrass approximation theorem provides the motivation to look for an optimal polynomial function and to seek for the optimal solution within a polynomial space.

Note, that Weierstrass approximation theorem requires only continuity, but not differentiability. In fact, Weierstrass provided an example of a function (the Weierstrass function) that it has the property of being continuous everywhere but differentiable nowhere [44]. Thus, we can approximate optimal functions with polynomials even when they are not differentiable.

In petroleum engineering, we might encounter in many cases optimal functions which are not differentiable at certain points. This often happens at points where a control function reach (or leave) a given constraint. In optimization, we call a constraint ”active” when the optimal solution lies on this constraint. At the transition point, where the constraint become active, the function might be not differentiable. Figure 2.2 depicts this situation, where we imposed an upper constraint of 80%, on the same control function from Figure 2.1. In this case, we assume, for example, that specification of some equipment installed in a well does not allow valve opening above a threshold of 80%.

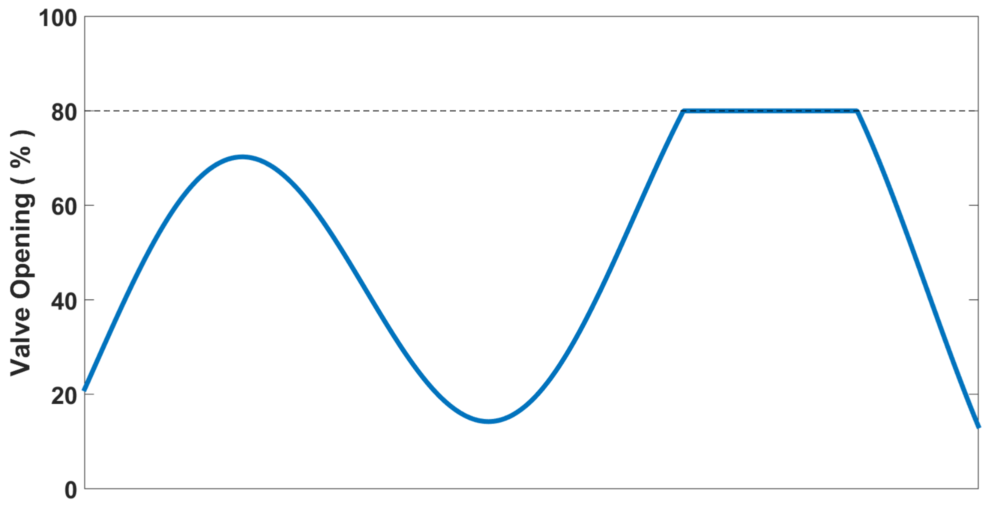


Figure 2.2: Control function with two-non-differentiable points, when it meets the upper constraints.

How the differentiability and the smoothness of a function affect the efficiency of its polynomial approximation? Weierstrass approximation theorem states only that for a given continuous function there exists a polynomial approximation. Specifically, the polynomial of degree n will converge to the approximated function as n approaches ∞. However, the theorem does not tell us the whole story. We still do not know at what value of n the approximation would suffice our needs. In different words, what is the smallest polynomial degree that is sufficient to approximate a function? Recall that for the sake of efficiency we would like to approximate a function with the fewest elements as possible, and thus with the smallest polynomial degree as possible.

To answer the question above let us first define a vital property for polynomial approximation, the smoothness property of $f(x)$. We can measure the smoothness of a function by how many derivatives it has which are continuous. Let us denote this number by ν. Then, we can define a class of continuously differentiable functions as $\mathbf{C}^{\nu}$. For example, the function $|x|$ is continuous, but its derivative is not continuous (as it is not differentiable at the origin), and thus it belongs only to the class $\mathbf{C}^0$. On the other hand, the functions $exp(x)$ and $sin(x)$ are infinite times differentiable and thus belong to the class $\mathbf{C}^{\infty}$.

Trefethen in his great book [43] demonstrates that the smoother a function, the faster its approximants converge when $n \to \infty$. In general, if a function is ν time differentiable, then the error of its approximation by a polynomial of degree n reduces at a rate proportionally to $n^{-\nu-1}$. For example, the continuous function $|x|$ is zero times differentiable, and its approximation accuracy improves at a rate of $O(n^{-1})$.[1] We demonstrate this in Figure 2.3 and Figure 2.4, where we show the improvement of the approximation as n gets bigger. We see that the polynomial degree should be very high to approximate a function when it is not differentiable.

Let us demonstrate it one more time with the two control functions shown in Figure 2.1 and Figure 2.2. The first function is differentiable (and in fact belongs to $\mathbf{C}^{\infty}$ as we built it from sinusoidal functions, and thus it is infinite time differentiable), whereas the second function is not differentiable at the two transition points where the constraints become from not active to active, and vice versa.

[1]The rate of error decay is dependent also on the total variation of a function, defined as the norm of its derivative. See chapter 7 in (Trefethen, 2013)

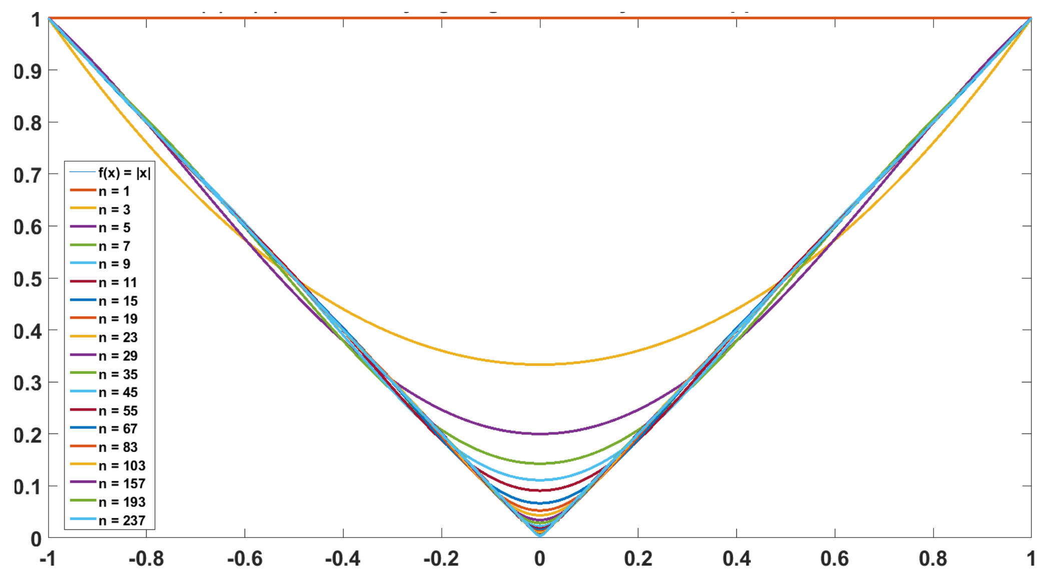


Figure 2.3: The function $f(x) = |x|$ and its fitting to varying degree polynomials

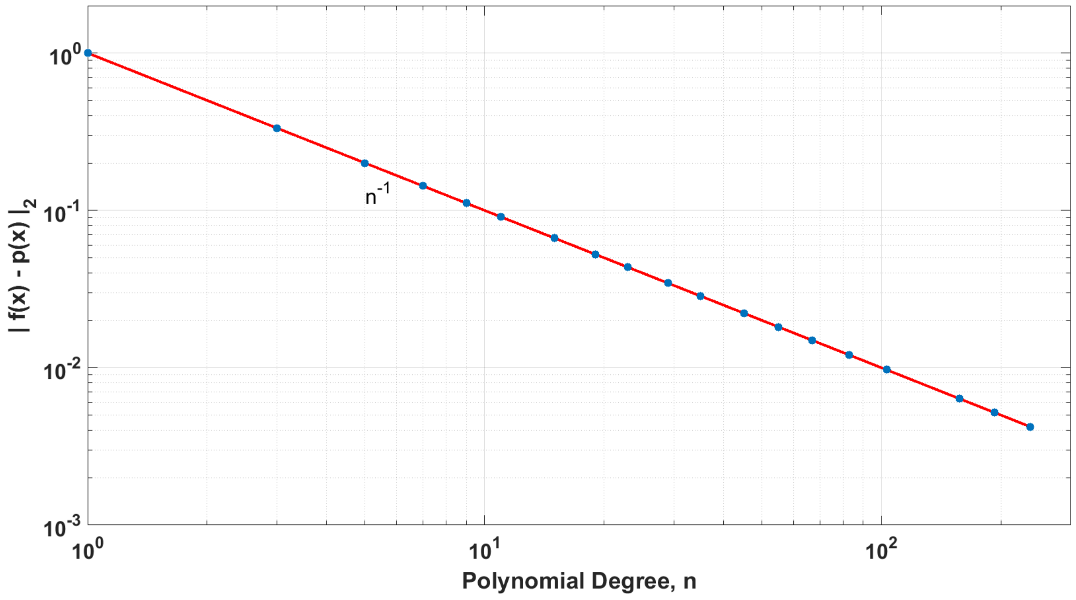


Figure 2.4: Fitting error as a function of a polynomial degree

In Figure 2.5 and Figure 2.6 we perform polynomial fitting with Chebyshev polynomials to the functions shown in Figure 2.1 and Figure 2.2, respectively. We observe that the smooth control can be perfectly approximated with an eight-degree polynomial. In the other case, we observe some oscillations still at a degree of 25 (note that the oscillations occur around the non-smooth-non-differentiable points). To demonstrate that the function indeed can be approximated with a polynomial (Weierstrass approximation theorem) we also plot the perfect approximation with a degree of 100. Note from both non-smooth cases [Figure 2.3 ($f(x) = |x|$) and Figure 2.6 (the constrained control)] the non-smooth-non-differentiable points suffers from the biggest approximation error.

This results lead to the question: can polynomial approximation be efficient also for non-differentiable functions. In this work, we devise a simple method to get similar polynomial approximation efficiency to non-smooth functions as smooth ones.

Recall that we generated the control function in Figure 2.6 by truncating the function in Figure 2.5 (same as Figure 2.1 and Figure 2.2) at the constraint value. Thus, we can simply approximate the original non-truncated control function and then ”chop” it wherever it violates some bound constraints before introducing the control into the simulator. We apply this method in this work

2.1.2 Polynomial Approximation:Non-Continuous Functions

Weierstrass approximation theorem requires a function to be continuous to guarantee the convergence of its approximation by a polynomial. What if the function is not continuous? Can we still approximate it with a polynomial?

There are several remedies to this non-continuity problem. The most common approach is, instead of approximating with a single polynomial throughout the finite time horizon, one can split the horizon into predefined segments and find the optimal polynomial for each segment.

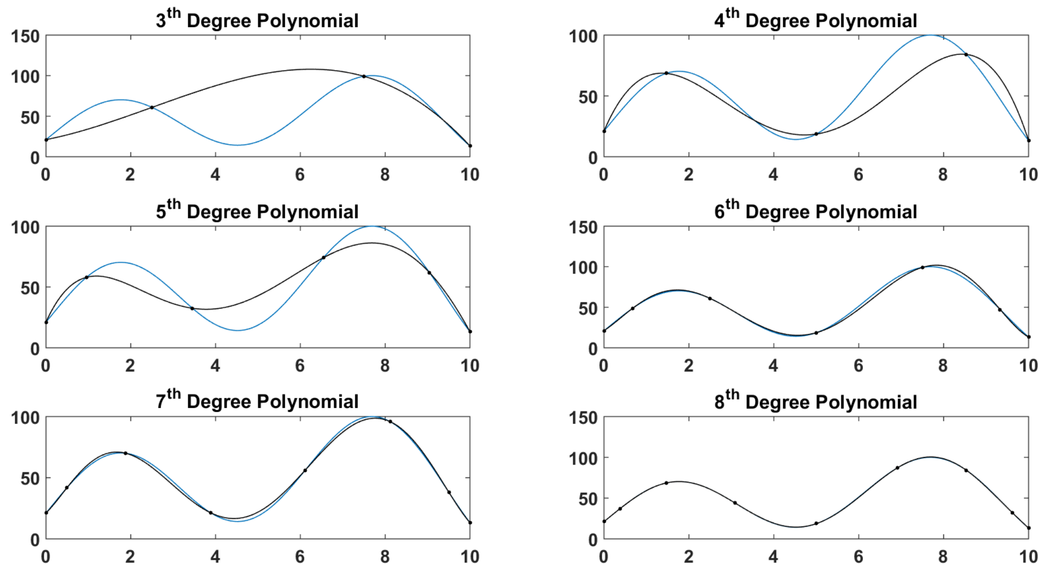


Figure 2.5: Control function from Figure 2.1 and its fitting to varying degree polynomials.

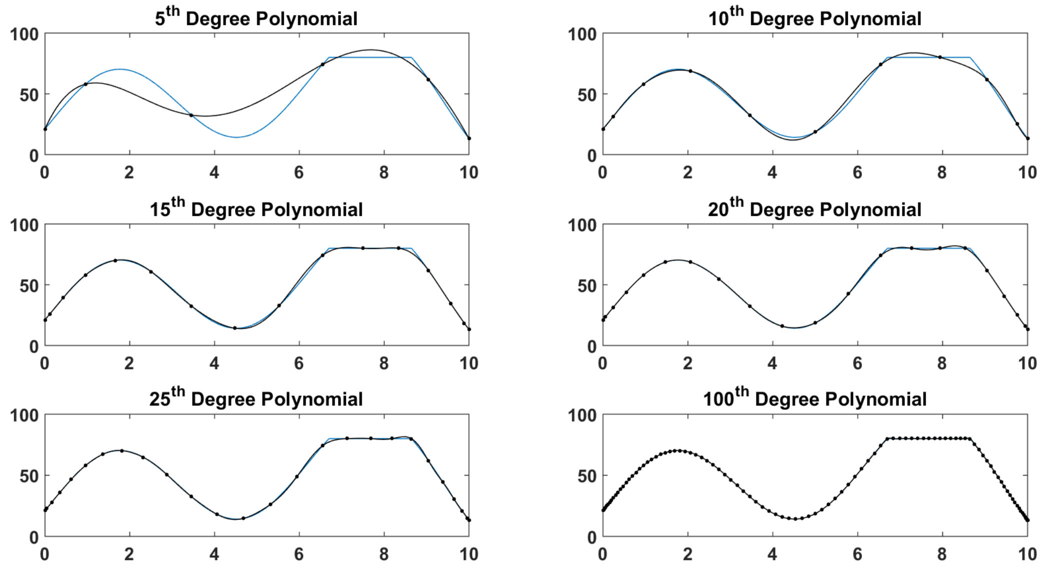


Figure 2.6: Control function from Figure 2.2 and its fitting to varying degree polynomials.

This approach of piecewise polynomials is, in fact, the most used approach in an approximation of well-valve opening level, or its representation as either flow-rate or bottom hole pressure (BHP). As we already mentioned, most applications found in the literature use zero order polynomial approximation in each segment, which translated into piecewise constant (PWC) parameterization. One of the reasons for its popularity might be that PWC requires only one variable per each interval, while higher order polynomials require more variables in each interval. Another important reason for its popularity is the ability to describe abrupt changes in the well valve position, for example, at a sudden shut-in of a well.

However, there might be cases, from a practical point of view, that using continuous approximations will still render a good job even when the original function is non-continuous. This can happen when the polynomial function exhibits a sharp slope, which when evaluated numerically it becomes non-continuous due to discretization.

2.2 Piecewise Zero Order Polynomial Approximation

As was described in section 1.1.4, the common way to parametrize the control trajectory is by piece-wise constant steps, which imply zero order piecewise polynomials. In this approach of reservoir flooding optimization, the decision variables values enter directly as input to the simulator using a surjective function $\xi : \mathbb{R}^{N_{sim}} \to \mathbb{R}^{N_u}$, where each simulation time step of the total N_{sim} simulation time steps is associated with one of the N_u control time intervals. In each interval, the control is kept constant, which render a piecewise constant approximation of the original continuous control (decision) variable.

To formalize this concept, for each well with index $j = \{1, \ldots, N_w\}$, we can obtain a piecewise constant parameterization over N_u intervals of equal duration $h = \frac{T_f}{N_u}$. The basis function are defined as:

$$\forall k \in \{0,\dots,N_{sim}\}, \forall i \in \{1,\dots,N_u\}, \forall j \in \{1,\dots,N_w\},$$
$$\Phi_{i,j}^{PWC}(t_k) = \begin{cases} t_k^0 = 1, & \text{if } (i-1)h \leq t_k \leq ih \\ 0, & \text{otherwise,} \end{cases} \tag{2.1}$$

where $t_k^0 = 1$ denotes the zero order polynomial basis function.

By applying a linear combination of basis functions $\Phi_1^{PWC}(t_k), \Phi_2^{PWC}(t_k), \dots \Phi_{N_u}^{PWC}(t_k)$, the control of each well j is approximated as follow:

$$u_j(t_k) \approx \tilde{u}_j(t_k) = \sum_{i=1}^{N_u} a_{i,j} \Phi_{i,j}^{PWC}(t_k), \tag{2.2}$$

where that equality $u_j(t_k) = \tilde{u_j}(t_k)$ holds when $N_u \to \infty$.

Again, we note $\tilde{\mathbf{u}}(t_k) = \left[\tilde{u}_1(t_k), \dots, \tilde{u}_j(t_k), \dots, \tilde{u}_{N_w}(t_k)\right]$ as the vector contains approximations for all the wells at every time step. Then, the new optimization problem will take the following form:

$$\max_{a_{i,j}} \quad \Psi(x(N_{sim}), N_{sim}) + \sum_{k=0}^{k=N_{sim}} L(x(t_k), \tilde{\mathbf{u}}(t_k), t_k) \tag{2.3a}$$

subject to

$$\dot{x}(t_k) = f_1(x(t_k)) + f_2(x(t_k))\tilde{\mathbf{u}}(t_k), \quad \forall t_k \in \mathbb{R}^{N_{sim}} \tag{2.3b}$$

$$\tilde{u}(t_k) = \sum_{i=1}^{N_u} a_{i,j} \Phi_{i,j}^{PWC}(t_k), \quad \forall j \in \{1,\dots,N_w\} \tag{2.3c}$$

$$u_{lb} \leq a_{i,j} \leq u_{ub} \tag{2.3d}$$

$$x(0) = x_0. \tag{2.3e}$$

2.3 Natural Polynomial Approximation

Following the earlier discussion we know that if a control function is continuous, it has a polynomial representation. Then, instead of optimizing the control values at different time steps, we can search for optimal coefficients to construct one polynomial function describing the control over time.

The main advantage of this approach is that the optimization dimension is decoupled from the reservoir simulation time discretization, and a fine control approximation can be optimized only with a few optimization (decision) variables. This cardinality reduction of the decision variables set gets important, as optimization algorithms might not converge in a reasonable time when the number of variables grows unbounded [45].

Let us denote $p(t_k)_j$ as the control polynomial function for well j, evaluated at discrete time steps t_k. We can construct this polynomial $p(t_k)_j$ with different families of basis functions, as we described in Eq. 1.11, as we discuss below.

The most basic polynomial representation is the natural polynomial with the simple basis function shown below:

$$\begin{gathered} \forall k \in \{0,\ldots,N_{sim}\} \in, \forall i \in \{1,\ldots,N_{nat}^{p}\}, \forall j \in \{1,\ldots,N_w\}, \\ \Phi_{i,j}^{P_{nat}}(t_k) = t_k^{i-1} \end{gathered} \tag{2.4}$$

where $i-1$ denotes the degree of each natural polynomial basis function, and $N_{nat}^{p}-1$ denotes the degree of the resulted polynomial after applying a linear combination of basis functions, as follow:

$$u_j(t_k) \approx \tilde{u}_j(t_k) = \sum_{i=1}^{N_{nat}^{p}} c_{i,j}^{nat} \Phi_{i,j}^{P_{nat}}(t_k), \tag{2.5}$$

where that equality $u_j(t_k) = \tilde{u}_j(t_k)$ holds when $N_{nat}^{p} \to \infty$, if the control is continuous.

Again, we note $\tilde{\mathbf{u}}(t_k) = \left[\tilde{u}_1(t_k),\ldots,\tilde{u}_j(t_k),\ldots,\tilde{u}_{N_{nat}^{p}}(t_k)\right]$ as the vector contains approximations for all the wells at every time step. Then, the new optimization problem will take the following form:

$$\max_{c_{i,j}^{nat}} \quad \Psi(x(N_{sim}), N_{sim}) + \sum_{k=0}^{k=N_{sim}} L(x(t_k), \tilde{\mathbf{u}}(t_k), t_k) \tag{2.6a}$$

subject to

$$\dot{x}(t_k) = f_1(x(t_k)) + f_2(x(t_k))\tilde{\mathbf{u}}(t_k), \quad \forall t_k \in \mathbb{R}^{N_{sim}} \tag{2.6b}$$

$$\tilde{u}(t_k) = \sum_{i=1}^{N_u} c_{i,j}^{nat} \Phi_{i,j}^{P_{nat}}(t_k), \quad \forall j \in \{1, \ldots, N_w\} \tag{2.6c}$$

$$-1 \leq c_{i,j}^{nat} \leq 1 \tag{2.6d}$$

$$u_{lb} \leq c_{i,j}^{nat} \leq u_{ub} \tag{2.6e}$$

$$x(0) = x_0. \tag{2.6f}$$

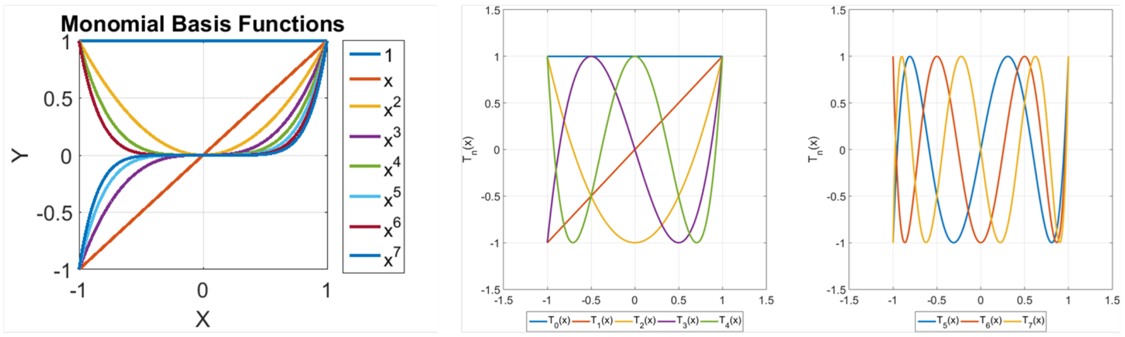
See a further discussion on the constraints shown in Eqs. **??**, in section 2.6.

2.4 Orthogonal Chebyshev Polynomial Approximation

The natural form of the polynomial might be inefficient during an optimization since all the monomials look very similar on $[0,1]$, such that large changes in the polynomial coefficients yield only small changes in the control [46].

Therefore, in this work, we consider orthogonal polynomial basis functions, which provide more variability and efficient search during the optimization. Some examples from the optimal control literature are Lagrange Polynomials [47], Legendre polynomials [25] and Chebyshev polynomials [48]. The latter is also our choice in this work.

Our choice of Chebyshev polynomials stemmed from an inspection of their image on the support $[-1,1]$. By scaling both times and controls coordinates to the box $[-1,1] \times [-1,1]$ (see later section 2.6), the Chebyshev polynomials of the first kind are promising candidates as the image is bounded and pass through the whole range of value in this box. We illustrate that in Figure 2.7 where we present the first seven natural and eight Chebyshev basis functions. Note that indeed all monomials look very similar on $[0,1]$, while all the similarity on $[-1,0]$ is between odd and even degree, separately. On the other hand, Chebyshev basis functions are significantly different from each other.



(a) **(b)** **(c)**

Figure 2.7: Comparison of Monomial basis functions and Chebyshev basis functions.

The Chebyshev polynomials of the first kind are defined as follow:

$$T_n(x) = cos(n \cdot arccos(x)), \tag{2.7}$$

which take the following recursion form:

$$T_0(x) = 1, \quad T_1(x) = x, \quad T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x). \tag{2.8}$$

These polynomials are orthogonal on the interval $-1 \leq x \leq 1$ with respect to the following weight function:

$$w(x) = \frac{1}{\sqrt{1-x^2}}, \tag{2.9}$$

such that:

$$\frac{1}{\sigma_n}\int_{-1}^{1} T_n(x)T_m(x)w(x)dx = \delta_{n,m} = \begin{cases} 0 & if n \neq m \\ 1 & otherwise, \end{cases} \tag{2.10}$$

where σ_n is a scaling factor:

$$\sigma_n = \begin{cases} \pi, & m = n = 0 \\ \frac{\pi}{2}, & m = n \neq 0 \end{cases} \tag{2.11}$$

Note that the integral bounds in Eq. 2.11 are $[-1,1]$, and thus we perform time scaling, as explained in section 2.6. Using the trigonometrical definition from Eq. 2.7, we can describe the Chebyshev control parameterization as follow:

$$\begin{gathered} \forall k \in \{0,\dots,N_{sim}\} \in, \forall i \in \{1,\dots,N^{p}_{cheb}\}, \forall j \in \{1,\dots,N_w\}, \\ \Phi^{P_{cheb}}_{i,j}(t_k) = cos(i \cdot arccos(t_k)) \end{gathered} \tag{2.12}$$

where $N^{P}_{cheb} - 1$ denotes the degree of the resulted polynomial after applying a linear combination of basis functions, as follow:

$$u_j(t_k) \approx \tilde{u}_j(t_k) = \sum_{i=1}^{N^{P}_{cheb}} c^{cheb}_{i,j} \Phi^{P_{cheb}}_{i,j}(t_k), \tag{2.13}$$

where that equality $u_j(t_k) = \tilde{u_j}(t_k)$ holds when $N^{P}_{cheb} \to \infty$, if the control is continuous.

Figure 2.8 demonstrates the control and time scaled box with three polynomials resulted from a linear combination of Chebyshev polynomials shown in Eq. 2.13. in this work ,we let the weight coefficients $c^{cheb}_{i,j}$ to lives in the range $[-1,1]$. Note that with this choice the trajectories might get slightly out of bounds to enable bang-singular control (which in this case the control is set by the simulator to the nearest boundary to the polynomial curve).

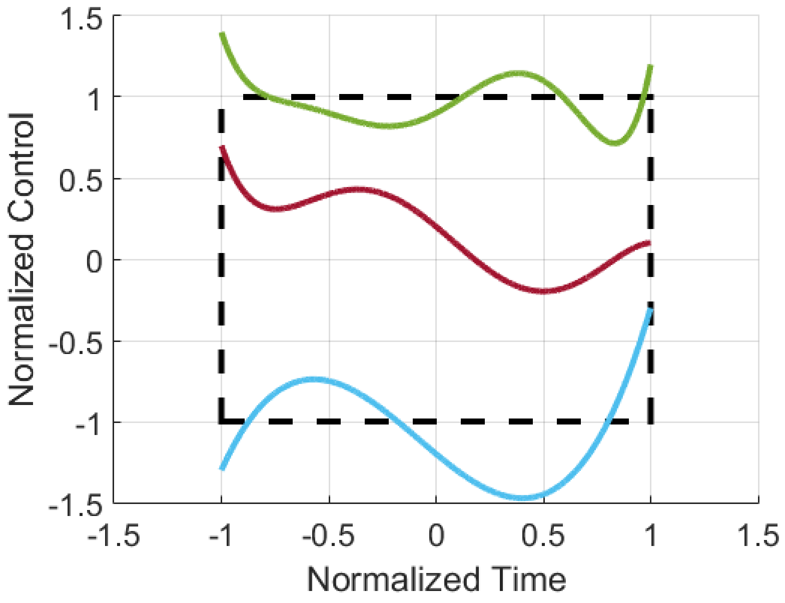


Figure 2.8: Demonstration of the control and time scaled box with three polynomials resulted from a linear combination of Chebyshev polynomials

Again, we note $\tilde{\mathbf{u}}(t_k) = \left[\tilde{u}_1(t_k), \ldots, \tilde{u}_j(t_k), \ldots, \tilde{u}_{N^P_{cheb}}(t_k)\right]$ as the vector contains approximations for all the wells at every time step. Then, the new optimization problem will take the following form:

$$\max_{c_{i,j}^{cheb}} \quad \Psi(x(N_{sim}), N_{sim}) + \sum_{k=0}^{k=N_{sim}} L(x(t_k), \tilde{\mathbf{u}}(t_k), t_k) \tag{2.14a}$$

subject to

$$\dot{x}(t_k) = f_1(x(t_k)) + f_2(x(t_k))\tilde{\mathbf{u}}(t_k), \quad \forall t_k \in \mathbb{R}^{N_{sim}} \tag{2.14b}$$

$$\tilde{u}(t_k) = \sum_{i=1}^{N_u} c_{i,j}^{cheb} \Phi_{i,j}^{P_{cheb}}(t_k), \quad \forall j \in \{1, \ldots, N_w\} \tag{2.14c}$$

$$-1 \leq c_{i,j}^{cheb} \leq 1 \tag{2.14d}$$

$$u_{lb} \leq c_{i,j}^{cheb} \leq u_{ub} \tag{2.14e}$$

$$x(0) = x_0. \tag{2.14f}$$

2.5 Piecewise Polynomial Approximation by Spline Interpolation

In the previous section, the bounds on the polynomial coefficients $c_{i.j}^{p}$ were chosen heuristically. A more natural choice for decision variables' bounds is obtained when using a polynomial-control representation using piecewise polynomial interpolation. In this case, the decision variables are bounded are the interpolation points, sampled from the original solution space. Some examples of interpolated curves used in the optimal control literature are B-spline functions [49] and Bézier curves [50]. The advantages of piecewise polynomial over a polynomial interpolation in terms of numerical stability is well-known [51, 52].

Our choice here is cubic spline interpolation with a not-a-knot condition (for details see [51]) . Cubic spline interpolation guarantees continuity up to the second derivative and thus impose a smooth control trajectory that might be beneficial from a production engineering point of view [34].

Similar to section 2.2, for each well with index $j = \{1, \ldots, N_w\}$, we shall devide the time horizon into intervals of equal duration $h = \frac{T_f}{N_z - 1}$. The basis function are defined as:

$$\forall k \in \{0, \ldots, N_{sim}\}, \forall i \in \{2, \ldots, N_z\}, \forall j \in \{1, \ldots, N_w\},$$
$$\Omega_{i,j}(\boldsymbol{z}_j, t_k) = \begin{cases} \gamma_{0,i,j} + \gamma_{1,i,j} t_k + \gamma_{2,i,j} t_k^2 + \gamma_{3,i,j} t_k^3, & \text{if } (i-1)h \leq t_k \leq ih \\ 0, & \text{otherwise,} \end{cases} \tag{2.15}$$

where $\boldsymbol{z}_j \in \mathbb{R}^{N_z}$ denotes the vector contains N_z interpolation points for each well j.

A piecewise cubic spline interpolation between the decision variables in the vector $\boldsymbol{z}_j$ determines the coefficient γ_i, with $i \in \{0, 1, 2, 3\}$, for each $\Omega_{i,j}(\boldsymbol{z}_j, t_k)$.

Then, the control of each well j is approximated as follow:

$$u_j(t_k) \approx \tilde{u}_j(t_k) = \sum_{i=1}^{N_z - 1} \Omega_{i,j}(\boldsymbol{z}_j, t_k), \tag{2.16}$$

where that equality $u_j(t_k) = \tilde{u}_j(t_k)$ holds when $N_z \to \infty$, if $u_j(t_k)$ is continuous.

Again, we note $\tilde{\mathbf{u}}(t_k) = \left[\tilde{u}_1(t_k), \ldots, \tilde{u}_j(t_k), \ldots, \tilde{u}_{N_w}(t_k)\right]$ as the vector contains approximations for all the wells at every time step. Then, the new optimization problem will take the following form:

$$\max_{\boldsymbol{z}_j} \quad \Psi(x(N_{sim}),N_{sim}) + \sum_{k=0}^{k=N_{sim}} L(x(t_k),\tilde{\mathbf{u}}(t_k),t_k) \tag{2.17a}$$

subject to

$$\dot{x}(t_k) = f_1(x(t_k)) + f_2(x(t_k))\tilde{\mathbf{u}}(t_k), \quad \forall t_k \in \mathbb{R}^{N_{sim}} \tag{2.17b}$$

$$\tilde{u}_j(t_k) = \sum_{i=1}^{N_z-1} \Omega_{i,j}(\boldsymbol{z}_j,t_k), \quad \forall j \in \{1,\dots,N_w\} \tag{2.17c}$$

$$u_{lb} \le a_{i,j} \le u_{ub} \tag{2.17d}$$

$$x(0) = x_0. \tag{2.17e}$$

Note that the decision variables, in this case, are the interpolation points inside $\boldsymbol{z}_j$ vectors. To the best of our knowledge, this is the first work which treats the waterflooding production optimization problem with the interpolation concept.

2.6 Time and Control Scaling

Scaling is a very crucial step in any optimization workflow. Here, we would like to describe how we scaled both time and control to the interval $[-1,1]$. We then give some justifications to this normalization compared to other scaling procedure shown in the literature.

The time and control scaling in this work are as follow:

$$\tilde{t} = 2 \cdot \frac{t - t_0}{t_{final} - t_0} - 1 \tag{2.18}$$

$$\tilde{\mathbf{u}} = 2 \cdot \frac{\mathbf{u} - u_{lb}}{u_{ub} - u_{lb}} - 1 \tag{2.19}$$

where, **u** again describes the control values (in units of either flow rate or pressure) at each time interval, t represents the time passed from production initiation, and $\tilde{\mathbf{u}}$ and $\tilde{t}$ are the normalized (and shifted) control and time, respectively.

A polynomial approximation method for production optimization was proposed in [53]. In this contribution, the author defined the fractional time as the ratio of the length of time counted from the beginning of operating the well to the total life of the field. Thus, the independent variable scaled from zero to one. As shown in Eqs. 2.18 and 2.19 here we took a different approach by normalizing both the operation time and the control values and shifting their scale between -1 and 1. This shifting enables to better capture multi-modality polynomial behaviors from both sides of the vertical and horizontal axes.

Also, the work in [53] bounded the variable to $[-1/n, 1/n]$ where n is the polynomial degree, to restrict the polynomial to stay within the original control bounds. This restriction allowed the control to be on the boundary at only one point in time (at the start or the end of the simulation).

However, following the already mentioned works by Sudaryanto and Yortsos [54, 55] and Zandvliet et al. [22], which show bang-bang or bang-singular solutions, the optimal control trajectory can be found on the boundary in several intervals. Therefore, we allow the polynomial to take values out of the $[-1,1]$ normalized-time-controls box, by setting the hard bounds on the polynomial coefficients to be also $[-1,1]$. In the case where the polynomial is indeed above (or below) the hard bounds, we set the control in the simulation schedule to be on the nearest boundary.

Finally, the range $[-1,1]$ is the natural choice when working with orthogonal polynomials.

3. OPTIMIZATION METHODS

This chapter discusses the optimization tools and approaches used in this work. We start with a discussion that brings the virtues and caveats of global-search and local-search methods, in the context of water-flooding optimization. Specifically, we discuss gradient-free global-search and gradient-based local-search. We describe the importance of re-parameterization, especially for global-search methods, to reduce the solution space dimension.

We then describe the algorithm used for global-search, the particle swarm optimization. Next, we describe how to obtain gradients required for local-search gradient-based optimization (GBO) with the adjoint method, where we contribute with adjoint derivation for polynomial approximations.

We continue with proposing a new work-flow for incorporating reduced order modeling with the POD-DEIM method, to be tested on different polynomial approximation parameterization approaches.

Finally, we lay our framework for gradient-based optimization under uncertainty. We proposed a multi-objective function that includes a new sub-objective function, which gives weight to the upper distribution tail. The other two sub-objectives, which are the ones used in the literature, provide weight to the expected value and the lower distribution tail.

3.1 Global versus Local Search in Production Optimization

The solution of well-control optimization problems is directly tied with the optimization solver which will be used. Historically, this optimal control problem was first solved by a local search with GBO methods, where the gradients are acquired using the adjoint method [29].

Local-search GBO methods use derivative information of the simulation to guide the search, and under certain continuity assumptions of the control profiles, as well as the prerequisite that derivatives can be calculated and are not noisy, they are designed to have a globally convergent behavior towards local stationary points, regardless of the initialization of the algorithm. A review of globally convergent gradient-based optimization methods for dynamic process optimization using interior-point methods can be found in [56]. Local gradient-based methods have the advantages of robustness and providing smooth solutions. However, they require an initial point which might affect the final local solution obtained. Two local-search alternatives to GBO, shown in the water-flooding optimization literature: approximate gradients optimization (see e.g., [57]) and local-search gradient-free optimization (GFO) (see e.g., [58]).

On the other hand, recently there is a growing interest in solving the well-control production optimization problem in waterflooding scenario using global-search gradient-free optimization (GFO) methods, usually in conjunction with additional problems such as well-placement. GFO algorithms are either stochastic and based purely on sampling [2, 59], or use surrogate approximating functions [60, 61].

GFO methods can be generically used for any simulation since they do not depend on derivatives. Efficient methods to obtain derivative values, such as the adjoint method (see later section 3.3), are not always available and its implementation requires a considerable effort. Note, that even when the adjoint method is available, derivative values may be damaged if one does not treat constraints handling carefully (see [62]).

Even a more important motivation to use GFO for well-control production optimization is to optimize well-control variables in conjunction with discrete variables. These are non-differentiable variables that render the problem as a Mixed Integer Non-Linear Programing (MINLP) problem. Such reservoir management variables may include well-placement coordinates, sequence of drilling, well type (drilled as injector, producer, or not drilled at all) and number of wells [63]. We note the recent attempt shown in [64] to solve this general oil-field development problem within a gradient-based framework.

However, the efficiency of GFO methods is limited by the number of decision variables, and often they tend to lead to non-smooth profiles for well-control optimization problems. Furthermore, although many of the GFO methods aim at obtaining a global solution, they might converge to poor solutions when the curse of dimensionality is present. In fact, in a more pragmatic sense, it is often necessary to accept good quality solutions rather than the sole (global) optimal solution.

Thus, parameterization procedures, which are both efficient and promote smoothness, are highly desirable for GFO methods. In that sense, polynomial approximation might be a good choice due to their smooth nature. Next, we discuss the effect of dimensionality reduction in more details.

3.1.1 Optimization Dimensionality Reduction

In reservoir simulation, the initial control parameterization takes place at the simulation level, where each simulation time step is associated with a correspondent control value. Then, in the optimization level, one might perform a re-parameterization to search within a smaller subspace than the original parametrized space. This procedure is essentially a reduction in the optimization search space, and hence we denote it as Optimization Dimensionality Reduction (ODR).

As we will show later, ODR plays a vital role for gradient-free optimization (GFO) methods. Below, we compare the solution spaces forms by the base parameterization and with the two approaches of polynomial approximation, namely function, and interpolation. By that, we set the stage for the results shown in the next chapter that demonstrates the importance of ODR to GFO methods.

Note that we distinguish between Model Order Reduction (MOR) and ODR. While the first aims at improving the efficiency by accelerating a single forward simulation using model reduction techniques (see examples in [1, 65, 17, 66, 67, 19]), the second accelerates the optimization by a reduced optimization representation. In other words, MOR methods reduce the cardinality of the simulation variable set, while ODR methods reduce the cardinality of the optimization variables set. We present work that combines the two approaches in section 3.4.

We address ODR for the production optimization step of Closed-Loop Reservoir Management (CLRM) [37, 68, 29] in a water flooding campaign. We consider a fixed location for all wells, and we strive to find an optimal (or near-optimal) controls-trajectory of each well. We compare three cases: fine parameterization, with the original parametrized space; polynomial approximation with Function Control Method (FCM), where we look for optimal (natural) polynomial function coefficients to describe well-controls; and Interpolation Control Method (ICM), where we look for an optimal piecewise polynomial interpolant, and the optimization variables in this case are the interpolation points (knots).

In the previous chapter, we formulated the optimization problems arises from the different parameterization approaches. Here, for the sake of clarity, we introduce a more concise formulation to describe the three box-constrained well-control optimization problems.

Optimization Problem P1: Fine Parameterization

The original parametrized box-constrained well-control optimization problem, called here $P1$, is defined as follows:

$$P1 : \max_{\mathbf{u}} \text{NPV}\left(\mathbf{u}(t)\right) \text{ subject to } \begin{cases} g(\mathbf{u}) = 0 \\ \mathbf{u} \in \mathbb{U} \end{cases}, \tag{3.1}$$

Let $N_w = (N_i + N_p)$ be the total number of injection and production wells, $j = 1, 2, \ldots, N_w$

denote the well number, and $k = 1, 2, \ldots, N_{t_{sim}}$ denote a time intervals. Each well-control variable $u_{k,j}$ can be described either by flow rates or by Bottom Hole Pressures with the corresponding upper and lower bounds. Let us define the vector of well-control variables as $\mathbf{u} = (u_{1,1}, u_{2,1}, \ldots, u_{N_{t_{sim}},1}, \ldots, u_{N_{t_{sim}},N_w})$. Then, all the possible $\mathbf{u}$ vectors forms the solution space $\mathbb{U}$:

$$\mathbb{U} = \{\mathbf{u} \in \mathbb{R}^{N_{t_{sim}} \cdot N_w} : \mathbf{u_{lb}} \leq \mathbf{u} \leq \mathbf{u_{ub}}\}, \tag{3.2}$$

where the box-constraints vectors inequality follows to definition below,

Definition 3.1.1 (Vector Relation)**.** For two vectors $\mathbf{v} = (v_0, \ldots, v_m), \mathbf{w} = (w_0, \ldots, w_m) \in \mathbb{R}^m$ we write $\mathbf{v} \leq \mathbf{w}$ if for every $0 \leq i \leq m$, $v_i \leq w_i$.

We note that an efficient log transformation can eliminate the hard bounds shown in Eq. 3.1 and Eq. 3.2, transforming the constrained optimization problem to an unconstrained problem (see [28, 69]). However, this implementation was not considered in this current work.

The measure for the size of a vector space is its dimension, which is defined as the cardinality of its basis vectors set. Since all $u_{i,j}$s are linearly independent, the dimension of $\mathbb{U}$ can be defined as follow:

$$dim(\mathbb{U}) = N_w \cdot N_{t_{sim}}. \tag{3.3}$$

Optimization Problem P2: Function Control Method

To achieve dimensionality reduction, while generating smooth and achievable control, let us assign for each well a polynomial function that describes the control. We use here natural polynomial representation (see section 2.3), where we use the short notation of $N^p = N^p_{nat}$ (recall that $N^p - 1$ denotes the polynomial degree), and denote the polynomial coefficients as $C = c^{nat}$. Eq. 3.4 shows a general form of a polynomial control function for well j:

$$\tilde{u}_j(C_0 \cdots C_n, \tilde{t_k}) = C_0 + C_1 \cdot \tilde{t_k} + C_2 \cdot \tilde{t_k}^2 + \cdots + C_n \cdot \tilde{t_k}^{N_p - 1}, \tag{3.4}$$

where $\tilde{t_k}$ is the normalized (discrete) time shown in Eq. 2.18, and $\tilde{u}_j$ is the normalized control shown in Eq. 2.19.

Let $i = 1, 2, \ldots, N_p$ denote the polynomial coefficient index, and define a new well-control variables vector as $\bar{c} = \left(C_{1,1}, C_{2,1}, \ldots, C_{N_p,1}, \ldots, C_{N_p,N_w}\right)$. In this formulation the polynomial coefficients, $C_{i,j}$, are now the optimization decision variables.

Note that we distinguish between vectors that reside in the original space $\mathbb{U}$ and those which reside in the reduced space $\mathbb{P}$ (the polynomial space). Thus, for the sake of notational clarity, we write $\bar{x}$ for elements in the space $\mathbb{R}^{N_p N_w}$ and $\mathbf{x}$ for elements in $\mathbb{R}^{N_{t_{sim}} N_w}$.

By inserting Eq. 3.4 as additional constraints into Eq. 3.1 and adding hard bounds constraints for the polynomial coefficients, we arrived at the new problem formulation, denoted here as $P2$, as shown in Eq. 3.5.

$$P2 : \max_{\bar{a}} \mathrm{NPV}\left(\mathbf{u}(\bar{a}, t)\right) \text{ subject to } \begin{cases} g(\mathbf{u}) = 0 \\ \mathbf{u}(\bar{c}, t) \in \mathbb{U} \\ \bar{c} \in \mathbb{P} \end{cases}, \tag{3.5}$$

where $\mathbb{P} = \{\bar{c} \in \mathbb{R}^{N_p N_w} : \bar{c}_{lb} \leq \bar{c} \leq \bar{c}_{ub}\}$. Where, again, the box constraints is defined according to Definition 3.1.1.

Note that the solution space $\mathbb{P}$ is a subspace of $\mathbb{U}$. The formulation in Eq. 3.5 indicates that the number of decision variables is independent of both optimization horizon time and the number of control intervals. More precisely, we reduce the solution space dimension by a factor of $\frac{N_{t_{sim}}}{N_p}$ as inferred from Eq. 3.6.

$$dim(\mathbb{P}) = N_p \cdot N_w \tag{3.6}$$

Although only hard bounds constraints were considered in this work, we mention here how one can impose smoothness constraints by adding additional constraints to P2 (eq. 3.5). Taking the derivative of the polynomial, as shown in Eq. 3.7, will enable determining the maximum allowable change in every control interval. This facilitates controlling the smoothness of the obtained solution.

$$\frac{d\tilde{u}_j(\bar{c},t)}{dt} = \sum_{i=1}^{N_p} i \cdot c_i \cdot t^{i-1}. \tag{3.7}$$

Note that for a linear case, the polynomial slope is not time-dependent. Thus, by formulating FCM as a linear control, we might limit the maximum allowable control change by only setting a hard bound on the coefficient a_1 for each well. On the contrary, for FCM with higher polynomial degrees, the constraints are time dependent and should be introduced as additional inequality constraints in problem $P2$. A quadratic control requires imposing linear constraints, and cubic and higher degree PCMs require non-linear constraints. However, in this work, we imposed only hard bound constraints, as we want to test the smoothness of the obtained control trajectories without adding linear and nonlinear constraints (if using Eq. 3.7).

Optimization Problem P3: Interpolation Control Method

The keen reader might notice that the hard bounds for the functions coefficients in FCM are determined heuristically. Thus, one can avoid the hurdle of debating which hard bounds to impose on the coefficients by using piece-wise polynomial interpolation. In this approach, which we call the Interpolation Control Method (ICM), the decision variables are control values at selected equidistant points in time. Those variables have the same bounds as the original optimization variables. Then, the control is determined based on a piecewise polynomial interpolation between these points.

In this chapter, among different piece-wise polynomial interpolation methods, we chose cubic spline interpolation, which requires continuity up to the second derivatives at the interpolation points (also called knots), and thus yields highly smooth curves. We used ”not-a-knot” cubic spline, which also requires third derivative continuity at the first and one before the last points [70, 71].

For N_z selected equidistant points per each well j, we shall denote each selected point as $z_{i,j}$, where $i = 1,2,\dots,N_z$. Let $\bar{z} = (z_{1,j}, z_{2,j}, \dots, z_{j,N_z})$, and define $P3$ as

$$P3 : \max_{\bar{z}} \text{NPV}\left(\mathbf{u}(\bar{z},t)\right) \text{ subject to } \begin{cases} g(\mathbf{u}) = 0 \\ \mathbf{u}(\bar{z},t) \in \mathbb{U} \\ \bar{z} \in \mathbb{Z} \end{cases}, \tag{3.8}$$

where $\mathbb{Z} = \{\bar{z} \in \mathbb{R}^{N_z N_w} : \bar{z}_{lb} \leq \bar{z} \leq \bar{z}_{ub}\}$.

The new solution space $\mathbb{Z}$ is again a subspace of $\mathbb{U}$, and note that we reduce the solution space dimension again by a factor of $N_t/(n+1)$ as inferred from Eq. 3.9.

$$dim(\mathbb{Z}) = N_z N_w, \tag{3.9}$$

were we reduce the solution space dimension by a factor of $\frac{N_{t_{sim}}}{N_z}$.

At this point, it should be mentioned that using either methods, FCM or ICM, the size of the optimization problem is going to remain the same even if the simulation control time-step is decreased. However, decreasing the simulation control time-step would increase the computational time of a single forward model and enable us to obtain a more detailed control scheme. Unlike several well-control parameterization approaches presented in the literature, the number of FCM and ICM parameters stay constant throughout the optimization process, which facilitates a simpler implementation. Next, we show two application examples.

3.2 Gradient-Free Optimization Using Particle Swarm Optimization

The Particle Swarm Optimization (PSO) method is a population based stochastic optimization technique developed by Dr. Eberhart and Dr. Kennedy in 1995 [59], inspired by social behavior of bird flocking or fish schooling. PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA). The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike GA, PSO has no evolution operators such as crossover and mutation.

In the PSO each particle carries memory through the iterations of his own best location it has found so far ("cognition" part) and the best location found so far in the whole swarm, or in the currently defined neighborhood ("social part"). Those best locations are the optimum particles. In PSO, the potential solutions (the particles) fly through the problem space by following the current optimum particles. The particle swarm optimization concept consists of, at each time step, changing the velocity of (accelerating) each particle toward optimum particles.

PSO has been successfully applied in many research and application areas. It is demonstrated that PSO gets better results in a faster and cheaper way compared with other methods. Another reason that PSO is attractive is that there are few parameters to adjust [72].

This work utilized the PSO implementations provided with the Global Optimization Toolbox in Matlab [73]. We used the default parameters unless we specified otherwise. The optimization algorithm is described in details in Figure 3.1 (modified version of the documentation shown in [74].).

This PSO version has a variable neighborhood size and variable inertia magnitude. In

```
1. Denote $K$ as the total number of particles in a swarm.
2. Initialize location of $K$ particles random uniformly within bounds. If there is an unbounded component, creates particles with a random uniform distribution from $initialRange$ to $itialRange$.
3. Initialize velocities of $K$ particles random uniformly within bounds. If there is an unbounded component, creates velocities with a random uniform distribution from $initialRange$ to $itialRange$.
4. Set: $bestFound = -\infty, inertiaCounter = 0, M = nFrac \cdot K, stallCounter = 0$.
5. Set $J = 1$.
6. Run in parallel for each particle $i$ with a position vector $x(i)$ of length $nVar$:
   (a) Choose a random subset $N$ of $M$ particles other than $i$.
   (b) Find $objBest(N)$, the best objective function $obj(x)$ among the neighbors.
   (c) Find $gBest(N)$, the position of the neighbor with the best objective function.
   (d) Update the velocities:
       $$v_{new} = I \cdot v_{old} + c_1 \cdot rand_1 \cdot (pBest - x) + c_2 \cdot rand_2 \cdot (gBest - x),$$
       where:
       i. $v_{new}$ and $v_{old}$ are the new and old particle $i$ velocities, respectively.
       ii. $I$ is the inertia component.
       iii. $c_1$ and $c_2$ are the self-adjustment and social-adjustment components, respectively.
       iv. $rand1$ and $rand2$ are uniformly (0,1) distributed random vectors of length $nVars$.
       v. $pBest$ is the location of the best objective function that particle $i$ has previously found.
       vi. $gBest$ is the location of the best objective function found in the current neighborhood.
   (e) Update the position: $x(i) = x(i) + v(i)$.
   (f) If any component of $x(i)$ is outside a bound, set it equal to that bound.
   (g) Evaluate the objective function $obj(i) = obj(x(i))$. Set $Flag(i) = False$.
   (h) If $obj(i) > obj(pBest)$, then update $pBest = x$.
   (i) If $obj(i) > bestFound$, then set $bestFound = obj(i), bestPosition = x(i)$ and $Flag(i) = True$.
   (j) Update inertia $I$:
       i. If $Flag(i) = True$:
          A. Set $inertiaCounter = max(0, inertiaCounter - 1)$.
          B. If $inertiaCounter < 2$, then set $I = I \cdot 2$.
          C. If $inertiaCounter > 5$ then set $I = I/2$.
          D. Ensure that I is within $inertiaRange$.
       ii. If $Flag(i) = False$: $inertiaCounter = inertiaCounter + 1$
   (k) Update neighborhood size, $M$:
       i. If $Flag(i) = True$: $M = nFracK$
       ii. If $Flag(i) = False$: $M = min(M + nFrac \cdot K, K)$.
7. If relative change in the best objective function value is less than specified tolerance set $stallCounter = stallCounter + 1$.
8. If $stallCounter < stallIterLimit$ (or not reaching other termination criteria) iterate back to 6.
9. Else, return $bestFound$ and $bestPosition$.
```

Figure 3.1: Pseudo-code for PSO maximization with a dynamic neighborhood (adopted from Matlab documantaion).

each iteration, each particle has a location (defined by its vector values) and a velocity. The velocity updating equation is the core of the PSO algorithm. The equation updates the new velocity, which carries each particle from one location (a feasible solution) to a neighboring location (a neighbor feasible solution).

As can be seen in the pseudo-code, the algorithm combines meta-heuristic features with local search approach. A particle is "attracted" to the best solution found by him and to the best solution found by his current dynamic-neighborhood, M. Thus, the local search here can be viewed as "attraction" to the good quality solutions.

However, at the same time, the algorithm introduces few variables that add stochastic search which might explore meta-heuristically the new areas. These variables are $rand_1$ and $rand_2$, that add random effect to each source of attraction.

Additional global exploration is achieved by adaptively changing the inertia parameter, I, and the dynamic neighborhood M, according to the optimization process. For example, once a better solution is found, the neighborhood is set to its minimum size. Thus, a particle can be attracted to a direction, which might be very different than a direction achieved, if the total swarm was considered as the neighborhood. This dynamical neighborhood approach was not presented in several previous works that utilized PSO to solve the problem under consideration.

Construction heuristic refers to the first ensemble of guesses that constitute the first location of each particle in the swarm, and it is the zeroth iteration of the optimization process. The construction heuristic of the particle swarm algorithm starts by creating the initial set of solutions (particles) and assigning each one of them initial velocities (steps 2 and 3 in Figure 3.1).

The algorithm has the capability of getting user supplied particles and creates the rest of the particles at random uniformly within bounds. This provides a sort of greedy randomized construction capability. However, in the current work, all the particles have been randomly initialized, where a greedy creation of particles can be introduced in future work. Greedy criteria can be based, for example, on engineering and geological reasoning.

Let us note the number of design variables as *nvars*, their upper bound as *ub* and lower bound as *lb*. The random initialization creates for particle i a position $x(i)$, which is a row vector with *nvars* elements. If any component of x is outside a bound, algorithm set it equal to that bound.

Table 3.1: Parameter values used in both examples for the Particle Swarm Optimization algorithm

	Example 1	**Example 2**
Function Tolerance	10^{-6}	10^{-4}
stallIterLimit	20	12
K	30	
nFrac	0.25	
inertiaRange	(0.1,1.1)	
c1, c2	1.49	

3.3 Adjoint Method for Polynomial Approximation

One of the key steps in designing a gradient-based optimization workflow is to assure efficient calculation of the objective function's gradients with respect to the control variables. Gradient computation is often not feasible analytically when the objective function is related to the control variable through a numerical simulation, and thus gradients are evaluated numerically.

The adjoint method is probably the most efficient method to obtain gradients numerically. The common example for its efficiency is by comparison to the finite-difference method: while the finite difference requires $n+1$ simulation runs to evaluate gradients with respect to n control variables, the adjoint only requires one forward simulation run and one backward run.

Many published works demonstrated applications of the adjoint method for the water-flooding production optimization problem [37, 29, 5, 22, 1, 27]. As the adjoint method gained popularity, it was only natural that adjoint equations have been implemented in several simulators at recent years. Eclipse 300 [75], MRST [76] and AD-GPRS [29, 30] are examples for simulators with an adjoint capability, that enables the user to efficiently obtain gradients of defined objective function with respect to control variables.

To the best of our knowledge, all previous works performed the adjoint method only with respect to zero order polynomial parameterization, i.e., to the traditional piece-wise constant (PWC) control steps. Usually, the gradients are provided for each forward simulation time step. Then, in the simple case of PWC, one simply sums up all the gradients values of all simulation time steps that belongs to the same control time step.

In this work, we explore adjoint-gradients computation with respect to higher degree polynomials, and, in general, to an arbitrary parameterization by a linear combination of basis functions. We show how to obtain the gradients with respect to the linear combination coefficients, using the adjoint simulator output.

We start with writing the optimization problem shown in Eq. 1.10 in a discrete form, taking into consideration, for now, only the initial values and PDE constraints. For better exposition, we assume no terminal cost function $\Psi(x_{N_{sim}})$. We arrange all variables into one side to form a residual, and by using discrete time steps n, the problem takes the following form:

$$\max_{u_n} \quad J = \sum_{n=0}^{N_{sim}} L_n(\mathbf{x}_n, \mathbf{u}_n) \tag{3.10a}$$

subject to

$$r(\mathbf{x}_n, \mathbf{x}_{n-1}, \mathbf{u}_n) = 0, \tag{3.10b}$$

Now, we can adjoin all constraints into an augmented objective function, using the Lagrange multiplier method:

$$J_{aug} = \sum_{n=0}^{N_{sim}} \big(L_n(\mathbf{x}_n, \mathbf{u}_n) + \boldsymbol{\lambda}_n^T r(\mathbf{x}_n, \mathbf{x}_{n-1}, \mathbf{u}_n) \big) \tag{3.11}$$

A necessary condition for optimality for both the objective function in Eq. 3.10 and the augmented function in Eq. 3.11, is that the first variation of J_{aug} is zero ($\delta J_{aug} = 0$) [77].

Using the chain rule, and after some variables grouping, δJ_{aug} can be written as:

$$\begin{aligned}
&\delta J_{aug}\left(\mathbf{x}_n, \mathbf{x}_{n-1}, \mathbf{u}_n, \boldsymbol{\lambda}_{n+1}, \boldsymbol{\lambda}_n\right) = \\
&\sum_{n=0}^{N_{sim}} \left(\frac{\partial L(\mathbf{x}_n, \mathbf{u}_n)}{\partial \mathbf{x}_n} + \boldsymbol{\lambda}_n^T \frac{\partial r(\mathbf{x}_n, \mathbf{x}_{n-1}, \mathbf{u}_n)}{\partial \mathbf{x}_n} + \boldsymbol{\lambda}_{n+1}^T \frac{\partial r(\mathbf{x}_{n+1}, \mathbf{x}_n, \mathbf{u}_{n+1})}{\partial \mathbf{x}_n} \right) \delta \mathbf{x}_n + \\
&\sum_{n=0}^{N_{sim}} \left(\frac{\partial L(\mathbf{x}_n, \mathbf{u}_n)}{\partial \mathbf{u}_n} + \boldsymbol{\lambda}_{n+1}^T \frac{\partial r(\mathbf{x}_{n+1}, \mathbf{x}_n, \mathbf{u}_{n+1})}{\partial \mathbf{u}_n} \right) \delta \mathbf{u}_n + \\
&\sum_{n=0}^{N_{sim}} r(\mathbf{x}_{n+1}, \mathbf{x}_n, \mathbf{u}_{n+1}) \delta \boldsymbol{\lambda}_{n+1}^T,
\end{aligned} \tag{3.12}$$

Note that since the variations of $\mathbf{x}_n, \mathbf{u}_n,$ and $\boldsymbol{\lambda}_n$ are independent of one another, each of the three summation terms must vanish for optimality [77]. The last summation term is zero by the residual definition of the reservoir simulation equations. Thus we left with two summation terms. As we further explain below, the first remaining term is used to obtain the Lagrange multipliers, and the second is immediately utilized to compute the required gradients.

For gradient-based optimization, we need the gradients of the objective function with respect to the control variables u_n. This information is found inside the second summation term in Eq. 3.12 that describes the variation of the objective function as a result of the perturbation δu_n. Specifically, the required gradients are as follows:

$$\frac{dJ}{d\mathbf{u}_n} = \frac{dJ_{aug}}{d\mathbf{u}_n} = \frac{\partial L(\mathbf{x}_n, \mathbf{u}_n)}{\partial \mathbf{u}_n} + \boldsymbol{\lambda}_{n+1}^T \frac{\partial r(\mathbf{x}_{n+1}, \mathbf{x}_n, \mathbf{u}_{n+1})}{\partial \mathbf{u}_n} \tag{3.13}$$

The two Jacobian matrices $\frac{\partial L}{\partial \mathbf{u}}$ and $\frac{\partial r}{\partial \mathbf{u}}$ (written here without subscripts), can be calculated either analytically [78] or by means of automatic differentiation [79][1].

The vector of Lagrange multipliers $\boldsymbol{\lambda}_{n+1}^T$ is obtained form the adjoint equations. These equation are resulted from the first summation term in Eq. 3.12. We can choose $\boldsymbol{\lambda}_{n+1}^T$ in such a way that the entire first summation term will be equal to zero as required by the optimality condition. Thus, for each time step n, we solve:

$$\frac{\partial L(\mathbf{x}_n, \mathbf{u}_n)}{\partial \mathbf{x}_n} + \boldsymbol{\lambda}_n^T \frac{\partial r(\mathbf{x}_n, \mathbf{x}_{n-1}, \mathbf{u}_n)}{\partial \mathbf{x}_n} + \boldsymbol{\lambda}_{n+1}^T \frac{\partial r(\mathbf{x}_{n+1}, \mathbf{x}_n, \mathbf{u}_{n+1})}{\partial \mathbf{x}_n} = 0. \tag{3.14}$$

This is a linear system of equations solved backward for $\boldsymbol{\lambda}_n^T$:

$$\boldsymbol{\lambda}_n^T = -\left(\frac{\partial L(\mathbf{x}_n, \mathbf{u}_n)}{\partial \mathbf{x}_n} + \boldsymbol{\lambda}_{n+1}^T \frac{\partial r(\mathbf{x}_{n+1}, \mathbf{x}_n, \mathbf{u}_{n+1})}{\partial \mathbf{x}_n}\right) \left(\frac{\partial r(\mathbf{x}_n, \mathbf{x}_{n-1}, \mathbf{u}_n)}{\partial \mathbf{x}_n}\right)^{-1}, \tag{3.15}$$

where we assume $\boldsymbol{\lambda}_{N_{sim}}^T = 0$ at the last time step.

[1]In MRST, the well controls appear as linear combinations of states, and thus, the objective J is a function of states only, and the partial derivative $\frac{\partial r}{\partial \mathbf{u}}$ becomes zero except for a small identity block resulted from the control equations [79].

We are now interested in finding the gradients with respect to the coefficients of the linear combination parameterization. This can easily be done by the chain rule as we describe below.

First, we shall define the parameterization for control u of each well j at each time step n, as a linear combination of N_p basis functions Φ:

$$\forall j \in \{1,\ldots,N_w\}, \forall n \in \{0,\ldots,N_{sim}\}, \quad u_{j,n} = \begin{cases} u_j^{lb}, & \sum_{i=0}^{N_p} w_{i,j}\Phi_{i,j,n} \leq u_j^{lb} \\ u_j^{ub}, & \sum_{i=0}^{N_p} w_{i,j}\Phi_{i,j,n} \geq u_j^{ub} \\ \sum_{i=0}^{N_p} w_{i,j}\Phi_{i,j,n}, & \text{otherwise,} \end{cases} \tag{3.16}$$

Taking the derivative of $u_{j,n}$ with respect to $w_{i,j}$, we get:

$$\forall i \in \{1,\ldots,N_p\}, \forall j \in \{1,\ldots,N_w\}, \forall n \in \{0,\ldots,N_{sim}\},$$
$$\left(\frac{\partial u_j}{\partial w_{i,j}}\right)_n = \begin{cases} 0 & \sum_{i=0}^{N_p} w_{i,j}\Phi_{i,j,n} \leq u_j^{lb} \\ 0 & \sum_{i=0}^{N_p} w_{i,j}\Phi_{i,j,n} \geq u_j^{ub} \\ \Phi_{i,j,n} & \text{otherwise,} \end{cases} \tag{3.17}$$

.

Summing up all contributions from all time steps and by using the chain rule we get:

$$\frac{dJ}{dw_{i,j}} = \sum_{n=0}^{N_{sim}} \left(\frac{dJ}{du_j}\frac{\partial u_j}{\partial w_{i,j}}\right)_n \tag{3.18}$$

Thus, insertion Eqs. 3.13 and 3.17 into Eq. 3.18 yields the desired gradients. From Eq. 3.17 we can see that at time steps where the control is found on the bounds there is no contribution to the total gradient value. Note that any scaling or normalization of the time domain should be performed to the gradients as well.

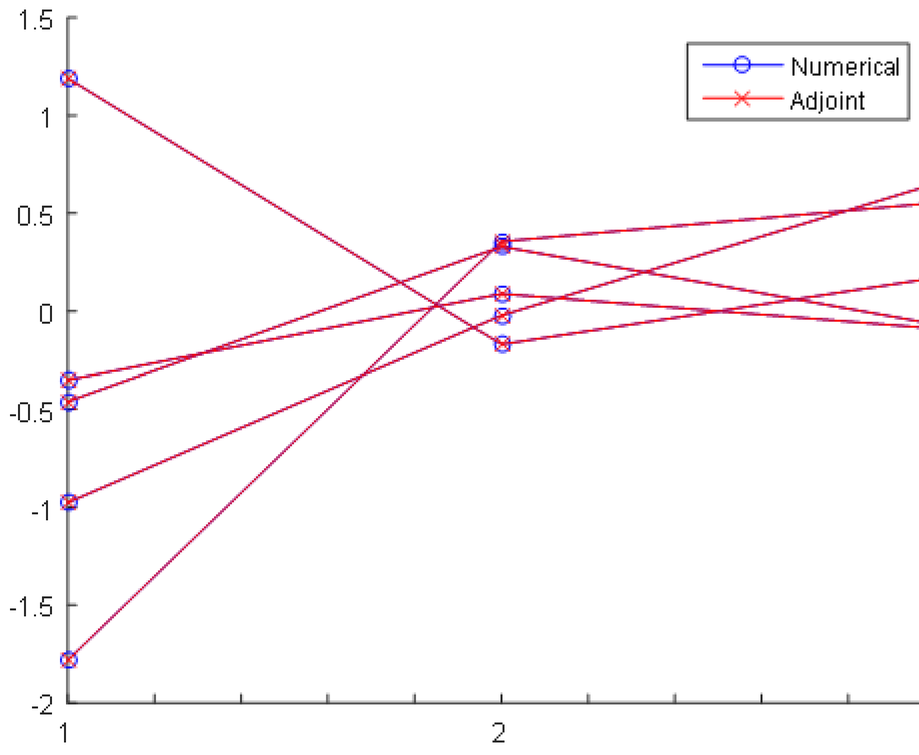


Figure 3.2: Implementation test for second order Chebyshev polynomial approximation (three coefficients per well): Adjoint gradients are compared against numerical finite-difference gradients. Each curve represents a different well. Horizontal axis: the three coefficients. Vertical axis: gradients magnitude. The test was performed on the 5-spot case presented in section 1.2.

Figure 3.2 shows a test of our implementation where the adjoint gradients for Chebyshev polynomial approximation are compared against numerical finite-difference gradients.

3.4 Model Order Reduction

In this work, in addition to fine-scale simulation, we also perform an optimization with reduced order modeling. We use the POD-DEIM technique for that propose and the detailed math behind it is provided in Appendix A.

In short, the fist step of this method requires a construction of a snapshot matrix. This matrix is generated by the outputs of training simulation runs. Then a *SVD* is applied to this matrix to obtain the POD basis functions. The solution is then approximated in terms of a linear combination of the POD basis functions. However, when POD is applied to a nonlinear problem, the evaluation of the nonlinear terms require projecting back to the fine scale solution with similar computational cost as the original system. Thus, Discrete Empirical Interpolation Method is used where one constructs another subspace for reducing the nonlinear function evaluations [80].

Local search optimization methods are usually used when coupled with POD-based approaches, since the vector of controls is expected to differ relatively slightly between consecutive iterations, and thus the POD-based approach can be expected to be accurate for several iterations. By contrast, consecutive iterations of global stochastic search methods can differ significantly, such that POD-based methods may not maintain accuracy [37]. However, in this work, we will show an example in which harnessing the stochastic nature of a global search to train the model leads to accurate results.

3.4.1 Model Order Reduction with On-line Training

The accuracy of the POD-DEIM procedure during the well control optimization process is dependent on the quality of the snapshot matrix and hence on the training BHP profiles. Thus, the training scheme that produces the snapshot matrix is a key factor to the success of reduced order modeling. A common practice is that well conditions in training should correspond as closely as possible to those encountered during the optimization [15, 81]. Our approach here is to start the optimization with fine-scale simulation runs and use them as an online training set. During the optimization process within the PSO framework, we spend a good amount of effort simulating different particles in parallel. Thus, assuming the initial swarm profiles will adequately represent the dynamics of the

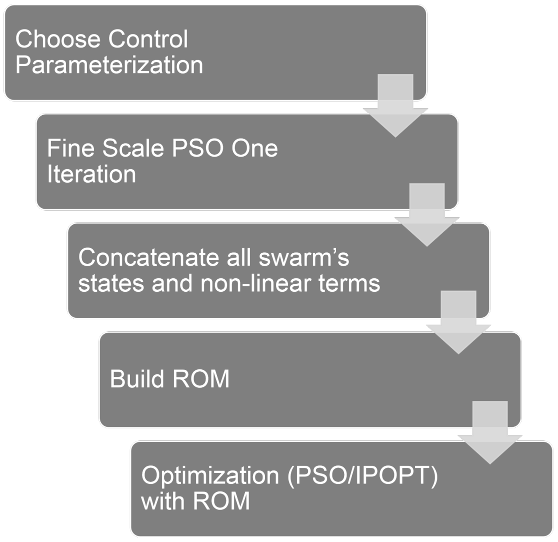

Choose Control Parameterization
Fine Scale PSO One Iteration
Concatenate all swarm's states and non-linear terms
Build ROM
Optimization (PSO/IPOPT) with ROM

Figure 3.3: Work-flow for optimization-online training of POD-DEIM.

reservoir during the subsequent optimization iterations, we use the fine-scale solutions of these particles from the first optimization iteration to construct the snapshot matrix.

In more detail, in the gradient-free approach using PSO, all simulations state outputs of the initial random swarm are concatenated into one snapshot matrix. After constructing the POD-DEIM basis, the optimization proceeds to the second iteration of the reduced order model. Similarly, in the gradient-based approach, we used an ensemble of random BHP profiles for training, where the profile which produced the best NPV was chosen as the initial guess for the optimization procedure. By doing so, we utilize the training effort also to advance the optimization process.

Figure 3.3 describes this work-flow. From the first block in this flowchart, we can see that we investigate this method while comparing its performance for different optimization parameterization techniques. Our contribution here is two-fold. First, we propose a workflow that uses the global-optimization technique (PSO), while all previously published work used local search methods for POD-based MOR. Secondly, and more importantly, we show that high order polynomial approximation provides more information in the snapshot matrix and thus both reduced order modeling and optimization are more accurate, which lead to higher obtain NPV.

3.5 Optimization Under Uncertainty

So far, we considered only a single reservoir model to maximize the NPV of a water flooding campaign. However, the true specification of the reservoir is never known, and a set of geological realizations is the common approach to depicting the geological uncertainty. Each geological realization might consist of a different permeability or porosity field, or varying structural features and facies distribution.

Hence, Eq. 1.8 that describes the NPV objective function for the deterministic case, can now be restated as a function of model parameters field θ_i, e.g., permeability or porosity, as follow:

$$J(u,\theta_i) = NPV(u,\theta_i) \tag{3.19}$$

There are several families of methods for optimization under uncertainty, and one can find different classifications by different optimization scholars. For examples, Grossman in his review paper [82], suggests the two categories of robust optimization and stochastic programming, while Sahinidis [83] suggests a categorization into stochastic programming, fuzzy programing, and stochastic dynamic programming, where he classifies robust and probability programming as sub-categories under stochastic programing[2].

[2]The stochastic programming approach models uncertainty through discrete or continuous probability functions. On the other hand, fuzzy programming describes random parameters and constraints as fuzzy numbers or fuzzy sets, respectively. In stochastic dynamic programming, one solves a series of sub-problems in time, where the random parameters follow some distribution that depends only on the current subproblem.

According to Sahinidis, stochastic programming includes recourse (or two-stage) models, robust stochastic programming, and probabilistic (or chance-constrained) models. Recourse models optimize an expected recourse function by assigning first and second stages variables, where the values of the first are determined before uncertainty is revealed and the values of the second acts as corrective measures after the first stage uncertainty has been revealed. Robust optimization methods minimize risks in addition to optimizing the expected value. In probabilistic methods, the constraints should be satisfied up to a certain probability value.

Thus, it is only natural that the confusion in terminology has diffused into the petroleum literature. For example, several published works define robust optimization as merely optimizing the expected value, e.g. [84, 85], while in [86] robust optimization refers to optimizing the worst case scenario. The latter is the more common definition outside the oil-reservoir engineering literature. Ben-Tal et al., in the book "Robust Optimization" [87], describe an optimization philosophy that guarantees feasibility to all scenarios. As a result, robust optimization model returns a solution that is optimal for the worst-case scenario, and hence it is equivalent to optimizing the worst-case (max-min problem, for maximization, or min-max problem, for minimization). However, considering the worst case is often overly conservative.

In oil-reservoir water flooding optimization, it is now customary to solve the problem with a scenario-based multi-objective optimization approach, where the expected value is optimized along with additional objective function that minimizes the risk. The additional function is required since optimizing only the average value does not reduce associated risk and uncertainty. Several such risk-averse functions have been tested for waterflooding optimization problems, such as the standard deviation [88, 89, 90], semi-variance [86], worst-case objective value [90, 89, 86] and conditional value at risk (CVaR) [91, 90, 89, 86]. All these works consistently demonstrated that CVaR serves as a better secondary objective function to reduce the risk.

For example, minimizing standard deviation as the secondary objective often leads to a significant NPV reduction for the best cases in a distribution (found in the upper tail of a distribution). On the other hand, CVaR optimizes the lower tail of the objective function while less compromising the upper-tail values. Figure 3.4 depicts the concept of CVaR and VaR measures, which are explained later in more details in section 3.5.3.

However, some deterioration in the upper values is often observed even with CVaR. Thus, in this work, we propose an additional risk measure, complimentary to CVaR, which serves to maximize the upper tail and to improve the best cases values further. The suggested function is a mirror image of CVaR, and we denote it as Conditional Value at Success(CVaS).

In addition to these terms, CVaR, Mean and CVaS, we investigate the results with statistical measures, which are more used in the petroleum industry, namely, P10, P50, and P90. We provide more details below.

3.5.1 Multi-Objective Optimization

Dealing with several objectives turns the problem into a multi-objective optimization problem, which can be treated as a simple linear combination of the objectives, or as a two-stage, primary-secondary, optimization approach [90]. The linear combination of objectives, which is the chosen method in this work, can be define as follow:

$$J_{tot} = \sum_{k=1}^{N_{obj}} \eta_k J_k(u), \tag{3.20}$$

where N_{obj} is the number of objective functions and η_k is the predefined weight per each objective j_k. Often, $N_{obj} = 2$, where two competitive objectives form a solution that lies on the so-called *pareto front*. For example, expected value and CVaR form such a front (see e.g., [89]).

3.5.2 Expected Value

The objective function of the expected value can be defined as follow:

$$J_E(u) = \sum_{i=1}^{N_r} p_i J(u, \theta_i), \tag{3.21}$$

where p_i is the probability associated with a realization scenario i. Assuming uniform distribution of the geological model realizations, we get:

$$J_E(u) = \frac{1}{N_r}\sum_{i=1}^{N_r} J(u,\theta_i). \tag{3.22}$$

Recalling that u is a function of parameters w_j, (see Eq. 3.18), we can compute the expected value of the gradient in a similar manner:

$$\frac{dJ_E}{dw_j} = \frac{1}{N_r}\sum_{i=1}^{N_r} \frac{dJ(u,\theta_i)}{dw_j}. \tag{3.23}$$

3.5.3 Conditional Value-at-Risk

Conditional Value-at-Risk (CVaR) introduced first in [92], is a popular measure for managing financial risk. For a given percentage α, CVaR indicates the probability-weighted avarage of the $\alpha-$tail of worst cases of a distribution, while Value at Risk (VaR) indicates the value of the α percentile, as illustrated in Figure 3.4.

Formally, for a random variable X with cumulative gain distribution function $F_X(Z) = P\{X \leq z\}$, the VaR and CVaR of X with $\alpha \in]0,1[$ are given as[3]:

$$VaR_\alpha(X) = max\{z|F_X(z) \leq \alpha\}, \tag{3.24}$$

$$CVaR_\alpha(X) = \mathbb{E}[X|X \leq VaR_\alpha(X)]. \tag{3.25}$$

CVaR has superior mathematical properties over VaR and it is is a coherent risk measure, while VaR is not (see [92, 93]).

Practically, if we assume again uniform distribution of the geological model realizations, we can compute the objective CVaR by sorting all $J(u,\theta_i)$ such that $J(u,\theta_1) \leq$

[3]The definition in Eq.3.25 for CVaR holds as long as α does not "split" scenarios. That, is $\alpha N_r \in \mathbb{N}$. If αN_r is not a natural number, the definition in Eq.3.25 corresponds to $CVaR^+_\alpha$, while $CVaR_\alpha$ is defined differently [93]. We assured that α does not split scenarios in all cases tested in this work.

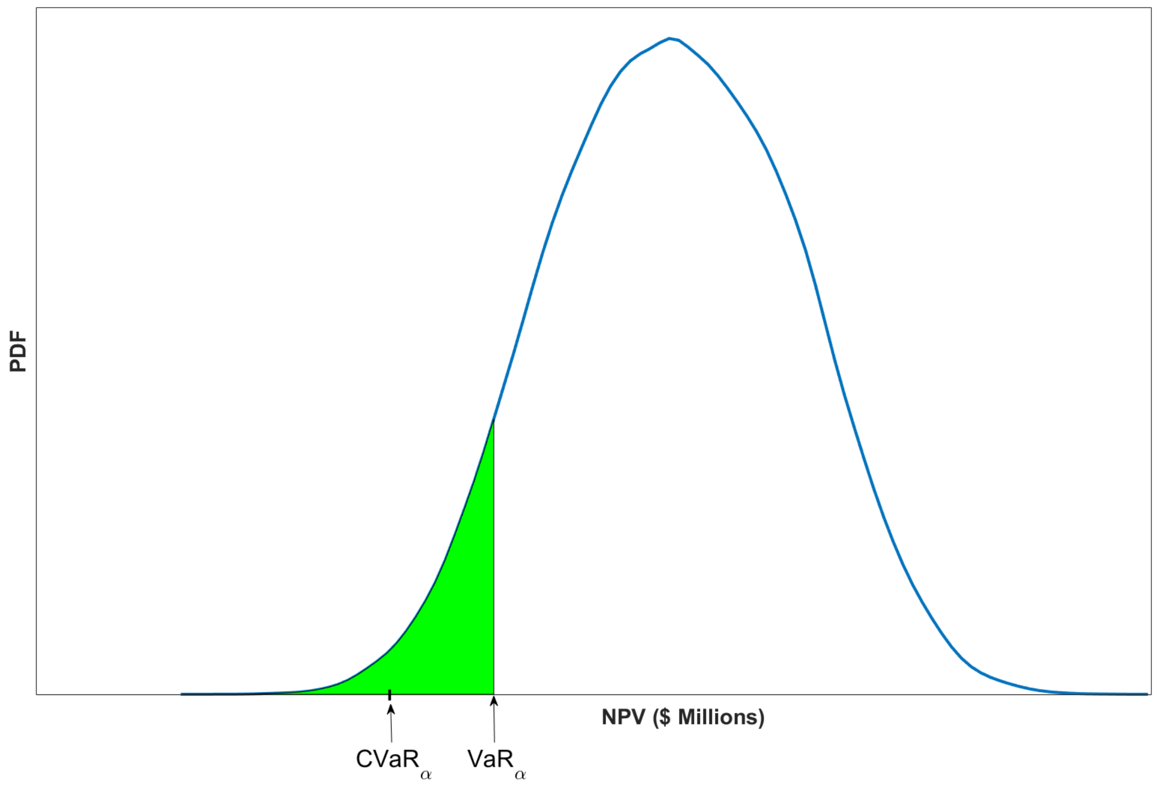


Figure 3.4: Illustration of VaR and CVaR for a given distribution. Realizations with cumulative distribution (shaded area) that falls below α constitute the distribution lower tail. VaR is the highest NPV amongst these realizations, while CVaR is their weighted average.

$J(u,\theta_2) \leq \cdots \leq J(u,\theta_{N_r})$, and compute the average for first αN_r objectives, as follow:

$$J_{CVaR_\alpha}(u) = \frac{1}{\alpha N_r} \sum_{i=1}^{\alpha N_r} J(u,\theta_i). \tag{3.26}$$

We can define the gradients in a similar manner:

$$\frac{dJ_{CVaR_\alpha}}{dw_j} = \frac{1}{\alpha N_r} \sum_{i=1}^{\alpha N_r} \frac{dJ(u,\theta_i)}{dw_j}. \tag{3.27}$$

3.5.4 Conditional Value-at-Success

The objective defined in Eq.3.26 strives to improve the worst cases, while the objective defined in Eq.3.22 addresses the mean value. Next, we propose an additional measure that also considers the best cases. Thus, we cal our proposed measure Conditional Value at Success (CVaS), that accounts for the success in fulfilling the tail of optimistic cases:

Definition 3.5.1 (VaS)**.** For a random variable X with a cumulative gain distribution function $F_X(Z) = P\{X \leq z\}$, and for $\beta \in]0,1[$, the VaS_β is the value

$$\zeta_\beta(X) = min\{\zeta | F_X(\zeta) \geq \beta\}. \tag{3.28}$$

Definition 3.5.2 (CVaS)**.** For a random variable X with a value-at risk $\zeta_\beta(X)$, the $CVaS_\beta$ is the value

$$\phi_\beta(X) = \mathbb{E}[X | X \geq VaR_\beta(X)]. \tag{3.29}$$

Since we assume uniform distribution of the geological model realizations, the expected value is the simple arithmetic mean. Thus, symmetrically to CVaR, let us sort all $J(u,\theta_i)$ such that $J(u,\theta_1) \geq J(u,\theta_2) \geq \cdots \geq J(u,\theta_{N_r})$, and compute the average for last βN_r realizations, as follow:

$$J_{CVaS_\beta}(u) = \frac{1}{\beta N_r} \sum_{i=1}^{\beta N_r} J(u,\theta_i). \tag{3.30}$$

where β is the percentage defines the lower bound of the upper tail.

We can define the gradients in a similar manner:

$$\frac{dJ_{CVaS_\beta}}{dw_j} = \frac{1}{\beta N_r} \sum_{i=1}^{\beta N_r} \frac{dJ(u,\theta_i)}{dw_j}. \tag{3.31}$$

3.5.5 Total Objective Function

Using Eqs. 3.20, 3.22, 3.26 and 3.30, we can write the following total objective function:

$$J_{tot} = \eta_1 J_E(u) + \eta_2 J_{CVaR_\alpha}(u) + \eta_3 J_{CVaS_\beta}(u), \tag{3.32}$$

Taking the derivative of Eq.3.32, and using Eqs. 3.23, 3.27 and 3.31, we can write the gradient of the total objective as follow:

$$\frac{dJ_{tot}}{dw_j} = \eta_1 \frac{dJ_E}{dw_j} + \eta_2 \frac{dJ_{CVaR_\alpha}}{dw_j} + \eta_3 \frac{dJ_{CVaS_\beta}}{dw_j}. \tag{3.33}$$

Note that only the ratio $\eta_1 : \eta_2 : \eta_3$ dictates the obtained solution, and not the absolute value of each weight η_i. For convenience, one can formulate the weights as a convex combination:

$$\eta_1 + \eta_2 + \eta_3 = 1, \quad \eta_i \geq 0, \tag{3.34}$$

and reformulate Equation 3.32, as follow:

$$J_{tot} = \eta_1 J_E(u) + \eta_2 J_{CVaR_\alpha}(u) + (1 - \eta_1 - \eta_2) J_{CVaS_\beta}(u). \tag{3.35}$$

Figure 3.5 illustrate the different measure found in the objective function 3.35.

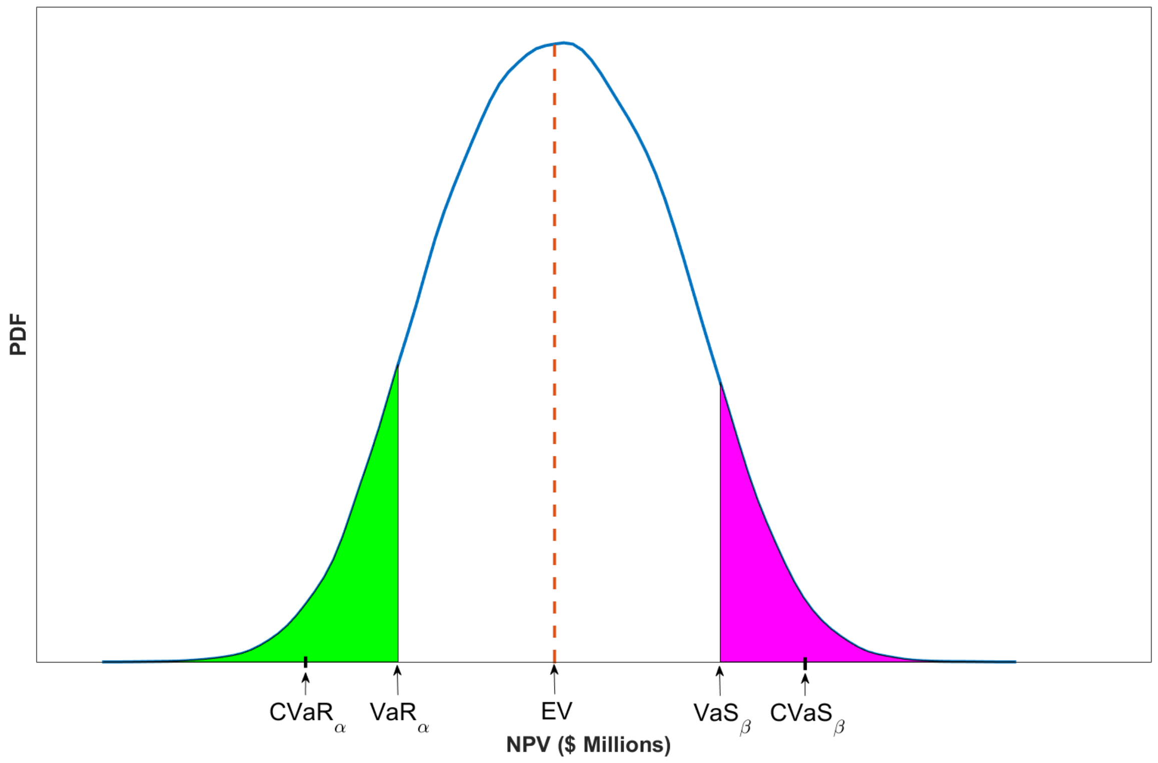


Figure 3.5: Illustration of VaR, CVaR, Expected Value (EV), VaS and CVaS for a given distribution.

4. NUMERICAL RESULTS

This chapter brings together all results for the computational experiments used to test the methods developed in chapters 2 and 3. In section 4.1 we demonstrate the importance of dimensionality reduction to global-search method, while we compare polynomial approximations with both function and interpolation control methods. In section 4.2, we compare the performances of gradient-free versus gradient-based methods. Those two first sections show experiments on a channelized 2-dimensional synthetic reservoir with inverted-five-spot wells configuration and on a 3-dimensional realistic reservoirs with 25 vertical and horizontal wells.

Then in section 4.3, we present computational results for polynomial approximation with model order reduction. We demonstrate, on another 2-dimensional synthetic reservoir, superior performances of the Chebyshev and Spline polynomial representation over the PWC parameterization. We conclude with results for optimization under uncertainty where we show the added value brought our proposed method, using 30 channelized 2-dimensional synthetic reservoirs.

We used Matlab Reservoir Simulation Toolbox (MRST) for two-phase incompressible flow [94], detailed in Appendix B. As an exception, in section 4.3 we used an in-house fully-implicit compressible solver, as described in section 1.1.1.

We conduct all experiments using Texas A&M High Performance Research Computing facility, which facilitated, through MATLAB Distributed Computing Server, running 30 simulations in parallel, each on a different node. Again, an exception is section 4.3, where we used our office 8-core machine, which enable 8 simulation in parallel.

4.1 Optimization Dimensionality Reduction

Results shown in this section were originally published in "Dimensionality Reduction for Production Optimization Using Polynomial Approximations" by Nadav Sorek, Eduardo Gildin, Fani Boukouvala, Burcu Beykal and Christodoulos A. Floudas, in Computational Geoscience, 2017, Springer[1].

In this section, we demonstrate the importance of ODR to gradient-free optimization (GFO) methods by polynomial approximation. We compare the fine parameterization with two polynomial approximation methods, namely, natural polynomials and spline method. We also show the importance of the scaling procedure.

We tested our proposed method on two cases with increasing complexity. First, we solved the same two-dimensional problem as in the motivating example. In the second example, we solved the realistic UNISIM benchmark [95] based on the Namorado reservoir from the Campus basin offshore Brazil [96], where we incorporated a total of 25 wells as implemented in the work shown in [1].

In order to test our method, we performed an optimization procedure for different polynomial degrees and for different number of interpolation points. We compared the FCM and ICM results with traditional optimization methods, employing both gradient-based and gradient-free approaches.

4.1.1 Example 1: Two-Dimensional Model

In the first example, we used the same two-dimensional problem settings as for the earlier motivating example (see Figure 1.2). We formulated the optimization problems as shown in Eq. 3.5 and 3.8. We ran the optimization for different polynomial controls with degrees ranging from one to four (Linear, Quadratic, Cubic and Quartic polynomials) and equivalently, for two, three, four and five interpolation points. We compared the

[1]Reproduced with permission of Springer. Further reproduction is prohibited without permission.

performances of different approaches along with a comparison with the traditional control formulation. We call this traditional formulation as ”Free” control, since the control trajectory is free to have any irregular shape, without being confined to follow any polynomial trajectory. In Table 4.1 we present the control set cardinality for each method, as followed from Eq. 3.3 and Eq. 3.6, and dimensionality reduction as a ratio between the Free method and the ICMs and FCMs. We can see that the problem becomes smaller as the polynomial degree or the number of interpolation points decrease.

Table 4.1: Control set cardinality and space dimensionality reduction for Example 1

Control Method	**Control Set Cardinality**	**Dimensionality Reduction Ratio**
Linear Two Points	10	18.5
Quadratic Three Points	15	12.33
Cubic Four Points	20	9.25
Quartic Five Points	25	7.4
Free	185	1.00

We show in Table 4.2 the box constraints used in this work, which are applied for both Example 1 and next in Example 2. For all cases we used PSO as the global-search algorithm. Since PSO is a stochastic optimizer, we ran the simulation multiple times in order to obtain statistically meaningful interpretations regarding the robustness of the compared methods. Each case was run 10 times, each with a different seed random generator.

Table 4.2: Variables hard bounds in both example 1 and example 2.

Control Method	Control Variables	Units	Injectors Lower Bound Ex.1	Injectors Upper Bound Ex.1	Producers Lower Bound Ex.1	Producers Upper Bound Ex.1
PCM	$a_{j,p}$	$1/psia$	-1	1	-1	1
ICM Free Control	$z_{j,p}$ $u_{j,k}$	$psia$	2,200	3,000	500	2,100

For all gradient free methods, Figure 4.1 and Figure 4.3 show the controls of the best solution found among the 10 runs. We provide the results in a ”side” view such that the polynomial shape can be observed. We also provide a ”top” view for the reader who is more comfortable to this sort of visualization. For both plots we show the Free control with a bang-bang solution for comparison (though not pure bang-bang, as some values are not exactly on the boundaries). Note that we can observe the polynomial behavior between the bounds. Whenever the polynomials went out of bounds, the actual control was set to the bound value. We can see that even without imposing Eq. 3.7 as a constraint, the solutions are very smooth and amendable for deployment in the field. It seems that due to high water cut all solutions drive the controls to gradually minimize the draw-down as can be inferred from the decreasing distance between the injector and producers curves.

Fig. 4.2 and Fig. 4.4 show the optimization progress along the first 100 iterations for the best, worst, and closest to the mean of all 10 runs. Note to the rapid objective function (NPV) improvement in the FCM and ICM comparing to the Free method. This fast improvement is due to efficient search in a small dimension space, which will have a major impact for larger optimization problems, as we will see in the next example.

Table 4.3 shows the average (mean), best, worst and standard deviations results of all

Table 4.3: Example 1: Results of 10 optimization runs, (Function tolerance: 10^{-6} for 20 consecutive stall iterations)

Control Method	**Best**		**Mean**		**Worst**		**Standard Deviation**	
	NPV $ Millions	Total Sim. (Equiv. Sim.) Runs	NPV $ Millions	Total Sim. (Equiv. Sim.) Runs	NPV $ Millions	Total Sim. (Equiv. Sim.) Runs	NPV $ Thousands	Total Sim. (Equiv. Sim.) Runs
Linear	17.37	3,720 (124)	17.36	3,894 (129.8)	17.35	2,550 (85)	6.64	1,086.5 (36.22)
Quadratic	17.42	12,120 (404)	17.40	11,190 (373.0)	17.34	10,128 (337.60)	3.57	798.77 (26.63)
Cubic	17.47	12,480 (416)	17.45	13,800 (460.0)	17.43	7560 (252)	14.20	4,529.40 (150.98)
Quartic	17.51	42,150 (1,405)	17.51	29,244 (974)	17.50	28,020 (934)	7.07	6,632.96 (221.10)
Two-Points	17.22	3,270 (109)	17.21	3,513 (171.1)	17.19	3,360 (112)	9.48	1,127.07 (37.57)
Three-Points	17.39	9,720 (324)	17.39	7,785 (259.5)	17.39	7,440 (248)	0.62	1,226.48 (40.88)
Four-Points	17.49	12,420 (414)	17.48	15,399 (513.3)	17.4	9,720 (324)	27.11	5,705.12 (190.17)
Five-Points	17.53	14,160 (472)	17.52	13,404 (446.8)	17.51	12,180 (406)	10.41	2,429.56 (80.99)
Free	17.74	59,040 (1,968)	17.67	38,295 (1,276.5)	17.62	33,660 (1,222)	38.69	9,264 (308.79)

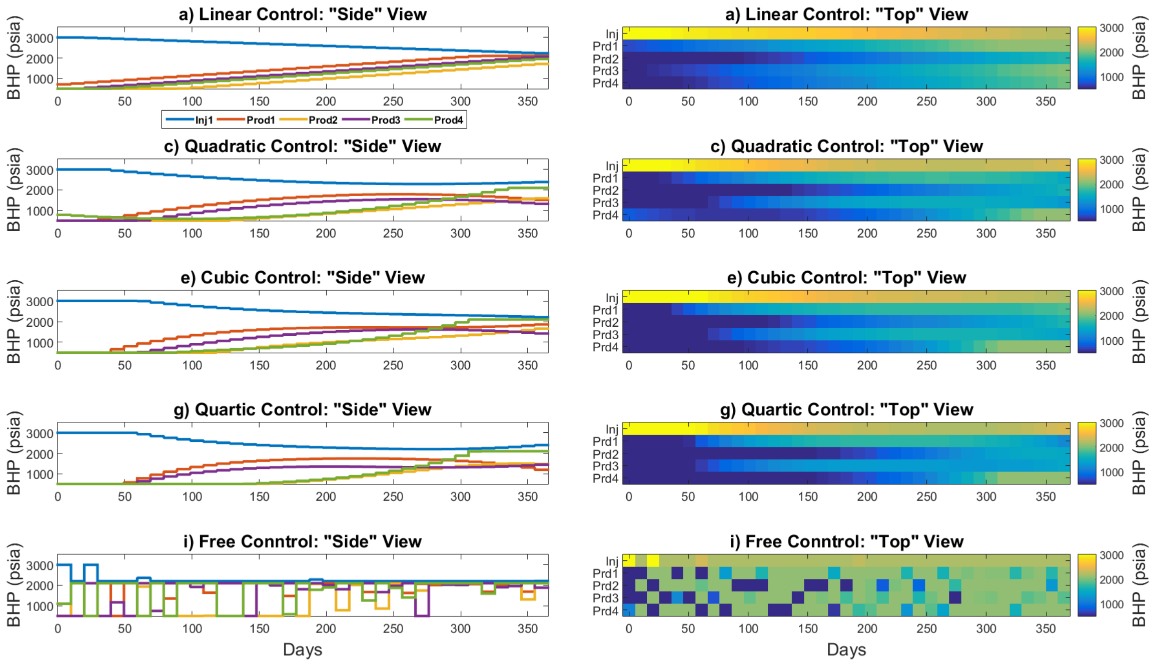


Figure 4.1: Example 1 - FCM and Free controls - results of the best objective function found among all 10 runs.

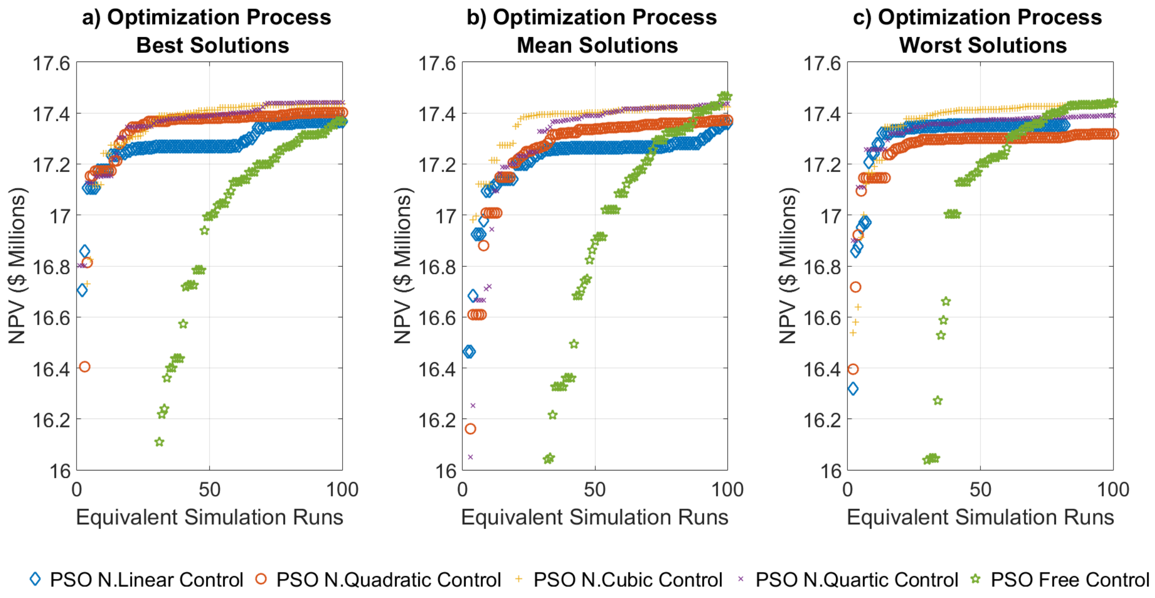


Figure 4.2: Example 1 - FCM and Free controls - First 100 iterations of the best, worst and closest to mean runs out of the 10 runs.

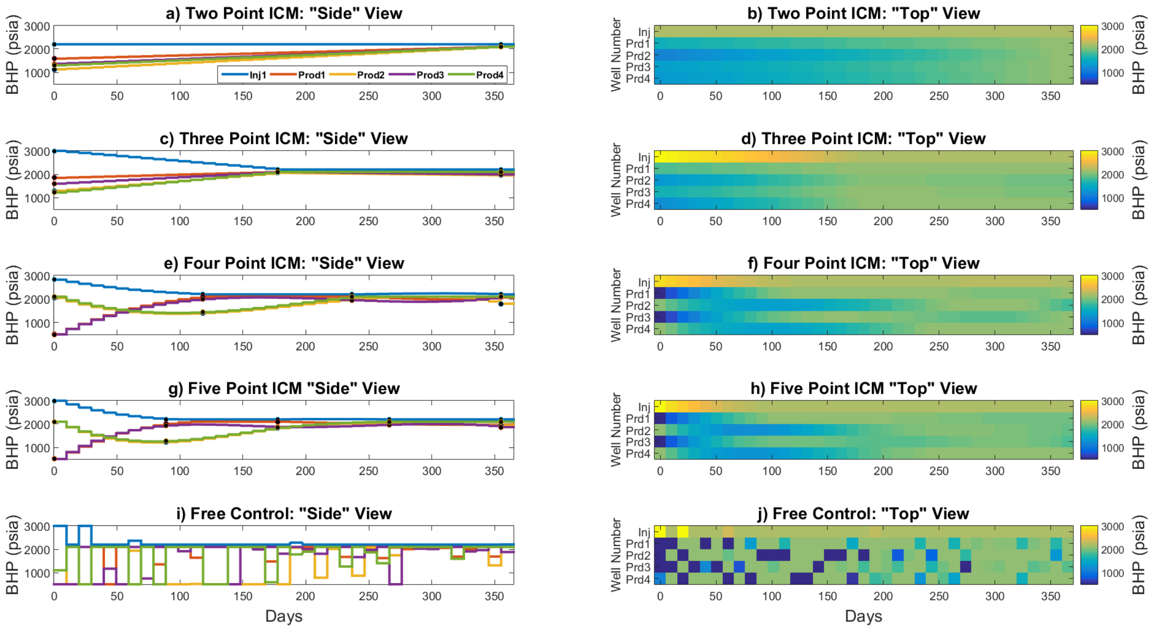


Figure 4.3: Example 1 - ICM and Free controls - results of the best objective function found among all 10 runs.

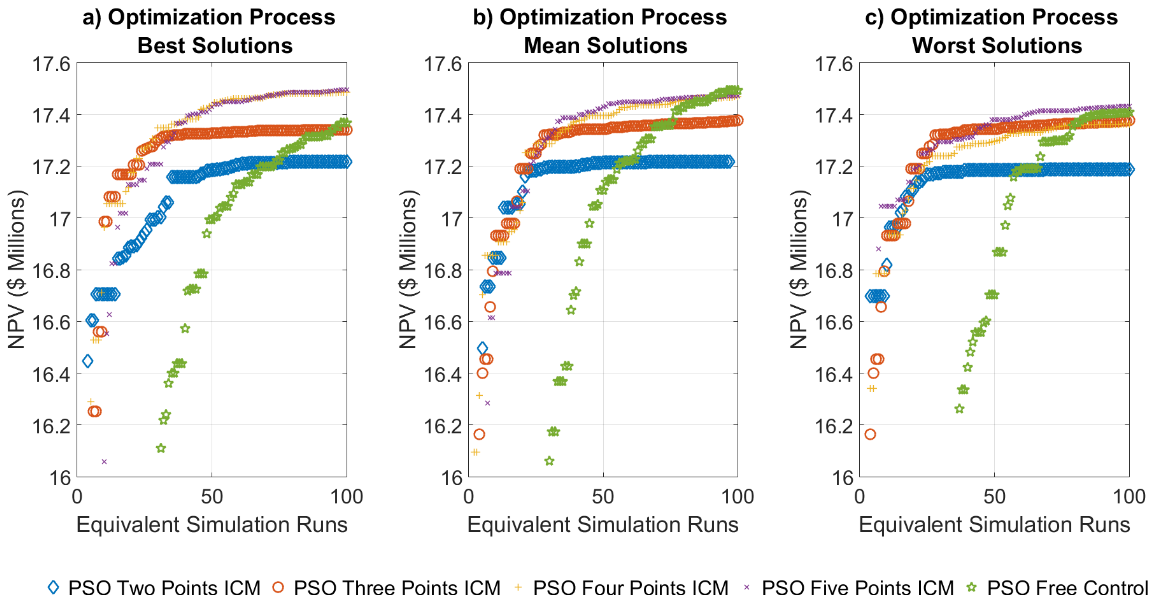
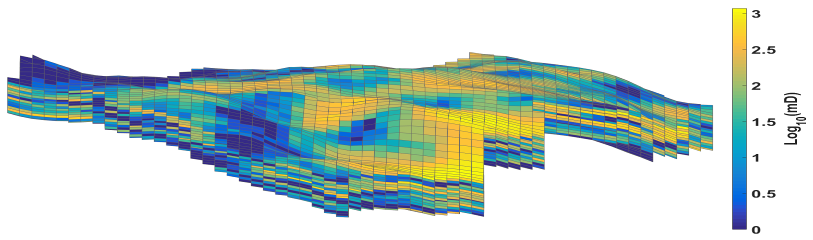
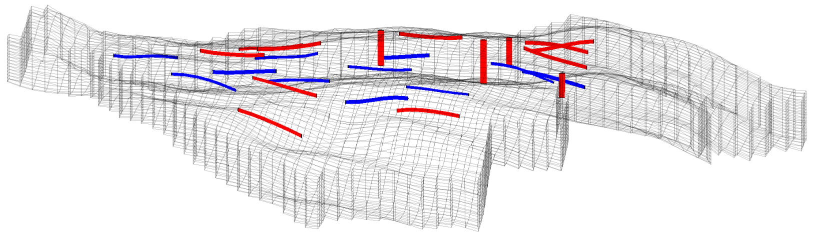


Figure 4.4: Example 1 - ICM and Free controls - First 100 iterations of the best, worst and closest to mean runs out of the 10 runs.



(a)



(b)

Figure 4.5: UNISIM Benchmark: a) Permeability distribution. b) Completed wells intervals of the four vertical and 10 horizontal producing wells (in red) and 11 horizontal injection wells (in blue) used for production optimization (following the work in [1]).

10 runs for each method. The results are given in terms of the obtained NPV, the number of simulations and the number of equivalent simulation runs (equal to one parallelized PSO iteration). We can see that in addition to dimensionality reduction, the flexibility and multi-modality of the function is also essential, as inferred from the results shown in Table 4.3. Note that in all categories (mean, best and worst) among the ICMs and FCMs, a higher polynomial degree (or number of interpolation points) corresponds to a superior NPV at convergence, along with larger number of iterations and simulation runs. Thus, we observe a trade-off between the rate of improvement and the quality of the solution.

4.1.2 Example 2: Three-Dimensional UNISIM-I-D Model

Under the curse of dimensionality paradigm, the idea is that the larger the problem, the computational cost should be exponentially increased, and thus the effects of a dimensionality reduction should be more evident. Thus, we test ICM and FCM on a much larger system, namely the UNISIM-I benchmark.

The original high resolution reference benchmark (with approximately 3.5 million active grid blocks) is based (with some modifications) on the structural, petrophysical and facies model of the Namorado oil field, located in Campos Basin, Brazil [95]. Avansi and Schiozer [97] upscaled the reference model into a medium-scale reservoir in order to make the model applicable to reservoir management optimization procedures that require many simulation calls. In our work, we used this upscaled version of UNSIM-I-D, which consists of 20 layers, a 100 x 100 x 8 m grid cell resolution, and about 37,000 active grid blocks (see [98] for further model specifications).

Fig. 4.5a shows the permeability distribution, in a log_{10} scale, of the UNISM-I-D model. In addition to the four vertical producing wells provided as part of the benchmark, we incorporated additional 10 horizontal producing wells and 11 horizontal injection wells, as it was done in [1]. That is, we considered in total 25 wells for the production optimization procedure (see Fig. 4.5b).

We set the simulation horizon time to be 5 years and adjusted the controls each month. Hence, the control set per each well included 60 decision (design) variables, and the total cardinality for all 25 wells amounted to 1500 variables for problem P1 as dictated by Eq. 3.3.

As in Example 1, we solved the problem with both P1, P2 and P3 formulations (Eqs. 3.1, 3.5 and 3.8, respectively). Due to relatively long simulation time of this reservoir model, we limit the number of methods tested. We chose quadratic and cubic polynomials, and their dimension-equivalent methods, the 3 and 4 points interpolation. Following the results of Example 1, these methods yield a combination of fast convergence and good quality solutions.

Following again from the results of the first example, which show excessive number of simulation runs without significant improvement, we loosen the convergence criteria for all methods solved by PSO algorithm from 20 stalled iterations below a relative objective function change of 10^{-6} in Example 1, to 12 stalled iterations with a 10^{-4} threshold (see Appendix A for more details). In Table 4.4 we present the obtained control set cardinality for each method, as followed from Eq. 3.3 and Eq. 3.6, and the dimensionality reduction ratio.

Table 4.4: Control set cardinality and solution space dimensionality reduction for Example 2.

Control Method	**Control Set Cardinality**	**Dimensionality Reduction Ratio**
Quadratic Three Points	75	20
Cubic Four Points	100	15
Free	1500	1.00

Due to the large computational cost, we conducted the optimization for each one of the PSO implementation only three times, each with a different seed random generator. These mere three optimization repetitions were sufficient to demonstrate the increasing effectiveness (comparing to Free PSO) of the ICM and FCM when solving a bigger problem. Table 4.5 shows the average (mean), best, worst and standard deviations results of the three runs of each tested method. As predicted, the improvement in computation effort is considerably more profound in a larger scale problem. While the Free method used an excessive number of simulation runs and did not converge within the optimization time limitation (36 hours), the FCMs and ICMs quickly converged to a considerable higher value.

Table 4.5: Example 2: Results of 10 optimization runs, (Function tolerance: 10^{-4} for 12 consecutive stall iterations)

Control Method	**Best**		**Mean**		**Worst**		**Standard Deviation**	
	NPV $ Millions	Total Sim. (Equiv. Sim.) Runs	NPV $ Millions	Total Sim. (Equiv. Sim.) Runs	NPV $ Millions	Total Sim. (Equiv. Sim.) Runs	NPV $ Thousands	Total Sim. (Equiv. Sim.) Runs
Quadratic	17.57	5,220 (174)	17.47	4,190 (139.67)	17.33	4,410 (147)	122.16	1,155.8 (38.53)
Cubic	17.67	4,410 (147)	17.60	5,140 (171.33)	17.48	2,940 (98)	100.94	725.8 (24.19)
3 Points	17.58	3510 (117)	17.53	3,660 (122.00)	17.48	3,750 (125)	50.04	130.77 (4.36)
4 Points	17.68	4,680 (156)	17.65	4,830 (161.00)	17.62	4,980 (166)	42.96	212.1 (7.1)
Free	15.94[a]	18,915 (630.5)	16.09[a]	18,960 (632)	15.80[a]	18,870 (629)	145.0	45.00[b] (1.5)[b]

[a] Optimization ended due to maximum allowable execution time: 36 hours. All other values in the table were obtained within less than 12 hours.

[b] Following [a], all optimization had approximately same number of iterations, and thus low standard deviation.

4.1.3 Dimensionality Reduction: Results Conclusions

In this chapter we demonstrated the importance of optimization dimensionality reduction to Gradient Free Optimization Method. We used Particle Swarm Optimization as the gradient-free optimizer in a high performance computing environment, and provided two parametrization procedures for water-flooding production optimization problems by projecting the original infinite dimensional controls space into a polynomial subspace.

The goal of these parametrization procedures is twofold: acceleration of a gradient-free optimization process and leading the controls to gain smooth solutions, which might be required from a production engineering point of view. We named our polynomial approximation approaches as Functional Control Method (FCM) and Interpolation Control Method (ICM). In the first method, the FCM, the (normalized) controls of each well are represented by a polynomial function with a normalized time as its variable. In the second method, the ICM, piecewise cubic polynomials are obtained by a ”not-a-knot” smooth spline interpolation between equidistant points in time. The optimization variables are either the polynomial coefficients, or the values sampled from the original space (BHP in this paper) at the selected points for interpolation.

We stressed the importance that any polynomial parametrization should be able to produce bang-singular solutions. Following this discussion, we showed how we improved previous FCM formulation shown in the literature by allowing the polynomial to take values out of the actual bounds. In these cases the control values introduced to the simulator are the bound values. We also show different normalization procedure, which better capture the polynomial multi-modality behavior.

We tested our proposed methods on two examples with increasing complexity: a two-dimensional synthetic channelized reservoir and the realistic three-dimensional UNISIM benchmark. In the first example, we considered a five spot pattern where we have extensively tested four different FCMs (and ICMs): linear (2 points), quadratic (3 points), cubic (4 points) and quartic (5 points). Furthermore, in a second three-dimensional example, we imposed horizontal and vertical wells (25 in total) where we compared two FCMs (and two ICMs): quadratic (3 points) and cubic (4 points).

In both examples we saw higher rate of objective function growth when optimizing over a reduced space. The difference was more distinct in the realistic case (Example 2), and demonstrates the importance of re-parameterization.

In both examples, we showed that ICM outperforms FCM. We attributed this behavior to the well-defined bounds on the decision variables in the ICM when compared to the FCM. We also observed that the higher the polynomial degree or the number of interpolation points, the higher the obtained objective function value, along with increasing number of iterations up to convergence. We showed that as the problem scale increases, the contribution of FCM or ICM is greater. In the larger problem, the traditional formulation solved by PSO did not converge within maximum allowed CPU time, while our approach did converge to a considerably higher NPV within significantly fewer iterations.

Though ICM outperforms FCM in the experiments shown in this chapter, this might not always be the case. Other functions than natural polynomials might be used and lead to better performances. Specifically, orthogonal polynomial basis functions, which will be tested in the next chapter seems to be a promising avenue.

4.2 Gradient-Based versus Gradient-Free

As mentioned in the introduction, well-control production optimization problems have been predominantly solved insofar by two approaches: stochastic global-search algorithms and local-search algorithms. The local search can be either gradient-based [37, 29, 31, 32, 99, 28], approximated-gradient-based [100, 57, 38] or gradient-free [101, 58]. Hybrid methods of global and local search have been suggested as well [63]. In this Chapter we will conduct a comparison between the two approaches. In the local search we use a GBO method with the adjoint method as implemented in [10].

Although gradient-based methods might converge to a local optimum, they have the obvious advantage of faster convergence and the ability to cope with considerable large number of decision variables, if implemented wisely. We obtained the gradients using MRST adjoint implementation, and compared different optimization methods provided by the built-in Matlab *fmincon* function [73]. Based on preliminary tests we performed to the different algorithms implemented in *fmincon*, we found three that perform better than the others. These three constrained optimization algorithms are active-set, interior point with the BFGS inverse hessian quasi-Newton approximation (IP-BFGS), and interior point with limited memory BFGS inverse hessian quasi-Newton approximation (IP-LBFGS).

We will use the same two examples from the previous chapter. In both examples we use PSO algorithm as the global solver. As for the local solver, for Example 1 we compare only IP-LBFGS, while for Example 2 we compare also IP-BFGS and Active-Set methods.

4.2.1 Example 1: Two-Dimensional Model

Results shown in this section were originally published in "Model Order Reduction and Control Polynomial Approximation for Well-Control Production Optimization" by Nadav Sorek, Hardik Zalavadia and Eduardo Gildin, in the proceeding of Reservoir Simulation Conference, 2017, Society of Petroleum Engineers[2].

We show in this section a test case for the three different parametrization methods, namely PWC, Spline and Chebyshev polynomials, each for ten different levels of parametrization and for two types of algorithms (gradient-based and gradient-free). In total, we performed 60 optimization runs for this section. Our goal here is to compare the different parametrization schemes as well as compare gradient-based and gradient-free methods.

[2]Copyright 2017, Society of Petroleum Engineers, Reservoir Simulation Conference. Reproduced with permission of SPE. Further reproduction is prohibited without permission.

For each parameterization, we tested the range from one decision variable per well (constant value) up to ten variables per well. We reiterate that this number translate to the number of control intervals for PWC, to the number of interpolation points in Spline and to the number orthogonal polynomial basis in the Chebyshev method.

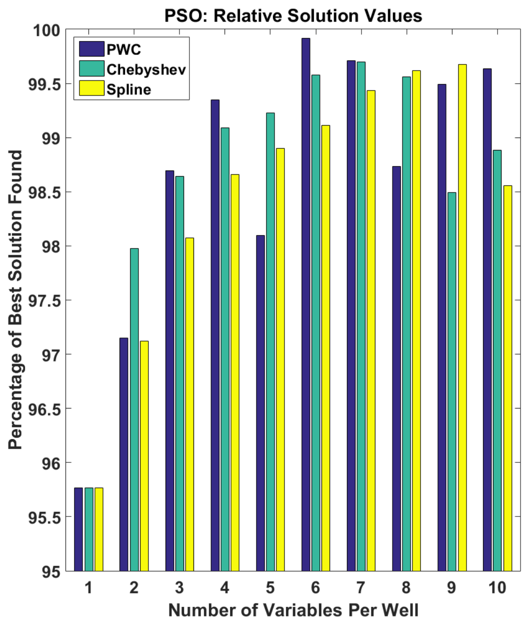
As for the gradient-based algorithm, we used the Interior Point Optimization (IPOPT) method with limited memory BFGS (L-BFGS) inverse hessian quasi-Newton approximation (MathWorks, 2016). Average values were used as initial guess for all gradient-based optimization runs. For the gradient-free method, we used Particle Swarm Optimization with dynamic neighborhood and dynamic inertia (MathWorks, 2015), with a swarm of 30 particles.

We assumed a ratio of ten to one between oil revenue and cost of water produced and injected ($100 : 10 : 10\$/STB$, respectivly), with an interest rate of 10%. Note that though these prices might not reflect the actual market prices for a given point in time, same optimal control solution would be obtained for different prices scenarios as long as the oil-water price ratios are kept the same, as can be inferred from Eq. 1.8.

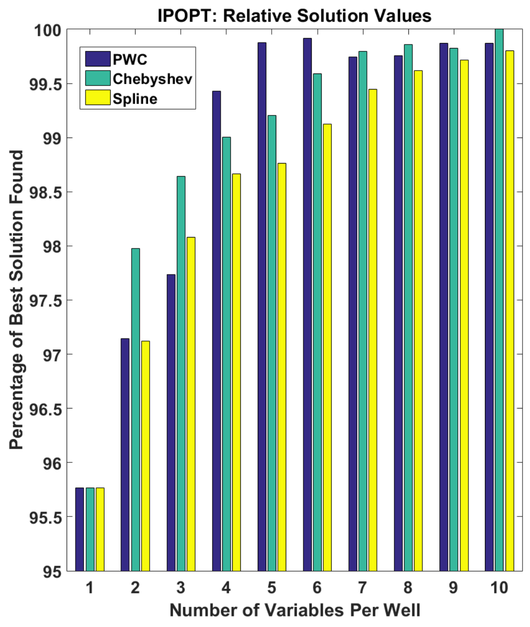
The only inequality constraints we considered are hard bounds. Output inequality constraints are out of the scope of this work. For all producers, we set a lower bound of 500 psia and an upper bound of 2200 psia. For the injector, the upper and lower bounds are 2100 and 3000 psia, respectively.

Figure 4.6 shows the relative comparison results and Figure 4.7 shows the number of iterations to convergence for all ten levels of parameterization, and for both gradient-free using PSO and gradient-based using IPOPT methods. All the values are given in percentage relative to the best solution (\$17.73 million) obtained by Chebyshev parameterization with 10 polynomials basis for each well.

The convergence criterion for the PSO was 10 stall iterations without a relative improvement above a threshold of 10^{-6}. All the IPOPT cases stopped after the relative norm



(a)



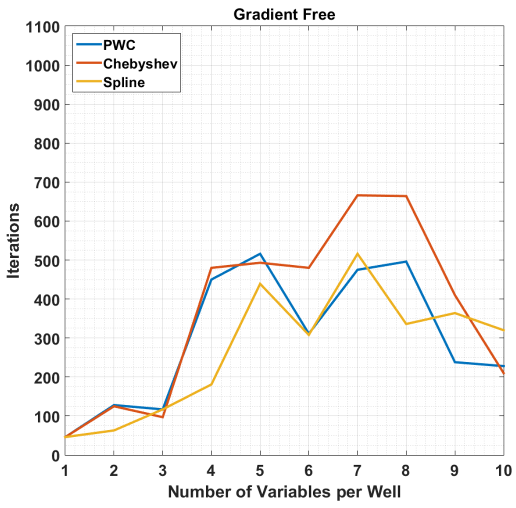
(b)

Figure 4.6: (a) Gradient free (PSO) and (b) gradient based (IPOPT) results for different numbers of basis functions, normalized to the best solution found ($17.73 million).

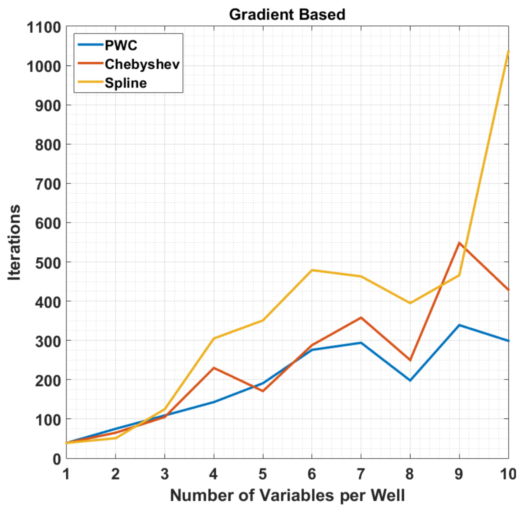
of the step (change in the decision variables) reached below a threshold of 10^{-8}. We note that at the time of convergence with respect to this criterion, all gradient base cases did not satisfy the first optimality condition. Thus, the algorithm did not guarantee the existence of KKT conditions, implying the local optimality was not necessarily obtained.

From both Figure 4.6a and Figure 4.6b it is hard to derive a general conclusion which parameterization methods yield the best results. Note that parameterization level 1 translates into optimization of one constant value, and thus all methods are equivalent in this level. In the gradient-free approach, (Figure 5a) PWC achieved the highest objective function at four levels (3,4,7 and 10), Chebyshev polynomials at three levels (2,5 and 6) and Spline at two levels (8 and 9) which had the highest value in the gradient free group ($17.67 million). Note that due to the stochastic nature of PSO, testing more realizations per each level of parameterization might lead to different observations.

In the gradient-based approach (Figure 4.6b), PWC achieved the highest objective function values at four levels (4,5,6 and 9) and Chebyshev polynomials at five levels (2,3,7,8 and 10, the last with the highest value in all 60 runs). Note that Spline method underperformed at all levels in the gradient-based approach.



(a)



(b)

Figure 4.7: (a) Gradient free (PSO) and (b) gradient based (IPOPT) number of iterations.

In terms of computational cost, note from Figure 4.7a that, in the gradient-free approach, Spline achieves the fastest convergence in most of the cases, and Chebyshev is the slowest in most of the cases. From Figure 4.7b, we can see that, in the gradient-based approach PWC converge the fastest in most of the cases, while Spline is the slowest in most of the cases.

Though local optimality was not proven, we can see from Figure 4.6 how the gradient-based method consistently outperforms the gradient-free method. We performed tests (not shown here) with more tight gradient-free convergence criteria, but they yielded only slight solution improvements with much higher computational cost (number of iterations). We note here the work of [2] which had a different observation, that the gradient free methods outperform SQP gradient based algorithm, where the gradients were acquired via finite difference derivatives (same as this current work). If the adjoint implementation is available, the gradient based methods become much more efficient in terms of the number of simulation runs.

From Figure 4.6b, the gradient-based method consistently produced better objective values upon adding more variables for most of the cases. PWC is an exception where, in some cases (in both algorithms) adding more variables causes a decrease in the obtained solution. Thus, it seems that PWC is more prone to be trapped in the local solutions.

From the previous observation of improving solutions as the number of basis increases, we reinforce the claim that as the number of basis approaches infinity, the approximation to the infinite solution turns more accurate.

Figure 4.8 shows the optimal control trajectories obtained by each parametrization method for a level of 10 variables per well. The control changes every 10 days in these cases. The results are given for both gradient-free (PSO) and gradient-based (IPOPT) methods. The control trajectories shown for IPOPT with Chebyshev polynomial approximation is the solution which produced the highest NPV.

4.2.2 Example 2: Three-Dimensional UNISIM-I-D Model

Results shown in this section were originally published in "Dimensionality Reduction for Production Optimization Using Polynomial Approximations" by Nadav Sorek, Eduardo Gildin, Fani Boukouvala, Burcu Beykal and Christodoulos A. Floudas, in Computational Geoscience, 2017, Springer[3].

In this section, we go back to the 25 wells UNISIM model shown in Figure 4.5b. In the previous section we saw that the gradient based method outperforms the gradi-

[3]Reproduced with permission of Springer. Further reproduction is prohibited without permission.

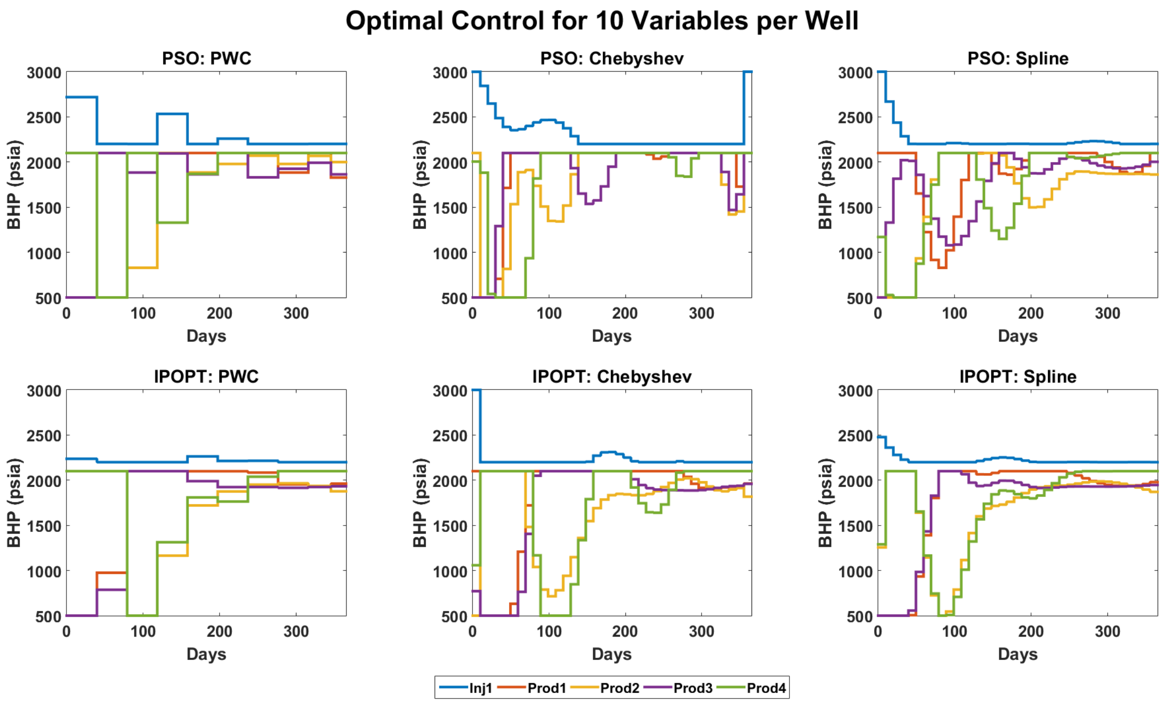


Figure 4.8: Optimal controls of level 10 obtained by each parametrization method. Top: Gradient-free (PSO). Bottom: Gradient-based (IPOPT).

ent free method. In fact, once the gradients are acquired by the adjoint model, gradient based method can perform well even with thousands of variables. Hence, in this section we would like to compare adjoint-based method with fine-scale parameterization with gradient-free method parametrized by polynomial approximation.

Fig. 4.9 shows three comparison of the processes along the optimization iterations. First, Fig. 4.9a compares the best runs of the gradient-free methods, where we can see how a relatively high dimensional problem (comparing to Example 1) impacts the optimization process. Note that the objective functions of ICMs and FCMs increase and converge much faster than the Free PSO. All ICMs and FCMs converged within less than 12 hours, while the Free PSO did not converge and terminated after 36 hours with an objective function values almost 10% lower than the ICMs and FCMs results.

Fig. 4.9b compares the gradient-based methods. We ran all three methods once from a single starting point, with a maximum of 1000 iterations. Note that IP-LBFGS performed far better than the other gradient-based methods (converging faster to a significant higher NPV). These superior performances of the L-BFGS algorithm is consistent with previous results for large-scale optimization problems (see [102, 103]).

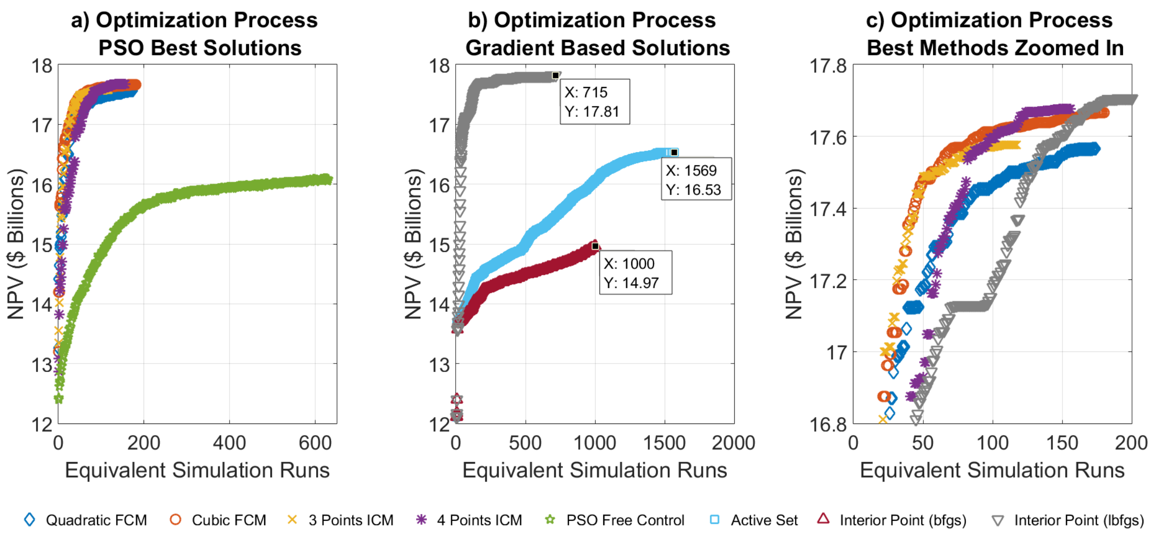


Figure 4.9: Example 2 - a) Gradient-free methods best runs comparison. b) Gradient-based methods comparison with final solutions. c) Zoomed-in to the first 200 iterations of the ICMs, PCMs and the IP-LBFGS methods.

Note also from Fig. 4.9b that IP-BFGS did not converge and the optimization ended when reached to the maximum allowable number of iterations. On the other hand, the active-set method had more than 1000 simulation runs, as each iteration might consist of several simulation runs when a quadratic sub-problem is solved. Active-set converged after 736 iterations (with 1570 simulation runs), when the magnitude of directional derivative in search direction was less than $2 \cdot 10^{-6}$ and maximum constraint violation was less than 10^{-6}. Contrarily, IP-LBFGS converged after only 644 iterations (with 716 simulation

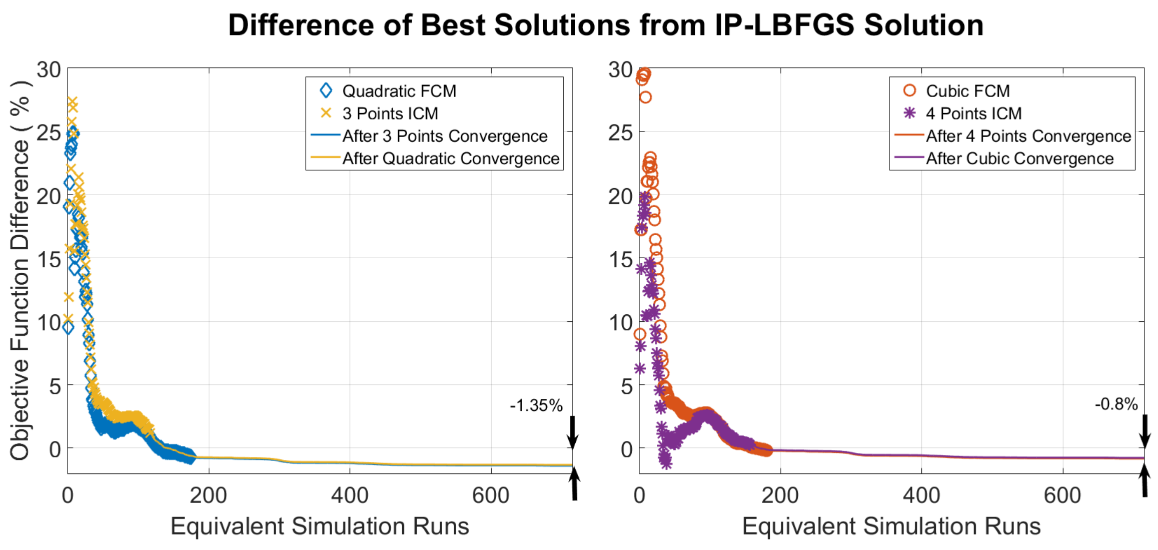


Figure 4.10: Example 2: Percentage difference between the best solutions of FCMs and ICMs to IP-LBFGS. Left: Three variables per well. Right: Four variables per well. Markers and lines are differences before and after convergence, respectively. Final differences after IP-LBFGS convergence are shown.

runs). Similar to the active-set method, the IP-LBFGS had more simulation runs than iterations. The reason here is that the interior point algorithm might not accept a particular step, for example when an attempted step does not decrease the merit function (see [73]). Thus, the algorithm attempts a new step and perform additional simulation. Comparing Figs. 4.9a-b, the FCMs and ICMs performed much better than the active-set and the IP-BFGS.

Fig. 4.9c is a zoom-in to the first 200 iterations of the best methods performed: ICMs, PCMs and the IP-LBFGS. Note that while the PSO-ICMs and PSO-FCMs grow faster than the IP-LBFGS initially, the IP-LBFGS gradually exceeds all other methods. Fig. 4.10 shows this process as an instantaneous difference between each ICM or FCM method to the IP-LBFGS method.

Following the obtained results, some additional remarks are in order:

- If the objective function has indeed a flat plateau, we shall assume in this discussion that the optimal solution can be described by the IP-LBFGS solution. Its value as shown in Fig. 4.9b is \$17.81 billions.

- Interestingly, each best run of the dimension-equivalent ICM and FCM converged to a similar objective function value, and thus to a similar distance from the optimal solution.
- As can be seen from Fig. 4.10 the difference from the optimal solution is rather small: 1.35% and 0.8% for the cases of 3 and 4 variables per well, respectively.
- Similar to the results of Example 1, a higher degree polynomial or a higher number of interpolation points, leads to better objective function values as well as an increased number of iterations up to convergence (see mean results in Table 4.5).
- The FCMs and ICMs objective function values can improve and reach the optimal solution by adding more variables per well. This can be done iteratively by refining the controls with adding more interpolation points or polynomial coefficients.
- Tightening the PSO convergence criteria will improve the obtained results.

Figs. 4.11 and 4.12 show the best control results obtained by the FCMs, ICMs, Free PSO and the gradient-based methods. Again, we can see the erratic control behavior obtained by the Free PSO (refer to Figs. 4.11e and 4.12e). Interestingly, the FCMs and ICMs exhibit smoother results than all gradient-based methods, even without imposing smoothness with Eq. 3.7. From Fig. 4.9 we can see that the two FCMs, two ICMs and the IP-LBFGS obtained the best objective function values. From Figs. 4.11a-d and 4.11h we observe that these five also share similar control patterns, where most wells were controlled by the maximum (injectors) or minimum (producers) allowable BHP. In all five, there are three producers wells (numbered 17, 19 and 21) in which their pressure

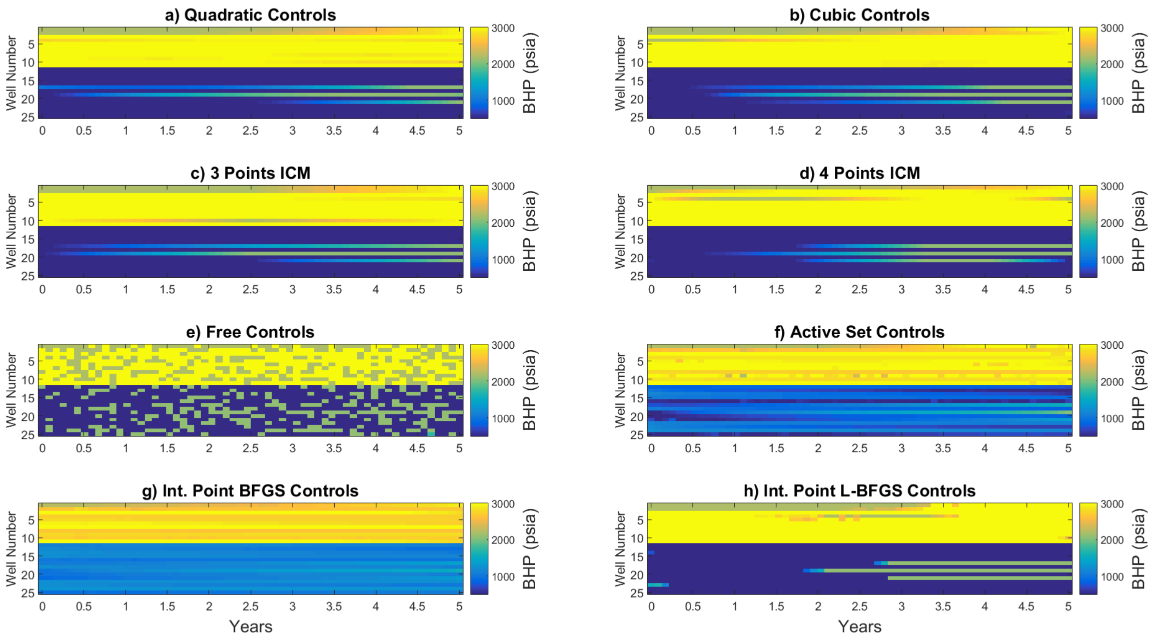


Figure 4.11: Top view: control results obtained by the ICMs, FCMs , Free PSO and the gradient-based methods.

is increase from minimum to maximum allowable BHP during the simulated 5 years. This is consistent with the problem of delaying the water breakthrough and minimizing water cut (as can be seen in Fig. 4.13). Note to the pure bang-bang solution of well 21, and to the short singular arc of wells 17 and 19 in the solution obtained by IP-LBFGS. Also from Fig. 4.13 we observed that the project has not yet reached the economic limit. This proves the need for setting the terminal time as an additional optimization parameter as previously suggested in [22]. Also, note that with ICM or FCM parametrizations, adding a terminal time as another decision variable would only increase the dimensionality of the problem by one. This might, however, require some adjustments to the parallel computing work-flow, as the simulation durations might vary significantly.

As previously done by others [104, 63], we use the term ”equivalent simulation runs” as a performance comparison measure. We note here that this comparison criterion is not "one to one" computational time convertible. That is, one equivalent simulation run

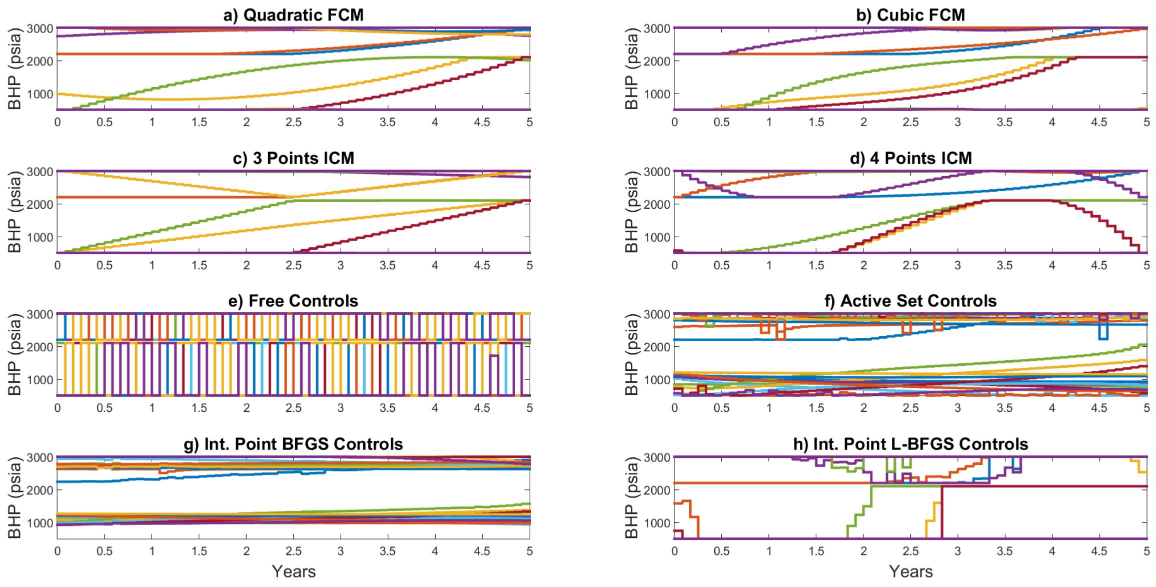


Figure 4.12: Side view: control results obtained by the ICMs, FCMs , Free PSO and the gradient-based methods.

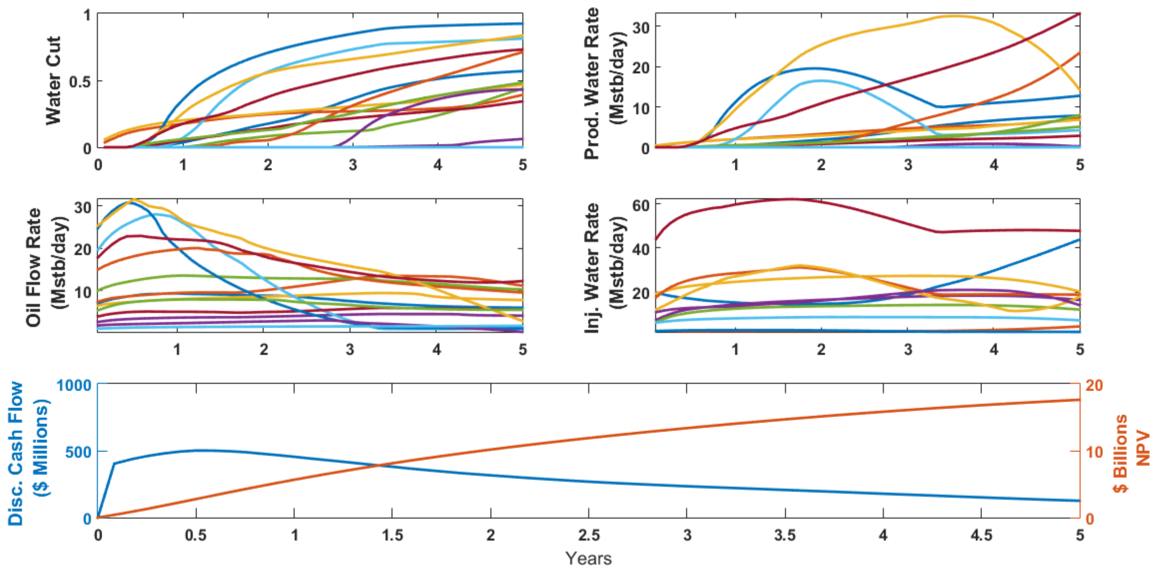


Figure 4.13: Best run of 4 points interpolation for Example 2 - water cut, producers oil and water rates, injector water rates and economics measures

Table 4.6: Average simulation duration for each method.

Method	Average Sim. Duration (Min.)
Act. Set	3.19
4 Points PSO	3.37
Free PSO	3.42
3 Points PSO	3.49
Cubic PSO	3.61
Quad PSO	3.83
IP-BFGS	3.91
IP-LBFGS	4.01

may not have the same duration amongst all methods. Since each PSO simulation (Free, ICM, or FCM) was executed in parallel on a separate node, we observed some increase in average simulation time, due to communication between nodes. On the other hand, in the gradient-based methods, which were run on a single node sequentially, most of the equivalent simulation runs include one adjoint simulation run in addition to the forward simulation run. Due to the high control resolution, each adjoint simulation run consumed a considerable amount of time, about 80% of one forward simulation run time. Thus, the average simulation duration of the gradient-based and gradient-free methods are rather similar, as can be observed from Table 4.6.

When comparing to gradient-based method, the FCMs and ICMs performed better than the active-set and the IP-BFGS. While the ICMs and FCMs grow initially faster than the IP-LBFGS, the IP-LBFGS gradually exceeded all other methods. The final difference from the best solution found by IP-LBFGS was in the range of 0.8%-1.35%.

4.2.3 Gradient-Based versus Gradient-Free: Results Conclusions

In this section, we compared between a gradient free method and a gradient based method for different parameterization method.

We conclude that our method might serve as a handy tool for ”black-box” and gradient-free optimization researchers who aim to optimize general field development (which entail more decision variables such as well-placement). When solving solely the optimal control problem, our methods can be useful in the absence of an adjoint model. If the adjoint model is available, our model can serve as a global search phase before switching into efficient local optimizer (such as IP-LBFGS used in this work) to refine the solution.

4.3 Model Order Reduction with Polynomial Approximation

Results shown in this section were originally published in "Model Order Reduction and Control Polynomial Approximation for Well-Control Production Optimization" by Nadav Sorek, Hardik Zalavadia and Eduardo Gildin, in the proceeding of Reservoir Simulation Conference, 2017, Society of Petroleum Engineers[4].

In this section, we test POD-DEIM methodology (as shown in Appendix A) coupled with control polynomial approximation. We tested three different parameterization methods: Piece-Wise Constant, Chaebyshev Polynoials and Spline.

4.3.1 Problem Description

The tested model is a modified version of the one shown in [2] . The 2D model used is a $40 \times 40 \times 1$ reservoir (1600 grid blocks) with 4 producers and 1 injector placed in the channelized area. The model is discretized using Cartesian grid of size $20 \times 20 \times 20 ft^3$. The porosity of the field is set constant to 0.2. Fig. 4.14 shows the permeability field and the well locations. Here, we use a black-oil fully implicit simulator which solve Eq. 1.1

[4]Copyright 2017, Society of Petroleum Engineers, Reservoir Simulation Conference. Reproduced with permission of SPE. Further reproduction is prohibited without permission.

during each simulation time step (In this section we used an in house black oil simulator). As mentioned earlier, we neglect the capillary pressure effects and consider a slightly compressible flow of oil and water. The initial saturation of oil and water is 0.8 and 0.2, respectively. The Corey-type relative permeability curves with exponent of 2 are used for both the fluids.

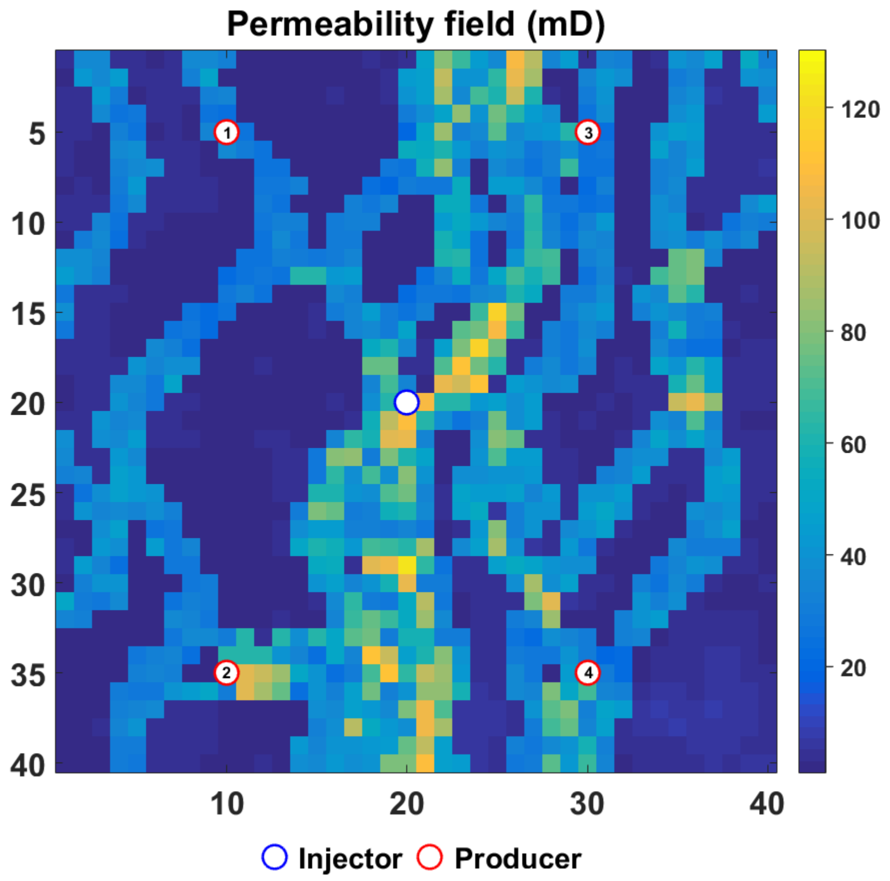


Figure 4.14: Reservoir permeability for testing POD-DEIM with polynomial approximation (after [2])

Initial reservoir pressure is 2100 psia. The injector has an upper bound of 3200 psia and a lower bound of 2200 psia, while the upper bound for all four producers is set to

2100 psia and lower bound to 500 psia. Control time step for each well is 10 days. Oil price is considered to be 50 \$/STB and the water-injection and production costs are each 10 \$/STB.

4.3.2 On-line Training

As we explained in section 3.4, the 8 simulation state outputs of the initial random swarm are concatenated into one snapshot matrix. After constructing the POD-DEIM basis, the optimization proceeds to the second iteration with the reduced order model. Similarly, in the gradient-based approach we used 8 random BHP profiles for training, where the profile which produced the best NPV was chosen as the initial guess for the optimization procedure. By doing so, we utilize the training effort also to advance the optimization process.

The snapshot matrix here is a (3200 x 1530) matrix collected from the 8 particles. The POD basis is constructed using 40 columns from the pressure left projection matrix and 70 columns from saturation left projection matrix. The number of basis for pressure and saturation are selected to capture 99% of the energy of snapshot matrix (see Fig. 4.15a-4.15b). The DEIM greedy algorithm selected 65% cells that need to be used for function evaluations (see Figure 9c). All optimization and parameterization methods use the same number of basis and interpolation cells. Here, we do not try to determine the strategies to reduce the number of interpolation points. This would be a future scope of study.

4.3.3 POD-DEIM Performance for a Representative Case

We will now show a detailed analysis of the POD-DEIM performance. Out of the 6 POD-DEIM optimization runs (3 parameterization methods with 2 different algorithms), we chose one representative case to analyze its POD-DEIM performance. Our choice is Chebyshev Polynomial Approximation coupled with PSO. As will be shown in the next section, Chebyshev method obtained the best results in both optimization search methods.

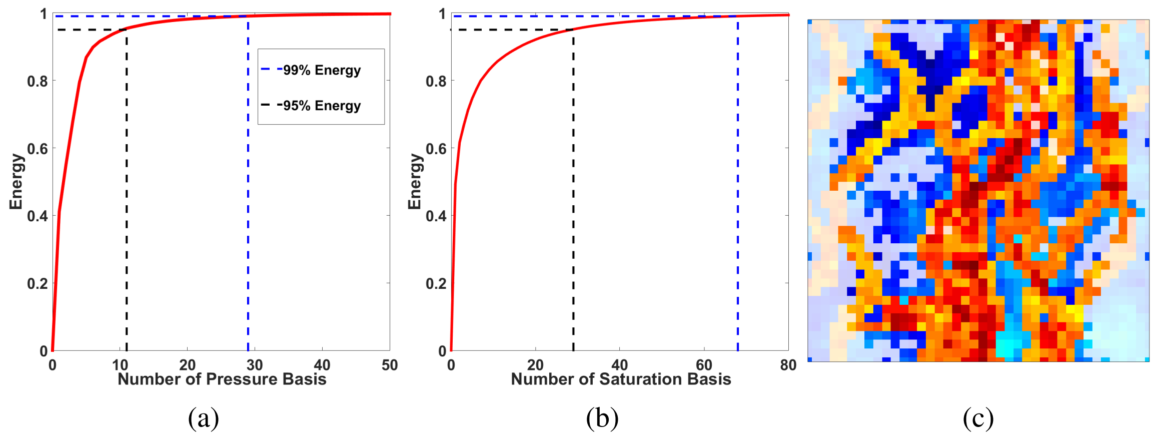


Figure 4.15: a) Pressure and (b) Saturation basis selection for POD and (c) cells selected for DEIM (in strong colors).

Figure 4.16 shows three training samples from the eight initial random Chebyshev profiles of four producers and an injector.

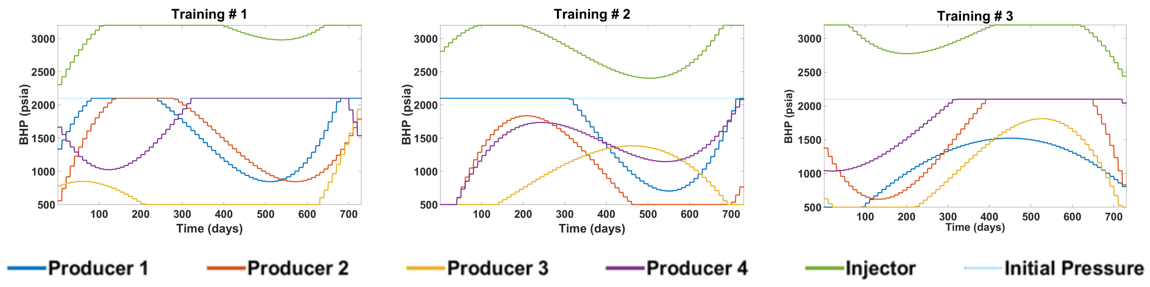


Figure 4.16: Three training samples from the initial eight-particles swarm of Chebyshev profiles.

Next, we validate the accuracy of the reduced order model, by taking the Chebyshev optimal control from the POD-DEIM optimization as an input and comparing the simulation outputs from both full and reduced order models. As can be seen in the results below, the POD-DEIM produces accurate results.

Figure 4.17 compares the saturation profiles for the POD-DEIM optimal control at three points in time: at the first time-step, after 1 year and after 2 years. The comparison is between fine scale and reduced scale simulations. It also shows the relative error for POD-DEIM saturation at these times.

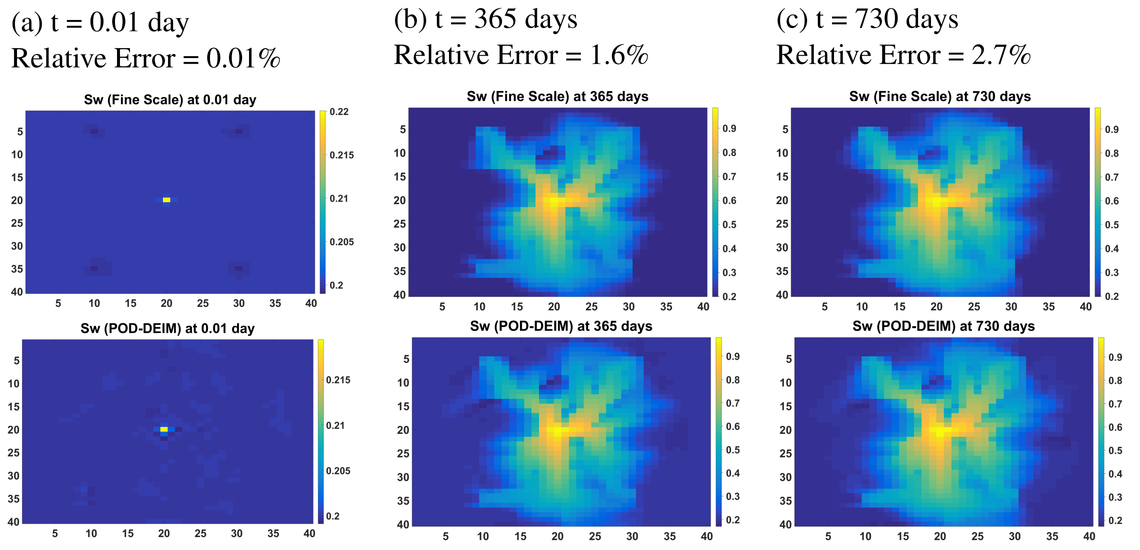


Figure 4.17: Top) Saturation profiles from POD-DEIM Chebyshev optimal control using fine scale simulation and (Below) using the reduced model at different times and their relative errors.

The oil production and water production rates for the POD-DEIM optimal control using fine scale and reduced scale simulations are shown in Figs 4.18 and 4.19, respectively, which show a good agreement in the production rates between the two. The production rates are off in a few initial time-steps but do not have a large effect on the NPV values since these time-steps are very fine. Although we can increase the number of saturation basis to remedy this issue, we found it unnecessary at this point.

We performed similar accuracy analysis to all optimization techniques mentioned and they show a reasonable accuracy as the Chebyshev results.

In the next section, we perform a different accuracy comparison, where we compare the

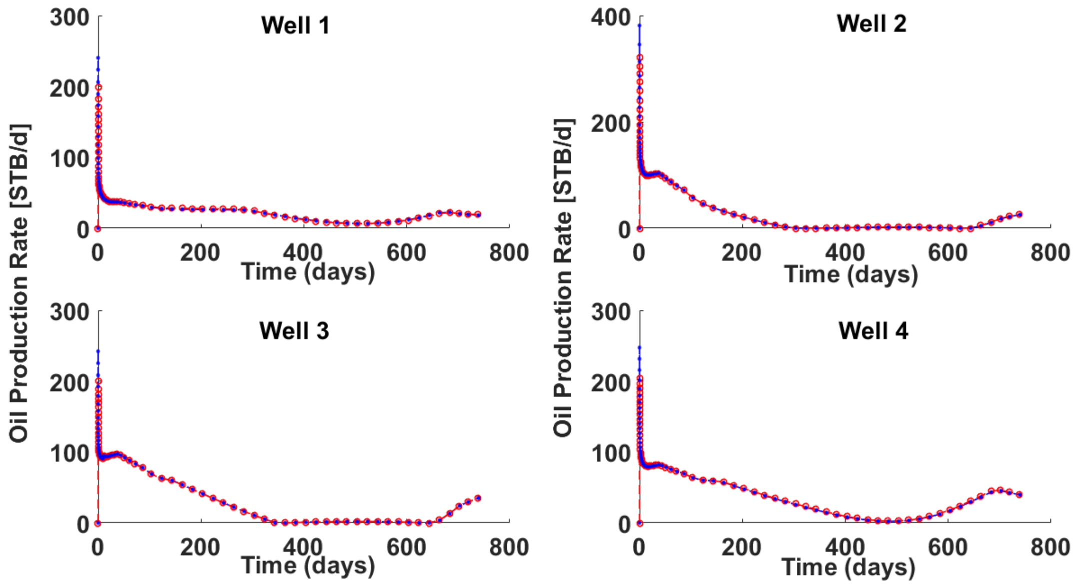


Figure 4.18: Oil production rate from the high fidelity simulation and reduced model simulation for 4 production wells using the Chebyshev optimal control solution from POD-DEIM.

optimal solutions obtained by an optimization using a full-order model to an optimization using a reduced order model.

4.3.4 Comparison: PSO Optimal Control Solutions

In this subsection, we show that global search method such as PSO with POD-DEIM can obtain reasonable optimization accuracy. For that purpose, we use a relative error indicator for the control trajectories, as defined below:

$$Error_{BHP}(\%) = \frac{\left\|BHP_{POD-DEIM} - BHP_{fine}\right\|}{\left\|BHP_{fine}\right\|}, \tag{4.1}$$

where, $BHP_{POD-DEIM}$ is the optimal control obtained from optimization with POD-DEIM and BHP_{fine} is the optimal control from fine scale optimization.

Fig. 4.20 shows the relative error, as defined in Eq. 4.1, for the control trajectories obtained by the PSO algorithm for the Chebyshev control, Spline control and piece-wise

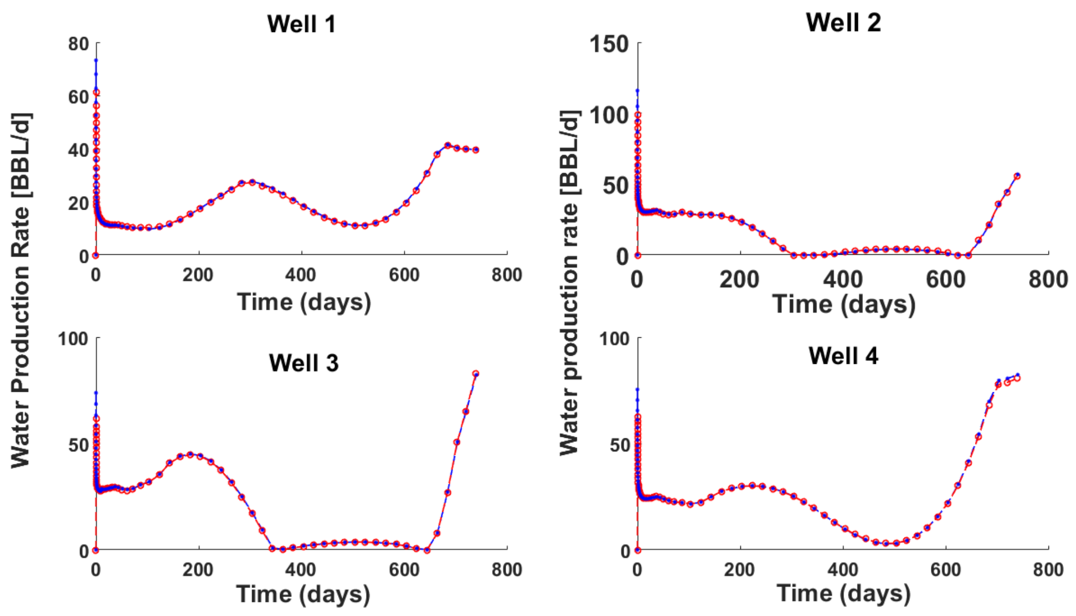


Figure 4.19: Water production rate from the high fidelity simulation and reduced model simulation for 4 production wells using the Chebyshev optimal control solution from POD-DEIM.

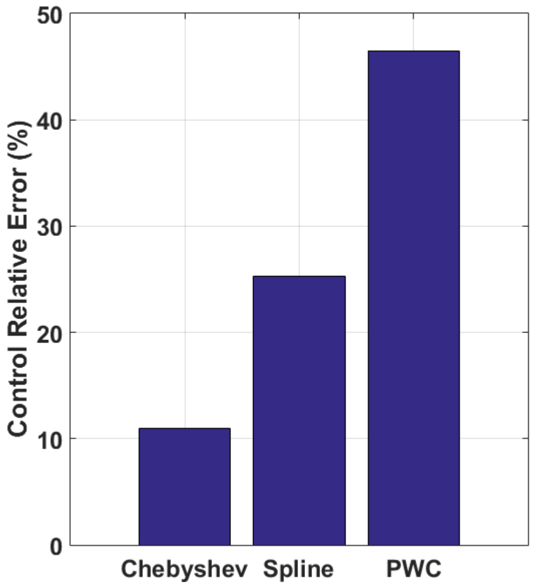


Figure 4.20: Control trajectories relative error for a PSO algorithm.

constant control. Figure 14 shows these optimal control trajectories. As is evident from these optimal control and relative error plots, the polynomial approximations from POD-DEIM and fine scale optimization show similar control strategies.

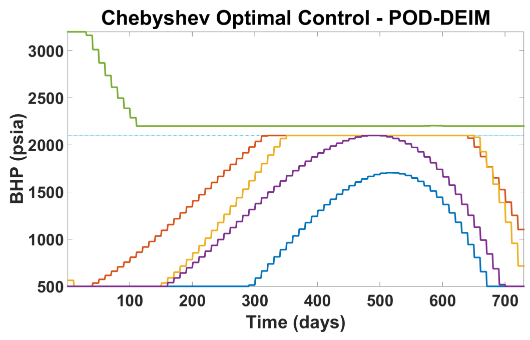
4.3.5 Global versus Local Search with POD-DEIM

In this section, we compare results of all the parameterization methods for optimization with POD-DEIM and fine scale simulation using gradient-based and gradient-free optimizers. The NPV from each method and number of iterations to convergence are shown in the Table 1 and Table 2 using IPOPT and PSO optimizers respectively. We also use the optimal control obtained from POD-DEIM to calculate NPV in the fine scale simulator (last columns of the Table 4.7 and Table 4.8).

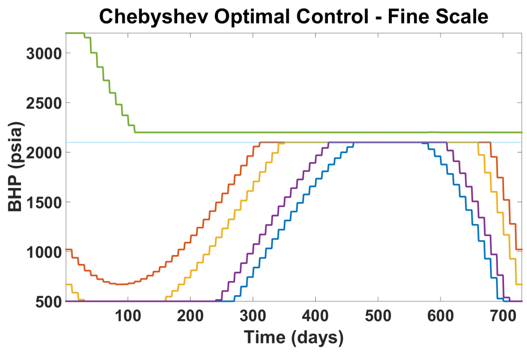
Table 4.7: NPV and number of iterations for fine scale and reduced scale optimization using the 3 methods with IPOPT. NPV values are in million USD

	Fine Scale		**POD-DEIM**		**Fine Scale with POD-DEIM Control**
	NPV	**Iterations**	**NPV**	**Iterations**	**NPV**
PWC	1.871	92	1.817	87	1.801
Spline	1.87	204	1.864	178	1.846
Chebyshev	1.881	46	1.903	25	1.875

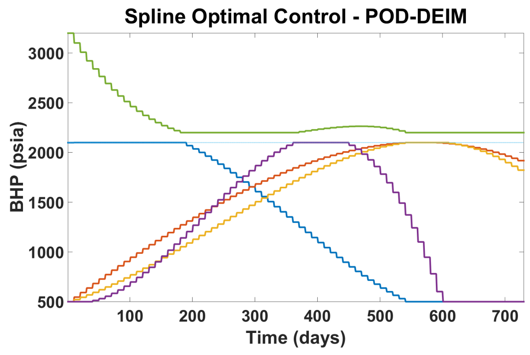
We can see from the tables that POD-DEIM proves to be a viable model reduction technique for production well control optimization problems especially for Chebyshev and Spline controls. We can see again that IPOPT outperforms the PSO.



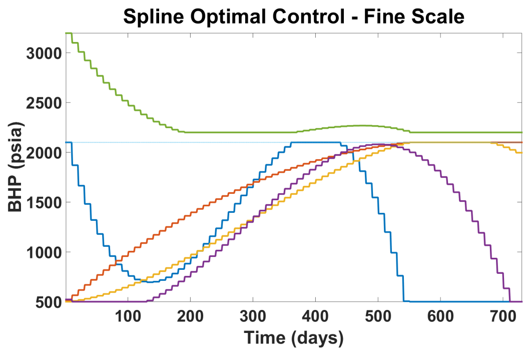
(a)



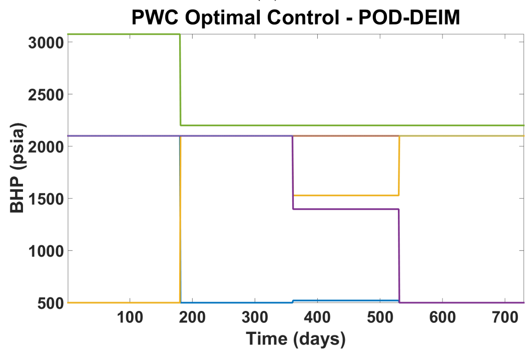
(b)



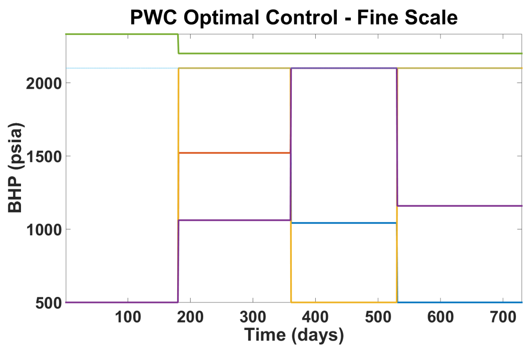
(c)



(d)



(e)



(f)

Figure 4.21: PSO optimal control solutions from POD-DEIM and fine-scale using all three parameterization methods.

Table 4.8: NPV and number of iterations for fine scale and reduced scale optimization using the 3 methods with PSO. NPV values are in million USD

	Fine Scale		**POD-DEIM**		**Fine Scale with POD-DEIM Control**
	NPV	**Iterations**	**NPV**	**Iterations**	**NPV**
PWC	1.828	205	1.79	159	1.74
Spline	1.814	413	1.801	446	1.803
Chebyshev	1.84	262	1.878	280	1.839

In Fig. 4.22, we compare the relative errors of the NPV estimated for each parameterization method using fine scale and POD-DEIM simulations. The relative NPV error is computed as follow:

$$Error_{NPV}(\%) = \frac{\left\|NPV_{POD-DEIM} - NPV_{fine}\right\|}{\left\|NPV_{fine}\right\|}, \tag{4.2}$$

where, $NPV_{POD-DEIM}$ is the optimal control obtained from optimization with POD-DEIM and NPV_{fine} is the optimal control from fine scale optimization.

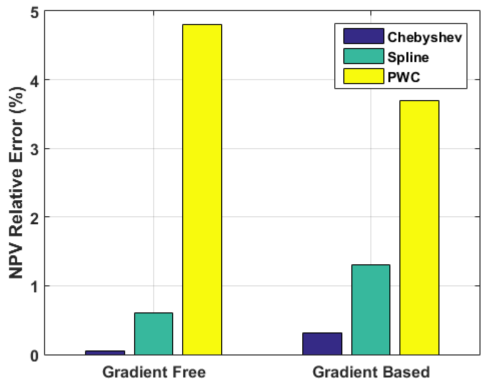


Figure 4.22: Relative NPV errors between the fine scale and reduced scale optimization using PSO and IPOPT for each parameterization methodology.

We can see from Figure 4.22, Table 1 and Table 2 that Chebyshev control optimization showed the best accuracy and the highest NPV for the two optimizers. Also note from Table 4.7 and Table 4.8 that it had a fast rate of convergence when used with IPOPT for this example. NPV with PWC and Spline was less than that of Chebyshev, and Spline showed the least convergence rates for all the cases.

Overall we can say that the POD-DEIM with PWC did not show good results. One of the reasons could be a need for retraining during optimization or a large number of basis need to be selected for such control profiles to represent the dynamics of all the controls.

Regarding runtime saving, one simulation run during the optimization on fine scale is about 20 seconds, whereas, using POD-DEIM takes on an average 12 seconds. Since the optimizations for all the methods were performed on different platforms, we do not report the overall time. But the total computational saving on the entire optimization run in all the cases using the reduced model was about 45-50% (including the offline basis computation stage) when compared to the fine scale optimization.

These results justify the use of POD-DEIM for well-control optimization problem and also a good set of training that is representative of all the BHP profiles during the optimization run particularly for Chebyshev and Spline control methods.

4.3.6 Model Order Reduction: Results Conclusions

In this section, we compare different parameterization methods to reduce the cardinality of the original infinite set of control-decision variables to a finite set. The parameterization techniques include a traditional piece-wise constant (PWC) approximation, a polynomial approximation by Chebyshev orthogonal polynomials and a piece-wise polynomial approximation by cubic Spline interpolation. This optimization problem is proposed to be used with POD-DEIM as the global model order reduction method for computational advantage at the reservoir simulation level, where a new training procedure is proposed that can be used as a part of an optimization run.

In the first example, the parameterization methods are tested for increasing level of refinement using fine scale simulation to evaluate their performances for gradient-free (PSO) and gradient-based (IPOPT) methods. The results showed an increasing performance of optimization with respect to the objective function upon increasing the number of decision variables. For the highest refinement level, Chebyshev polynomial approximation found the maximum NPV for the case considered. PWC exhibited the fastest convergence, but with more likelihood to be trapped in local solutions. Spline method suffered from slow convergence when gradient-base method was used.

In the second example, we tested the control parameterization strategy with POD-DEIM for a given level of parameterization refinement. We observed that Chebyshev polynomials performed better than Spline control and PWC control. The training selection for POD-DEIM here utilizes the information from the first fine scale optimization iteration on multiple random particles run in parallel, which is then used as a snapshot matrix. The optimization then follows from the next swarm chosen by PSO in case of gradient free method and from the best solution as the initial guess for the IPOPT method.

Choice of such a training set proved to perform consistently well for the polynomial and piece-wise polynomial control approximations. POD-DEIM when used for optimization using Chebyshev polynomial approximation, produced very good accuracy with the fine scale optimization as was evident from the errors in optimal control solution and NPV. A 45-50% speedup on the reduced scale optimization runs was achieved for all the cases compared to high fidelity runs. This work demonstrates the use of control parameterization with model order reduction as an efficient methodology for optimization particularly when used with polynomial controls.

In future work, investigation of this methodology on benchmark models and longer simulation horizons will be more appropriate to demonstrate the benefits of such approach.

For example, the test cases in this work performed significantly well without a need for retraining. However, it might be necessary to retrain the model during the course of the optimization run for more complex cases, which may require a robust error indicator to quantify the errors from POD-DEIM.

4.4 Optimization Under Uncertainty

In this section, we test control polynomial approximations under uncertainty, using the methodology presented in section 3.5. We acquire the gradients of each individual realization with the adjoint method described in section 3.3.

In section 4.4.1 we describe the ensemble of scenarios used to represent the geological uncertainty. Each scenario is a different realization of permeability field.

We use the objective function shown in Eq. 3.32. Following the recommendation shown in [90], we tested CVaR at a value of $\alpha = 20\%$ and symmetrically, we chose $\beta = 20\%$ for CVaS, such that the objective function reads as follow:

$$J(u) = \eta_1 J_E(u) + \eta_2 J_{CVaR_{20\%}}(u) + \eta_3 J_{CVaS_{20\%}}(u), \tag{4.3}$$

First, in section 4.4.2, we test the case when we only maximize the expected value. That is the weights vector $\boldsymbol{\eta} = [\eta_1, \eta_2, \eta_3]$ equal to $[1,0,0]$. We used the piece-wise zero-order polynomials and the Chebyshev polynomials, as described in sections 2.5 and 2.4, respectively. We show in both cases that by increasing the number of variables we do not just increase the obtained objective, as we saw in the detrminstic case, but we also mitigates the risk expressed by a reduction in the standard deviation of the objective values.

Then, in section 4.4.3, we test the objective function shown in Eq. 4.3, for different weights vectors $\boldsymbol{\eta}$. We show that the including the proposed objective CVaS in the total objective leads not only to values increase in the upper tail, but also increase the mean and the median ($P50$). On the other hand, it have a negative effect on the mode and the standard deviation.

4.4.1 Test Case Description

Figure 4.23 shows the permeability of the 30 different realizations of channelized reservoir. The realizations generated with a sinusoidal functions with uniformly random amplitudes, periods and phase shifts.

We use the same inverted 5-spot pattern, as in section 4.1.1, where four production wells are located at the corners and the injector well sits in the center. Figure 4.24, shows the histogram of the permeability values present in all realizations. The rest of the simulation-settings are identical to the 2-dimensional deterministic case.

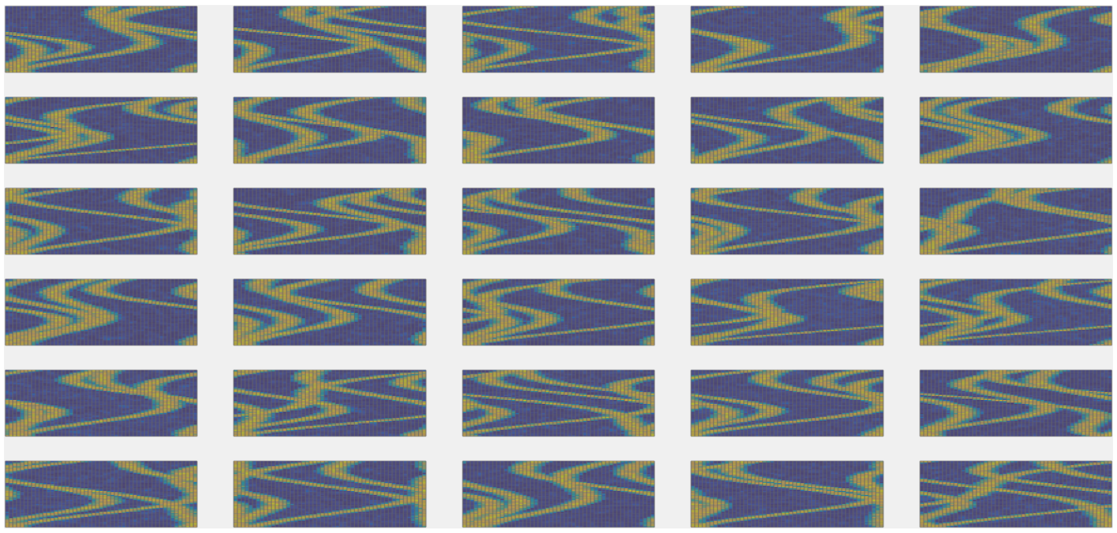


Figure 4.23: Ensemble of 30 permeability field realizations of channelized reservoirs

4.4.2 Uncertainty Reduction by Control Refinement

We now test the case where we consider only the expected value. Thus, the weights vector used in the objective in Eq. 4.3 is $\boldsymbol{\eta} = [1, 0, 0]$. We compare the zero-order poly-

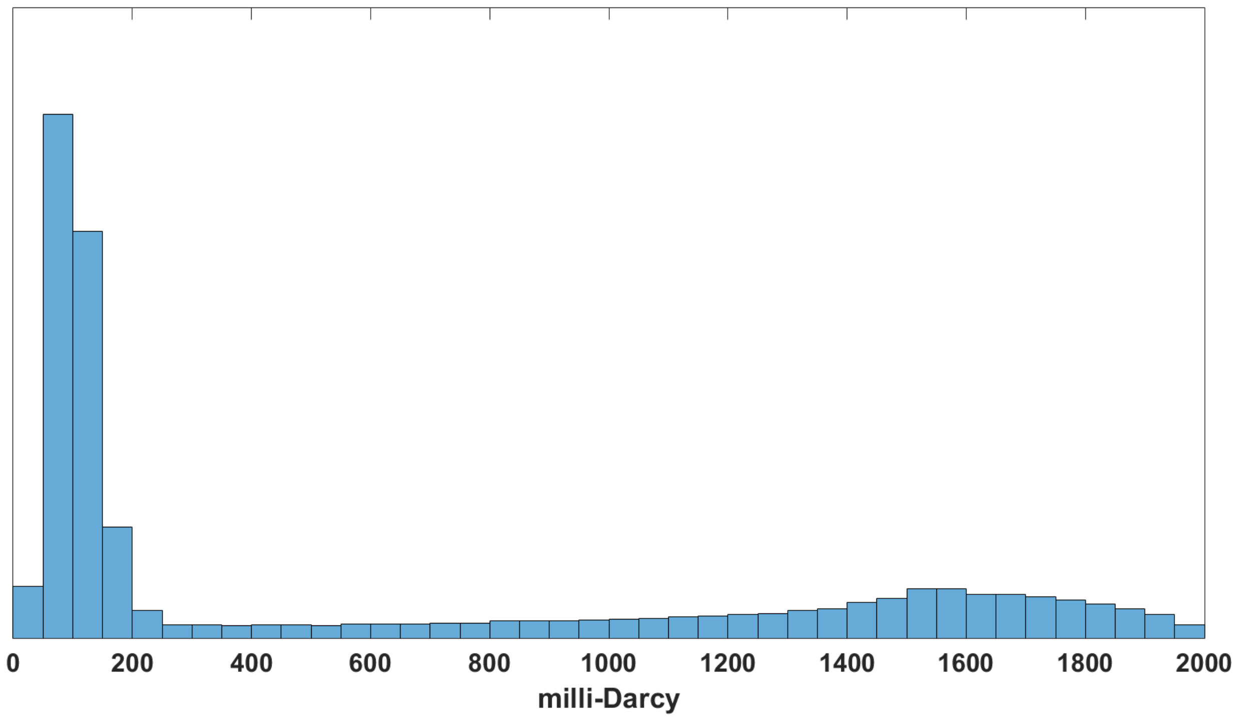


Figure 4.24: Histogram of permeability values for all realizations.

nomial approximation and the Chebyshev polynomial approximation, as described in sections 2.5 and 2.4, respectively. Again, we used the LBFGS interior-point algorithm, while we computed the gradients as describe in section 3.3.

For both parameterization methods we performed different optimization runs ranging from one to ten variables per well. That is, $N_p^{cheb} = \{1,\ldots,10\}$ and $N_u = \{1,\ldots,10\}$, where N_p^{cheb} and N_u are the number of basis functions, as described in Eqs. 2.13 and 2.2, respectively.

Figure 4.25 shows the objective function value (the mean) per each run and per each parameterization method, along with the standard deviations. As we saw in the deterministic case, the objective function increases as more variables are presented. This follow the logic that more variables provide more degrees of freedom to improve the solution. A non-trivial observation though is that the standard deviation decreases as we add more variables. Thus, increasing the number of variables might serves as a natural tool to mitigate risk, and vice versa, using a too-small number of variables might increase the risk.

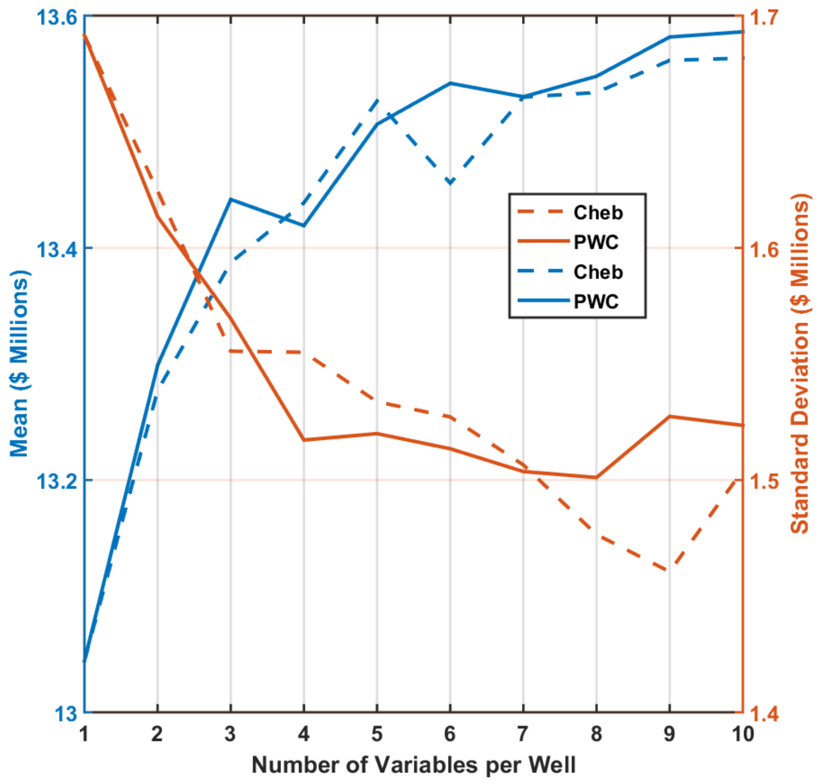


Figure 4.25: Maximizing expected NPV for all realizations: Sensitivity of number variables per well to expected (mean) and standard deviation values.

Figure 4.26 compares the optimization efficency when each parameterization method is used. Note that the number of simulation forward runs is larger than the number of iterations, as each iteration might includes several simulation runs, for example when several conjugate gradient inner iterations are required.

We observed, that in all cases, but one (5 variables per well), PWC requires more iterations up to convergence. However, in many cases Chebyshev requires more total simulation runs.

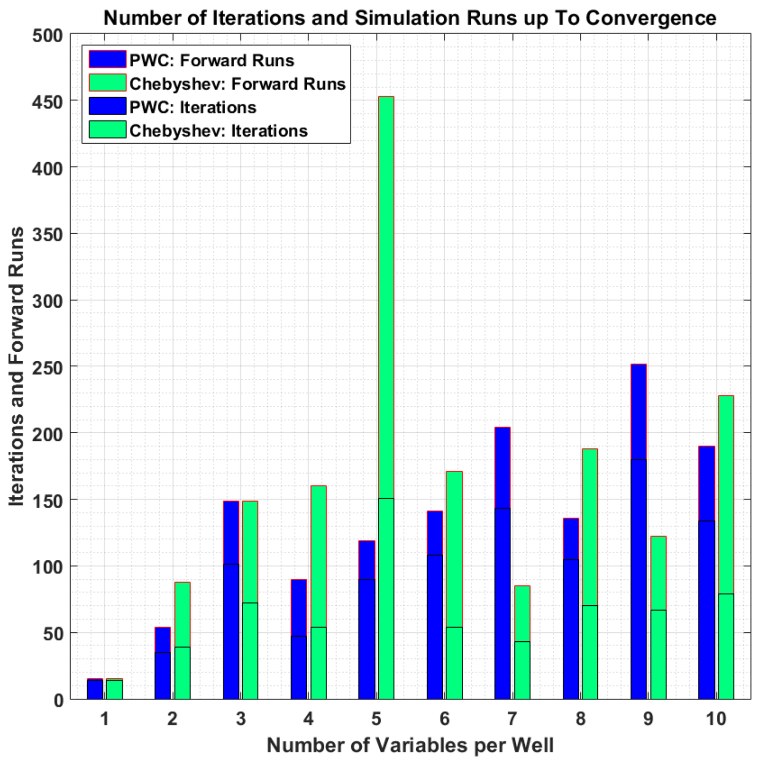


Figure 4.26: Maximizing expected NPV for all realizations: optimization efficiency.

4.4.3 Testing the Multi-Objective Function

In this section, we conducted different optimization runs to test the proposed multi-objective function. We compared different combinations for the vector (η). The motivation is to test whether we can increase the upper tail of the objective PDF while still mitigating risk.

We chose a PWC parametrization (zero order polynomials). We used 8 variables per well, since this number led to the smallest standard deviation for the expected optimization case, as we observed from Figure 4.25.

Figure 4.27 shows the results for different combinations of weights η_i. Results are given for the averages of all realizations ensemble, and of the bottom and top 20% (AVG, CVaR and CVaS, respectively). Best (MAX) and worst (MIN) results are also plotted.

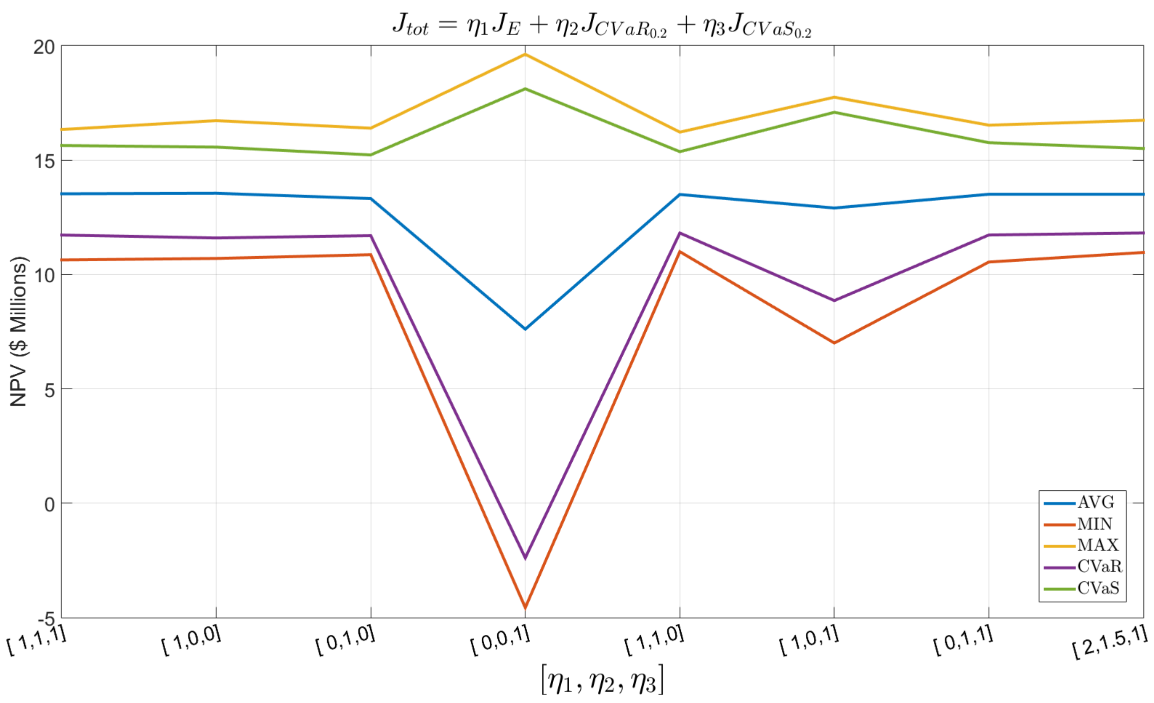


Figure 4.27: Maximizing total objective J_{tot}: results for different combinations of weights η_i. Results are given for the averages of the all ensemble of realizations, and of the bottom and top 20% (AVG, CVaR and CVaS, respectively). Best (MAX) and worst (MIN) results are also plotted.

From Figure 4.27, it's hard to tell the pros and cons of each combinations. Thus, in the following we discuss several comparison options.

First, we rank each combination according to each of the partial objectives, J_E, J_{CVaR} and J_{CVaS}. We show each one of the 3 rankings in Figure 4.28. In each ranking the scores range from 1 (left) to 8 (right). Then, in Figure 4.29 we show the accumulated ranking as a measure to compare all combinations. Note that according to this comparison measure, the proposed CVaS function is presented in all top three combinations.

To perform a more intuitive comparison, we now use a more common tool used in the petroleum industry. Rather than measures like mean, CVaR and CVaS, it is customary to describe uncertainty in terms percentiles such as P90, P50 and P10. Both the Petroleum

Resource Management System (PRMS) and the Securities and Exchange Commission (SEC) use this measures to define the reserves and resources estimates [4]. SPE convention is to quote cumulative probability of exceeding or equaling a quantity where P90 is the small estimate (conservative) and P10 is the large estimate (optimistic) [105].

Note that this measures of P90, P50 and P10 are similar to the Value-at-Risk (VaR) discussed in section 3.5.3. For example, P90 is equivalent to Value-at-Risk of 10 percent ($VaR_{10\%}$).

As also mentioned in section 3.5.3, VaR fails to account for cases found below the specified percentage. Consider the case that P90 estimate is very high and attractive, but P91 is extremely small. Albeit this obvious disadvantage, SPE still uses this measure.

Next, we specify the percentile estimates achieved with each combination η. Since we used CVaR for a confidence level of 20%, we shall indicate the P80, P50 and P20 estimates, rather than P90, P50 and P10. Note that we optimize CVaR values rather than VaR (or equivantly P90, P50, P10), and optimizing directly for the P's estimates would yield better nominal results, but with higher associated risk.

We characterized the uncertainty with 30 realizations, and thus computation of P80, P50 and P20 estimates boils down to find the NPV of the 6^{th}, 15^{th} and 24^{th} best realizations, receptively. Figure 4.30 shows the ranking of each combination η according to P80, P50 and P20 (again lowest ranking on the left and highest ranking on the right, ranging from 1 to 8). Figure 4.31 shows the accumulated ranking as a measure to compare all combinations. Note that according to this comparison measure, the purposed CVaS function is presented in top two combinations.

Figure 4.32 shows the empirical cumulative distribution function for five selected ηs. Note that $\eta = [2, 1.5, 1]$ yield the best-best case and the second best-worst case (very close to the first best-worst). Recall that $CVaR_{20\%}$ and $CVaS_{20\%}$ account for the average of the NPV values correspond to the first and last 6 stairs, respectively.

Figure 4.33 shows the probability density function, as approximated by a non-parametric Kernel Density Estimation (KDE) with MATLAB function *ksdensity* on the NPV data values (as was done in [89]). Note that η vectors that includes CVaS (with $\eta_3 = 1$) tend to drive the mode of the distribution toward lower values (to the left), which is unfavorable.

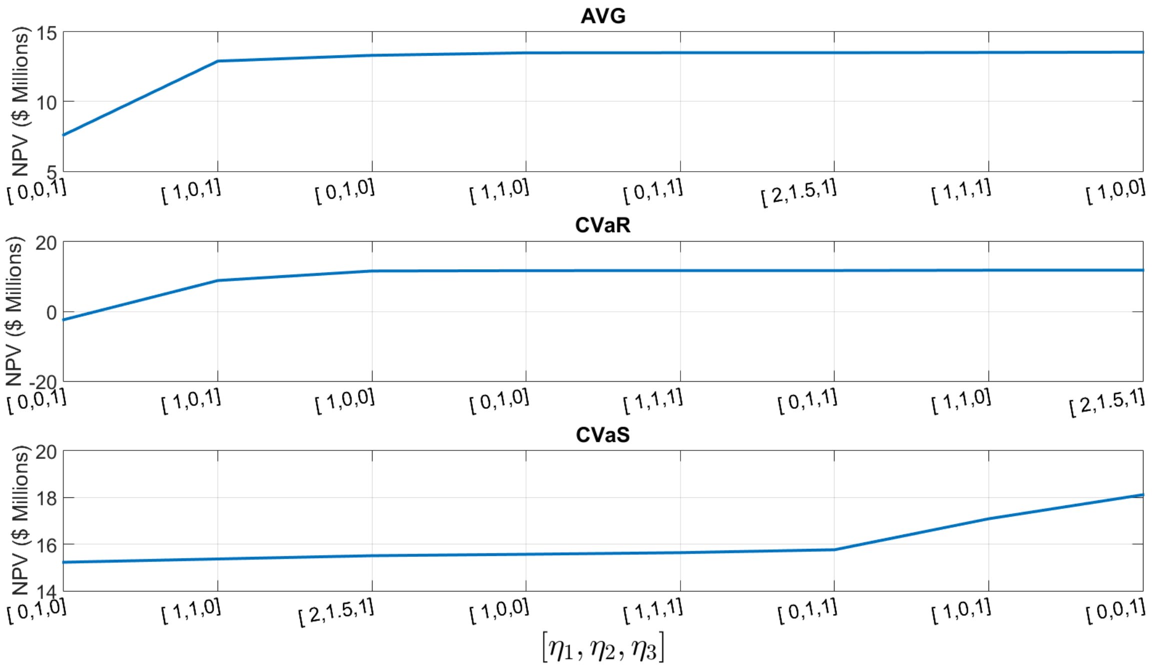


Figure 4.28: Maximizing total objective J_{tot}: Ranking of each η_i combination according to each objective function: (top) J_E, (middle) J_{CVaR}, (bottom) J_{CVaS}. Ranking score range from 1 (left) to 8 (right)

4.4.4 Optimization Under Uncertainty: Results Conclusions

In this section, we performed optimization under uncertainty described by ensemble of 30 geological realizations. We used the interior-point algorithm with LBFGS inverse Hessian approximation, and acquired the gradients with the purpose adjoint method for polynomial approximation.

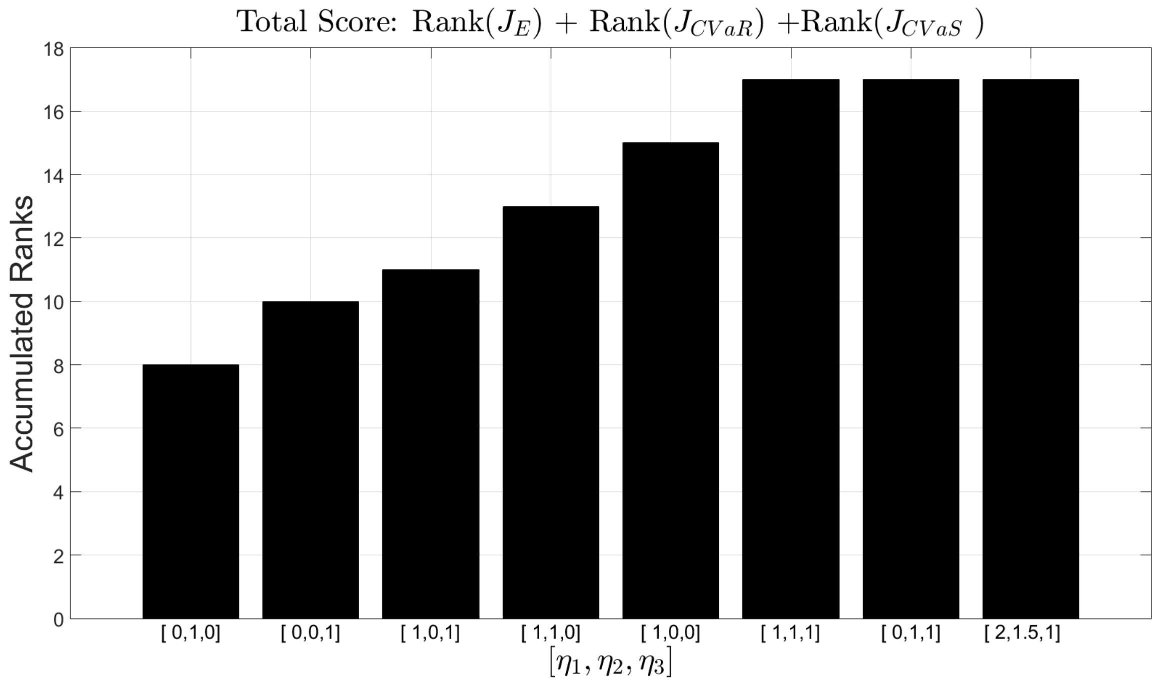


Figure 4.29: Accumulated ranking according to Fig. 4.28.

First, by maximizing only the expected value, we showed with two polynomial approximation methods, the traditional PWC and Chebyshev polynomials, that addition of more control parameters can both improve the expected NPV and reduce the standard-deviation and hence the risk.

Then, we perform sensitivity analysis to the purposed multi-objective function, which enable to consider, in addition to the expected value and the average of the lower tail, also the average of the upper tail. We called the new measure of the upper tail average as Conditional Value at Success (CVaS), and we showed that this new measure might serve as tool to improve mean, best (median or $P50$) and high estimates ($P20$). However, we observed some reduction in the probability mode (shift to the "left", see Figure 4.33).

We performed ranking to different objective's weights combinations. First, we ranked according to sub-objective values achieved by each combination, and second, we ranked

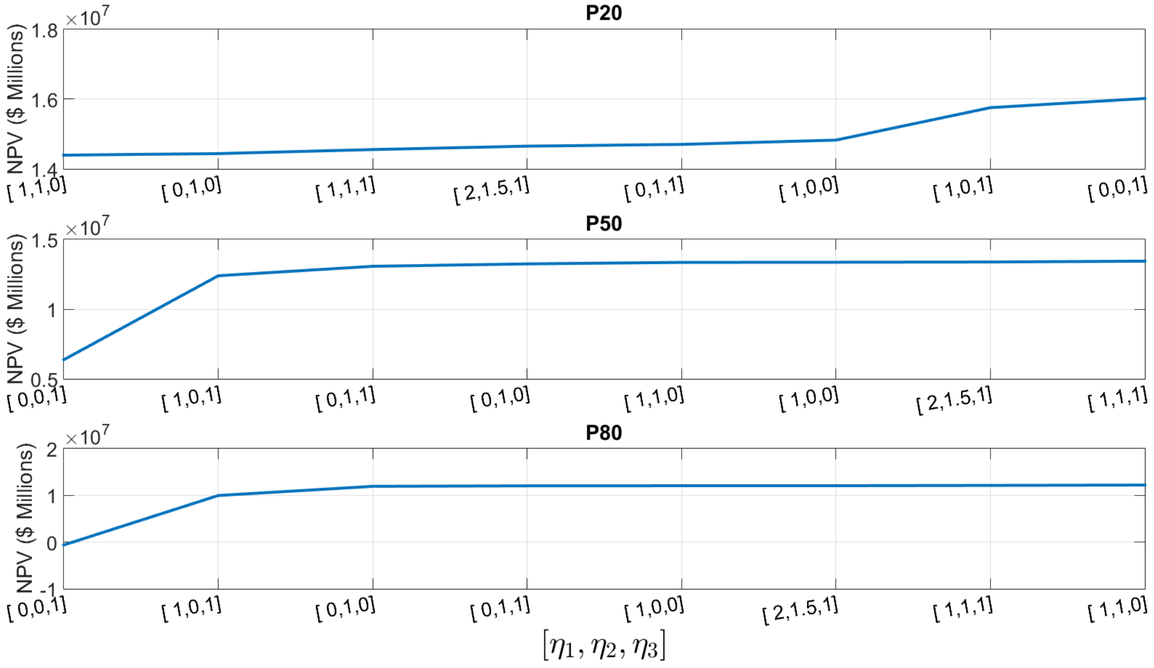


Figure 4.30: Ranking of each η_i combination according to their P percentile: (top) $P20$, (middle) $P50$, (bottom) $P80$. Ranking score range from 1 (left) to 8 (right)

according to the low, best and high estimates. We showed that the purposed CVaS participates in the best combinations.

According to the experiments performed and to the ranking methods, the combination $[1, 1, 1]$ produced the best results. Thus, we recommend on equal positive weights for all three sub-objectives.

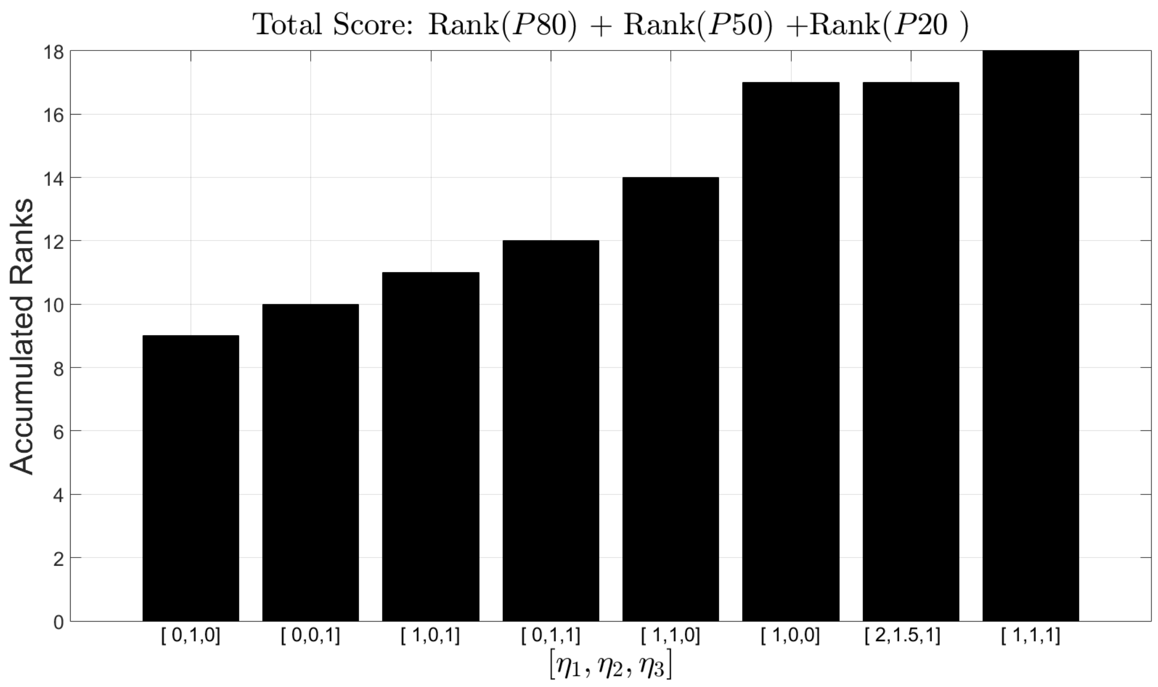


Figure 4.31: Accumulated ranking according to Fig. 4.30.

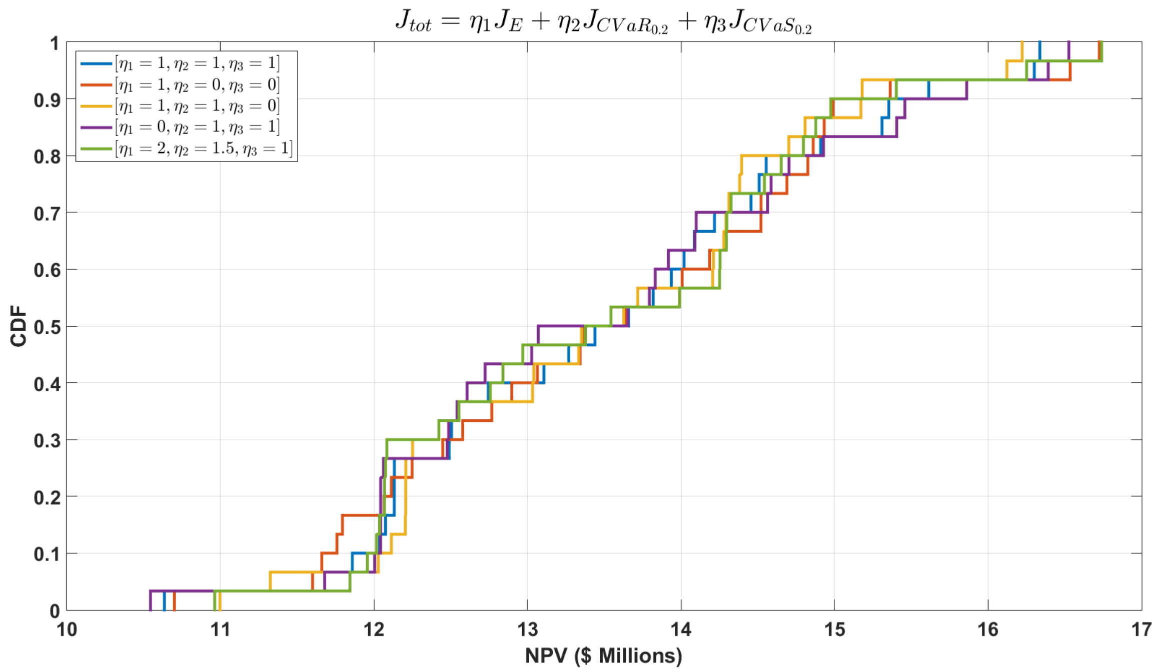


Figure 4.32: Empirical cumulative distribution function (CDF) for selected η_is combinations.

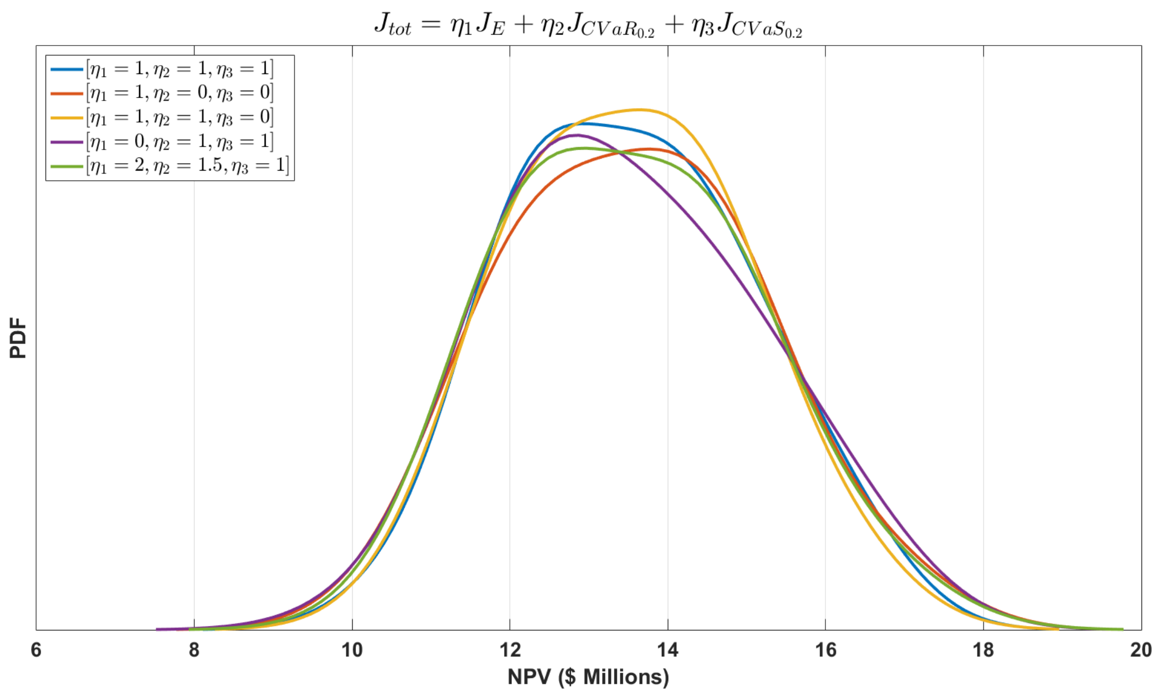


Figure 4.33: probability density function for selected η_is combinations.

5. CONCLUSIONS AND FUTURE SCOPE FOR STUDY

In this dissertation, we provided novel parametrization procedures for water-flooding production optimization problems, using polynomial approximation techniques. The methods project the original infinite dimensional controls space into a polynomial subspace.

We compared the purposed methods with global and local optimization algorithms, both by fine and reduced order modeling, and considered both deterministic and probabilistic optimization cases.

In addition to the common piece-wise zero order polynomial approximation found in most water-flooding optimization literature, we contributed with parameterization formulations using natural polynomials, orthogonal Chebyshev polynomials, and Cubic spline interpolation.

The proposed methods are well suited for the stochastic global-search method as they produce smooth control trajectories while reducing the solution space size. We demonstrated its efficiency on synthetic two-dimensional problems and a realistic 3-dimensional problem.

We also contributed with a novel workflow that combines polynomial approximation with reduced order modeling. We demonstrated a synergistic effect when combining optimization dimensionality reduction (via polynomial approximation) with model order reduction.

Using deterministic cases, with a single model realization, we performed optimization runs with both gradient-free global-search method and gradient-based local-search method. Particle Swarm Optimization serves as the global optimizer in the black-box approach. We showed that polynomial approximation serves as an efficient tool for the black-box approach executed in a high-performance computing environment.

Regarding gradient-based, we tested three algorithms: active-set, interior-point-BFGS, and interior-point-LBFGS. The last proved to be much more efficient for large scale problems than the others. By contributing with a new adjoint method formulation for polynomial approximation, we enabled the use of polynomial approximation method also to gradient-based algorithms.

We also considered optimization under uncertainty represented by multiple realizations. For that purpose, gradient-free methods are not feasible, as the high-performance computing power is devoted to simulating all realizations in parallel. Thus, the gradient-based interior-point-LBFGS algorithm was used. By maximizing only the expected value to a varying number of control parameters, we contributed with the observation that using a too small number of well-control parameters might increase the associated risk (see 4.25).

Finally, we proposed a new multi-objective function with three components, one that maximizes the expected value of all realizations, and two that maximize the average of distribution tails from both sides. The objective provides decision makers with the flexibility to choose the amount of risk they are willing to take while performing reserves estimation ($P10, P50, P90$).

5.1 Suggested Future Work

We recommend further exploration of other polynomial approximation methods. A promising candidate would be the B-spline method, discussed outside of the petroleum literature in [40]. One advantage of B-spline might be that bounds on the coefficients are well-defined in $[0,1]$, and there is no need to choose the bounds heuristically, as we did in this work. Also, a linear combination of B-spline basis functions does not violate bounds constraints, and there is no need to 'chop' the obtained control function, and thus also well-defined. The work in [40] also recommends performing orthogonalization to the basis functions, to improve the performance of gradient based methods.

To improve cases of non-continuous control functions, such that resulted, for example, from a combination of bang-bang and bang-singular control, one might investigate adaptive schemes that refine the mesh control along the optimization process. Adaptive schemes can be pursued by gradient refinement indicators [32, 34] or through wavelet analysis [49].

We also recommend on further investigation of the Chebyshev polynomial method. Specifically, it would be interesting to investigate a collocation-like method, where the simulation time steps are enforced to be calculated at Chebyshev nodes.

Finally, further study on optimization that balance between the distribution mean and both tails can shed more light on this approach. One can test range of values for α and β for $CVaR_\alpha$ and $CVaS_\beta$ (here we used only 0.2), and test unbalanced cases where $\alpha \neq \beta$.

REFERENCES

[1] M. A. S. Pinto, M. Ghasemi, N. Sorek, E. Gildin, and D. J. Schiozer, "Hybrid optimization for closed-loop reservoir management," in *SPE Reserv. Simul. Symp.*, vol. 3, pp. 1500–1510, Society of Petroleum Engineers, feb 2015.

[2] D. E. Ciaurri, T. Mukerji, and L. J. Durlofsky, "Derivative-free optimization for oil field operations," in *Computational Optimization and Applications in Engineering and Industry*, pp. 19–55, Springer, 2011.

[3] Y. Choi, W. Murray, and P. Avery, "Pde-constrained optimization and beyond," 2011.

[4] PetroWiki, "Uncertainty range in production forecasting."

[5] D. Brouwer and J.-D. Jansen, "Dynamic optimization of waterflooding with smart wells using optimal control theory," *SPE J.*, vol. 9, pp. 391–402, dec 2004.

[6] K. Aziz and A. Settari, *Petroleum reservoir simulation*. Chapman & Hall, 1979.

[7] Z. Chen, G. Huan, and Y. Ma, *Computational methods for multiphase flows in porous media*. SIAM, 2006.

[8] T. Ertekin, J. H. Abou-Kassen, and G. R. King, *Basic Applied Reservoir Simulations*. Society of Petroleum Engineers, 2001.

[9] J. D. Jansen, *A systems description of flow through porous media*. Springer, 2013.

[10] K. A. Lie, " An introduction to reservoir simulation using MATLAB: User guide for the Matlab reservoir simulation toolbox (MRST).," 2016.

[11] P. LeGresley and J. Alonso, “Airfoil design optimization using reduced order models based on proper orthogonal decomposition,” in *Fluids 2000 Conference and Exhibit*, p. 2545, 2000.

[12] S. Lall, J. E. Marsden, and S. Glavaški, “A subspace approach to balanced truncation for model reduction of nonlinear control systems,” *International journal of robust and nonlinear control*, vol. 12, no. 6, pp. 519–535, 2002.

[13] T. Lieu, C. Farhat, and M. Lesoinne, “Pod-based aeroelastic analysis of a complete f-16 configuration: Rom adaptation and demonstration,” in *46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, p. 2295, 2005.

[14] J. F. van Doren, R. Markovinović, and J.-D. Jansen, “Reduced-order optimal control of water flooding using proper orthogonal decomposition,” *Computational Geosciences*, vol. 10, no. 1, pp. 137–158, 2006.

[15] M. A. Cardoso, L. J. Durlofsky, *et al.*, “Use of reduced-order modeling procedures for production optimization,” *SPE Journal*, vol. 15, no. 02, pp. 426–435, 2010.

[16] J. He, L. J. Durlofsky, *et al.*, “Reduced-order modeling for compositional simulation by use of trajectory piecewise linearization,” *SPE Journal*, vol. 19, no. 05, pp. 858–872, 2014.

[17] M. Fragoso, B. Horowitz, and R. Jose Roberto P., “Retraining criteria for TPWL/POD surrogate based waterflodding optimization,” in *SPE Reserv. Simul. Symp.*, pp. 23–25, Society of Petroleum Engineers, feb 2015.

[18] S. Trehan and L. J. Durlofsky, “Trajectory piecewise quadratic reduced-order model for subsurface flow, with application to pde-constrained optimization,” *Journal of Computational Physics*, vol. 326, pp. 446–473, 2016.

[19] M. Ghasemi, Y. Yang, E. Gildin, Y. Efendiev, and V. Calo, "Fast multiscale reservoir simulations using POD-DEIM model reduction," in *SPE Reserv. Simul. Symp.*, pp. 23–25, Society of Petroleum Engineers, feb 2015.

[20] Y. Yang, M. Ghasemi, E. Gildin, Y. Efendiev, V. Calo, *et al.*, "Fast multiscale reservoir simulations with pod-deim model reduction," *SPE Journal*, 2016.

[21] D. Brouwer, J. Jansen, *et al.*, "Dynamic optimization of water flooding with smart wells using optimal control theory," in *European Petroleum Conference*, Society of Petroleum Engineers, 2002.

[22] M. Zandvliet, O. Bosgra, J. Jansen, P. V. den Hof, and J. Kraaijevanger, "Bang-bang control and singular arcs in reservoir flooding," *J. Petrol. Sci. Eng.*, vol. 58, no. 1âĂŞ2, pp. 186 – 200, 2007.

[23] D. Liberzon, *Calculus of variations and optimal control theory: a concise introduction*. Princeton University Press, 2012.

[24] L. T. Biegler, "An overview of simultaneous strategies for dynamic optimization," *Chemical Engineering and Processing: Process Intensification*, vol. 46, no. 11, pp. 1043 – 1053, 2007. Special Issue on Process Optimization and Control in Chemical Engineering and Processing.

[25] B. Houska and B. Chachuat, "Branch-and-lift algorithm for deterministic global optimization in nonlinear optimal control," *Journal of Optimization Theory and Applications*, vol. 162, no. 1, pp. 208–248, 2014.

[26] J. Åkesson, "Overview of Direct Methods for Dynamic OptimizationâĂŤCollocation Overview of Direct Methods for Dynamic Optimization,"

[27] A. Codas Duarte, B. Foss, E. Camponogara, *et al.*, "Output-constraint handling and parallelization for oil-reservoir control optimization by means of multiple shoot-

ing," *SPE J.*, vol. 20, no. 04, pp. 856 – 871, 2015.

[28] C. Wang, G. Li, and A. C. Reynolds, "Production optimization in closed-loop reservoir management," *SPE J.*, vol. 14, pp. 506–523, sep 2009.

[29] P. Sarma, L. J. Durlofsky, K. Aziz, and W. H. Chen, "Efficient real-time reservoir management using adjoint-based optimal control and model updating," *Comput. Geosci.*, vol. 10, pp. 3–36, may 2006.

[30] D. Kourounis, L. J. Durlofsky, J. D. Jansen, and K. Aziz, "Adjoint formulation and constraint handling for gradient-based optimization of compositional reservoir flow," *Comput. Geosci.*, vol. 18, no. 2, pp. 117–137, 2014.

[31] G. van Essen, P. Van den Hof, and J.-D. Jansen, "Hierarchical long-term and short-term production optimization," *SPE J.*, vol. 16, pp. 191–199, mar 2011.

[32] M. Lien, D. Brouwer, T. Mannseth, and J.-D. Jansen, "Multiscale regularization of flooding optimization for smart field management," *SPE J.*, vol. 13, no. 2, pp. 195–204, 2008.

[33] Y. Shuai, C. D. White, H. Zhang, and T. Sun, "Using multiscale regularization to obtain realistic optimal control strategies," in *SPE Reserv. Simul. Symp.*, no. February, pp. 21–23, Society of Petroleum Engineers, apr 2011.

[34] A. C. Reynolds and D. Oliveira, "An adaptive hierarchical algorithm for estimation of optimal well controls," in *SPE Reserv. Simul. Symp.*, no. February, pp. 1–26, 2013.

[35] S. Krogstad, K. A. Lie, O. Møyner, H. Møll Nilsen, X. Raynaud, ard Skaflestad, and S. Ict, "Spe 173317-ms mrst-ad âĂŞ an open-source framework for rapid prototyping and evaluation of reservoir simulation problems," *SPE Reserv. Simul. Symp.*, pp. 23–25, 2015.

[36] K. A. Lie, S. Krogstad, I. S. Ligaarden, J. Roald, N. . Halvor, M. Nilsen, and B. Skaflestad, “Open-source MATLAB implementation of consistent discretisations on complex grids,” *Comput. Geosci.*, vol. 16, pp. 297–322, 2012.

[37] J.-D. Jansen, R. Brouwer, and S. G. Douma, “Closed loop reservoir management,” in *SPE Reserv. Simul. Symp.*, pp. 2–4, Society of Petroleum Engineers, apr 2009.

[38] H. Zhao, C. Chen, S. Do, D. Oliveira, G. Li, and A. Reynolds, “Maximization of a dynamic quadratic interpolation model for production optimization,” *SPE J.*, vol. 18, no. 06, pp. 1012–1025, 2013.

[39] B. E. Usevitch, “A tutorial on modern lossy wavelet image compression: foundations of jpeg 2000,” *IEEE signal processing magazine*, vol. 18, no. 5, pp. 22–35, 2001.

[40] A. L. Schwartz, *Theory and implementation of numerical methods based on Runge-Kutta integration for solving optimal control problems*. PhD thesis, University of California at Berkeley, 1996.

[41] E. W. Cheney, *Multivariate approximation theory: Selected topics*. SIAM, 1986.

[42] A. F. Timan, *Theory of approximation of functions of a real variable*, vol. 34. Elsevier, 2014.

[43] L. N. Trefethen, *Approximation theory and approximation practice*. Siam, 2013.

[44] G. H. Hardy, “WeierstrassâĂŹs non-differentiable function,” *Trans. Amer. Math. Soc*, vol. 17, no. 3, pp. 301–325, 1916.

[45] B. HOUSKA and B. CHACHUAT, “Global optimization in hilbert space,”

[46] G. K. Smyth, “Polynomial approximation,” *Encyclopedia of Biostatistics*, 1998.

[47] L. T. Biegler, "Solution of dynamic optimization problems by successive quadratic programming and orthogonal collocation," *Computers & chemical engineering*, vol. 8, no. 3-4, pp. 243–247, 1984.

[48] J. Vlassenbroeck, "A chebyshev polynomial method for optimal control with state constraints," *Automatica*, vol. 24, no. 4, pp. 499–506, 1988.

[49] M. Schlegel, K. Stockmann, T. Binder, and W. Marquardt, "Dynamic optimization using adaptive control vector parameterization," *Computers & Chemical Engineering*, vol. 29, no. 8, pp. 1731–1751, 2005.

[50] A. Ghosh, A. Chowdhury, R. Giri, S. Das, and A. Abraham, "A hybrid evolutionary direct search technique for solving optimal control problems," in *Hybrid Intelligent Systems (HIS), 2010 10th International Conference on*, pp. 125–130, IEEE, 2010.

[51] C. De Boor, "Applied mathematical sciences 27, a practical guide to splines, revised version," 2001.

[52] D. R. Kincaid and E. W. Cheney, *Numerical analysis: mathematics of scientific computing*, vol. 2. American Mathematical Soc., 2002.

[53] A. A. Awotunde, "On the joint optimization of well placement and control," in *SPE Saudi Arabia Sect. Tech. Symp. Exhib.*, Society of Petroleum Engineers, apr 2014.

[54] B. Sudaryanto and Y. C. Yortsos, "Optimization of fluid front dynamics in porous media using rate control. I. Equal mobility fluids," *Phys. Fluids*, vol. 12, p. 1656, 2000.

[55] B. Sudaryanto and Y. C. Yortsos, "Optimization of displacements in porous media using rate control," in *SPE Annu. Tech. Conf. Exhib.*, Society of Petroleum Engineers, apr 2001.

[56] L. T. Biegler, A. M. Cervantes, and A. Wachter, “Advances in simultaneous strategies for dynamic process optimization,” *Chem. Eng. Sci.*, vol. 57, no. 4, pp. 575 – 593, 2002.

[57] R. Fonseca, O. Leeuwenburgh, P. Van den Hof, and J.-D. Jansen, “Improving the ensemble-optimization method through covariance-matrix adaptation,” *SPE J.*, vol. 20, pp. 155–168, feb 2014.

[58] D. Echeverria Ciaurri, O. J. Isebor, and L. J. Durlofsky, “Application of derivative-free methodologies to generally constrained oil production optimisation problems,” *Int. J. Math. Model. Num. Optim.*, vol. 2, no. 2, pp. 134–161, 2011.

[59] R. C. Eberhart, J. Kennedy, *et al.*, “A new optimizer using particle swarm theory,” in *Proceedings of the sixth international symposium on micro machine and human science*, vol. 1, pp. 39–43, New York, NY, 1995.

[60] F. Boukouvala, M. M. F. Hasan, and C. A. Floudas, “Global optimization of general constrained grey-box models: new method and its application to constrained pdes for pressure swing adsorption,” *J. Global Optim.*, pp. 1–40, 2015.

[61] F. Boukouvala and C. A. Floudas, “ARGONAUT: Algorithms for global optimization of constrained grey-box computational problems,” *Optim. Lett.*, pp. 1 – 19, 2014.

[62] O. Volkov and M. C. Bellout, “Gradient-based production optimization with simulation-based economic constraints,” *Computational Geosciences*, pp. 1–18.

[63] O. J. Isebor, D. EcheverrÄś, and L. J. Durlofsky, “Generalized field-development optimization with derivative-free procedures,” *SPE J.*, vol. 19, no. 05, pp. 891–908, 2014.

[64] S. Navabi, R. Khaninezhad, and B. Jafarpour, “A generalized formulation for oilfield development optimization,” *IFAC-PapersOnLine*, vol. 48, no. 6, pp. 56–61, 2015.

[65] M. Ghommem, E. Gildin, and M. Ghasemi, “Complexity reduction of multiphase flows in heterogeneous porous media,” *SPE J.*, vol. 21, pp. 144–151, feb 2016.

[66] J. He and L. J. Durlofsky, “Reduced-order modeling for compositional simulation by use of trajectory piecewise linearization,” *SPE J.*, vol. 19, pp. 858–872, oct 2014.

[67] R. W. de Holanda, E. Gildin, and J. L. Jensen, “Improved waterflood analysis using the capacitance-resistance model within a control systems framework,” in *SPE Lat. Am. Caribb. Pet. Eng. Conf.*, Society of Petroleum Engineers, nov 2015.

[68] V. Bukshtynov, O. Volkov, L. J. Durlofsky, and K. Aziz, “Comprehensive framework for gradient-based optimization in closed-loop reservoir management,” *Computational Geosciences*, vol. 19, no. 4, pp. 877–897, 2015.

[69] G. Gao and A. C. Reynolds, “An improved implementation of the LBFGS algorithm for automatic history matching,” *SPE J.*, vol. 11, pp. 5–17, mar 2006.

[70] C. De Boor, “Convergence of cubic spline interpolation with the not-a-knot condition.,” tech. rep., DTIC Document, 1985.

[71] G. Behforooz, “The not-a-knot piecewise interpolatory cubic polynomial,” *Appl. Math. Comput.*, vol. 52, no. 1, pp. 29–35, 1992.

[72] X. Hu, “Particle swarm optimization,” 2006.

[73] MathWorks, “Constrained nonlinear optimization algorithms,” 2016.

[74] MathWorks, “Particle swarm optimization algorithm,” 2015.

[75] schlumberger, *ECLIPSE, Reservoir Simulation Software, Technical Description*. schlumberger.

[76] S. Krogstad, V. L. Hauge, and A. Gulbransen, “Adjoint multiscale mixed finite elements,” *SPE Journal*, vol. 16, pp. 162–171, mar 2011.

[77] R. F. Stengel, *Optimal control and estimation*. Courier Corporation, 2012.

[78] E. Suwartadi, S. Krogstad, and B. Foss, “Nonlinear output constraints handling for production optimization of oil reservoirs,” *Computational Geosciences*, vol. 16, pp. 499–517, sep 2011.

[79] S. Krogstad, X. Raynaud, and H. M. Nilsen, “Reservoir management optimization using well-specific upscaling and control switching,” *Computational Geosciences*, vol. 20, pp. 695–706, may 2015.

[80] S. Chaturantabut and D. C. Sorensen, “Discrete empirical interpolation for nonlinear model reduction,” in *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pp. 4316–4321, IEEE, 2009.

[81] J. D. Jansen and L. J. Durlofsky, “Use of reduced-order models in well control optimization,” *Optimization and Engineering*, pp. 1–28, 2016.

[82] I. E. Grossmann, R. M. Apap, B. A. Calfa, P. Garcia-Herreros, and Q. Zhang, “Recent advances in mathematical programming techniques for the optimization of process systems under uncertainty,” *Computers & Chemical Engineering*, vol. 91, pp. 3–14, 2016.

[83] N. V. Sahinidis, “Optimization under uncertainty: state-of-the-art and opportunities,” *Computers & Chemical Engineering*, vol. 28, no. 6, pp. 971–983, 2004.

[84] A. Capolei, E. Suwartadi, B. Foss, and J. B. Jørgensen, “Waterflooding optimization in uncertain geological scenarios,” *Computational Geosciences*, vol. 17, no. 6, pp. 991–1013, 2013.

[85] G. van Essen, M. Zandvliet, P. Van den Hof, O. Bosgra, J.-D. Jansen, *et al.*, "Robust waterflooding optimization of multiple geological scenarios," *SPE Journal*, vol. 14, no. 01, pp. 202–210, 2009.

[86] M. M. Siraj, P. M. Van den Hof, and J. D. Jansen, "Robust optimization of water-flooding in oil reservoirs using risk management tools," *IFAC-PapersOnLine*, vol. 49, no. 7, pp. 133–138, 2016.

[87] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust optimization*. Princeton University Press, 2009.

[88] A. Capolei, E. Suwartadi, B. Foss, and J. B. Jørgensen, "A mean–variance objective for robust production optimization in uncertain geological scenarios," *Journal of Petroleum Science and Engineering*, vol. 125, pp. 23–37, 2015.

[89] M. M. Siraj, P. M. Van den Hof, and J. D. Jansen, "Risk management in oil reservoir water-flooding under economic uncertainty," in *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*, pp. 7542–7547, IEEE, 2015.

[90] B. Chen, R.-M. Fonseca, O. Leeuwenburgh, and A. C. Reynolds, "Minimizing the risk in the robust life-cycle production optimization using stochastic simplex approximate gradient," *Journal of Petroleum Science and Engineering*, 2017.

[91] D. M. Valladao, R. R. Torrado, B. Flach, S. Embid, *et al.*, "On the stochastic response surface methodology for the determination of the development plan of an oil & gas field," in *SPE Middle East Intelligent Energy Conference and Exhibition*, Society of Petroleum Engineers, 2013.

[92] R. T. Rockafellar and S. Uryasev, "Conditional value-at-risk for general loss distributions," *Journal of banking & finance*, vol. 26, no. 7, pp. 1443–1471, 2002.

[93] S. Sarykalin, G. Serraino, and K. Kalinchenko, "Stan Uryasev Joint presentation with VaR vs CVaR in Risk Management and Optimization,"

[94] K.-A. Lie, "An introduction to reservoir simulation using matlab: User guide for the matlab reservoir simulation toolbox (mrst)," *SINTEF ICT, May*, 2016.

[95] A. T. Gaspar, G. D. Avansi, A. A. d. S. dos Santos, J. C. v. H. Filho, and D. J. Schiozer, "UNISIM-I-D: Benchmark studies for oil field development and production strategy selection," *Int. J. Model. Simul. Pet. Ind.*, vol. 9, no. 1, 2015.

[96] G. Bacoccoli, R. G. Morales, and O. A. J. Campos, "The Namorado oil field: a major oil discovery in the Campos Basin, Brazil," in *Giant Oil Gas Fields Decad. 1968-1978*, vol. 30, pp. 329–338, 1980.

[97] G. D. Avansi and D. J. Schiozer, "UNISIM-I: Synthetic model for reservoir development and management applications," *Int. J. Model. Simul. Pet. Ind.*, vol. 9, no. 1, pp. 21–30, 2015.

[98] UNICAMP, "UNISIM-I: Benchmark case," 2016.

[99] D. C. Doublet, S. I. Aanonsen, and X.-C. Tai, "An efficient method for smart well production optimisation," *J. Pet. Sci. Eng.*, vol. 69, no. 1, pp. 25–39, 2009.

[100] Y. Chen, D. S. Oliver, and D. Zhang, "Efficient ensemble-based closed-loop production optimization," in *SPE Symp. Improv. Oil Recover.*, Society of Petroleum Engineers, apr 2008.

[101] M. C. Bellout, D. E. Ciaurri, L. J. Durlofsky, B. Foss, J. Kleppe, M. C. Bellout, J. Kleppe, D. E. Ciaurri, L. J. Durlofsky, and B. Foss, "Joint optimization of oil well placement and controls," *Comput. Geosci.*, vol. 16, pp. 1061–1079, 2012.

[102] D. S. Oliver, A. C. Reynolds, and N. Liu, "Optimization for nonlinear problems using sensitivities," in *Inverse Theory Pet. Reserv. Charact. Hist. Matching*, ch. 8, pp. 143–192, 2008.

[103] F. Zhang and A. C. Reynolds, "Optimization algorithms for automatic history matching of production data," in *ECMOR VIII-8th European Conference on the Mathematics of Oil Recovery*, 2002.

[104] M. G. Shirangi and L. J. Durlofsky, "Closed-loop field development under uncertainty by use of optimization with sample validation," *SPE J.*, vol. 20, no. 05, pp. 908–922, 2015.

[105] "Petroleum Resources Management System," tech. rep., Society of Petroleum Engineers (SPE) American Association of Petroleum Geologists (AAPG) World Petroleum Council (WPC) Society of Petroleum Evaluation Engineers (SPEE).

[106] M. Hinze and S. Volkwein, "Proper orthogonal decomposition surrogate models for nonlinear dynamical systems: Error estimates and suboptimal control," in *Dimension reduction of large-scale systems*, pp. 261–306, Springer, 2005.

APPENDIX A

MODEL ORDER REDUCTION: POD-DEIM

A.1 Proper Orthogonal Decomposition (POD)

As a first step to generate reduced order model, the full order system is solved to generate a matrix of states called the snapshot matrix, X.

$$X = [x^1, x^2, x^3, \ldots, x^{N_{sim}}], X \in \mathbb{R}^{n \cdot N_{sim}} \quad , \tag{A.1}$$

where n is the full order dimension of each state (here pressure and saturation), and N_{sim} is the total number of snapshots (at most equal to the total time steps). The snapshot matrix is computed by numerically solving the fine scale system of nonlinear Eqs. 1.1. The idea here is to project a fine scale system to a low dimensional space.

In POD, the orthonormal basis $\{\Phi_i\}_i^r$ is obtained by the solution of the minimization problem:

$$\min_{\Phi_i} \quad \left\| x_j - \sum_{i=1}^{r} (x_j^T \Phi_i) \Phi_i. \right\| \tag{A.2}$$

The solution to this minimization problem [106] is obtained by applying the *SVD* to the snapshot matrix and then selecting the first r columns of the left projection matrix.

$$X = U \Lambda V^T, \tag{A.3}$$

where U and V are the left and right singular matrices and Λ is a diagonal matrix with eigenvalues in decreasing order. The r columns of U are selected by indication of the fraction of energy to be captured $\mathbb{E}$:

$$\mathbb{E} = \frac{\sum_{i=1}^{r} \sigma_i}{\sum_{i=1}^{N_{sim}} \sigma_i}, \tag{A.4}$$

where σ_i is the i^{th} diagonal element of Λ. The number of columns are usually chosen such that $0.9 < \mathbb{E} < 1$. Thus the *SVD* of pressure and saturation snapshot matrices gives the projection matrix for each variable and they can be projected to the reduced states by,

$$p \approx \Phi_p p_r, \tag{A.5}$$

$$s \approx \Phi_s s_r. \tag{A.6}$$

Thus, we may write,

$$\begin{bmatrix} p \\ s \end{bmatrix} = \begin{bmatrix} \Phi_p & 0 \\ 0 & \Phi_s \end{bmatrix} \begin{bmatrix} p_r \\ s_r \end{bmatrix} \tag{A.7}$$

which take the short notation:

$$x = \Phi x_r, \quad r \ll n. \tag{A.8}$$

The reservoir simulation involves solving a system of linear equations as given in Eqs. 1.3 and 1.4, which for convenience of reading is also given below:

$$J^{n+1}\delta^{n+1} = -R^{n+1}$$

$$x^{n+1} = x^n + \delta^{n+1}.$$

Thus, using Galerkin projection, this equation can be written as follows:

$$\Phi^T J^{n+1} \Phi \delta_r^{n+1} = -\Phi^T R^{n+1}, \tag{A.9}$$

which again take a short notation:

$$J_r^{n+1} \delta_r^{n+1} = -\Phi^T R^{n+1}, \tag{A.10}$$

and after solving this reduced linear problem, the reduced state space variable is updated as follow:

$$x_r^{n+1} = x_r^n + \delta_r^{n+1}, \tag{A.11}$$

POD-Galerkin reduces the computation cost moderately for nonlinear systems since it requires computation of the fine scale Jacobian and Residual. Thus, we seek a technique that avoids projecting back to the fine scale. This is done by DEIM that constructs a separate subspace of nonlinear terms, selects interpolation points by greedy algorithm, and then approximate the nonlinear terms in the subspace by a combination of projection and interpolation [80]. The description of DEIM proposed by these authors is explained briefly below.

A.2 Discrete Empirical Interpolation (DEIM)

Let $f(t) \in \mathbb{R}^n$ be a nonlinear function of time or other parameter. f can be approximated by projecting it into a subspace spanned by the basis function $U = (U_1, U_2, \ldots, U_m) \in \mathbb{R}^{n \cdot m}$ as

$$f(t) \approx Uc(t). \tag{A.12}$$

This basis function is determined by assembling the function evaluations in a matrix

$S_f \in \mathbb{R}^{n \cdot N_{sim}}$ and then *SVD* is employed to this matrix to compute m modes that are used as the basis functions.

The coefficient vector $c(t)$ can be determined uniquely from the following:

$$P^T f(t) = (P^T U)c(t), \tag{A.13}$$

where, $P = [e_{\gamma 1}, e_{\gamma 2}, \ldots, e_{\gamma m}] \in \mathbb{R}^{n \cdot m}$, and $e_{\gamma i} = [0, \ldots, 0, 1, 0, \ldots, 0]^T \in \mathbb{R}^n$ is the γi^{th} column of the identity matrix $I_n \in \mathbb{R}^{n \cdot n}$. These interpolation indices P used for determining the coefficient vector $c(t)$ are selected inductively from the basis U by the greedy algorithm to have the non-singular matrix $P^T U$. The greedy algorithm is as follows:

```
DEIM GREEDY ALGORITHM:
INPUT     $u_{l}{}_{l=1}^{m} \subset \mathbb{R}^n$ linearly independent
OUTPUT    $\vec{\gamma} = [\gamma 1, \gamma 2, \ldots, \gamma m] \in \mathbb{R}^m$
1. $[|\rho|, \gamma 1] = max\{|u1|\}$
2. $U = [u1], P = [e_{\gamma 1}], \vec{\gamma} = [\gamma 1]$
3. for $l = 2$ to m do
4.     Solve $(P^T U)c = P^T u_l$ for c
5.     $r = u_l - Uc$
6.     $[|\rho|, \gamma l] = max\{|r|\}$
7.     $U \leftarrow [U, u_l]$,   $P \leftarrow [P, e_{\gamma l}]$,   $\vec{\gamma} \leftarrow [\gamma l, \vec{\gamma}]^T$
8. end for
```

Thus from Eqs. A.12 and A.13 we may write

$$f(t) \approx U(P^T U)^{-1} P^T f(t) \tag{A.14}$$

The nonlinear functions (in our case the Transmissibility, Gravity, Accumulation, Source/Sink, Jacobian and the Residual) are thus computed only at the selected interpolation points by the greedy algorithm ($P^T f(t)$) and then approximated in fine scale by the above expression. For in depth discussions on the applications of POD-DEIM in reservoir simulation, the reader is referred to [20].

APPENDIX B

INCOMPRESSIBLE SEQUENTIAL SOLVER

In the majority of this work we used the sequential solver in Matlab Reservoir Simulation Toolbox (MRST) for two-phase incompressible flow of a wetting and non-wetting fluids [36]. By neglecting gravity effects, the PDE can be described as follow:

$$\nabla \cdot \vec{v} = q, \vec{v} = -\mathbf{K}(\lambda \nabla p) \tag{B.1}$$

$$\phi \frac{\partial s_w}{\partial t} + \nabla \cdot (f_w(s_w)\vec{v}) = q_w, \tag{B.2}$$

where $\vec{v}$ denotes the Darcy velocity, q total liquid flux, p pressure, s_w water saturation, $\mathbf{K}$ permeability tensor, λ the (total) mobility, $f_w(s_w)$ the water fractional flow as a function of saturation, ϕ porosity and t time. First, pressure and fluxes are provided by fixing the saturation values and solving Eq. B.1. Subsequently, fluxes are used in Eq. B.2 to solve for the saturation in the next simulation time step. For further details the reader is referred to [36, 10].